Collaborative Sensor Network Algorithm for Predicting the Spatiotemporal Evolution of Hazardous Phenomena

Dimitris V. Manatakis and Elias S. Manolakos

Department of Informatics and Telecommunications,
University of Athens

{dmanatak, eliasm}@di.uoa.gr

***Abstract*— We present a novel decentralized Wireless Sensor Network (WSN) algorithm which can estimate both the speed and direction of an evolving diffusive hazardous phenomenon (e.g. a wildfire, oil spill, etc.). In the proposed scheme we approximate a progressing hazard's front as a set of line segments. The spatiotemporal evolution of each line segment is modeled by a modified 2D Gaussian function. As the phenomenon evolves, the parameters of this model are updated based on the analytical solution of a Kullback – Leibler (KL) divergence minimization problem. This leads to an efficient WSN distributed parameters estimation algorithm that can be implemented by dynamically formed clusters (triplets) of collaborating sensor nodes. Computer simulations show that our approach is able to track the evolving phenomenon with reasonable accuracy even if a percentage of sensors fails due to the hazard and/or the phenomenon has a time varying speed.**

Keywords- Environmenal hazard; predictive modeling; WSN; Spatio-temporal evolution; Kullback-Leibler divergence

I. INTRODUCTION

Predicting with reasonable accuracy the spatiotemporal evolution of a diffusive hazardous phenomenon (such as a wild fires, tsunamis, oil slick, etc.) is of paramount importance since it helps the authorities to organize efficiently and effectively their response (hazard suppression, possible evacuations etc.). Research efforts around the globe are focusing in developing hazard-specific predictive models. However, most of these mechanistic models depend on a large number of space- and time-varying parameters which are difficult or even impossible, to estimate in real time, making model predictions deviate significantly from reality. To address this limitation many researchers have proposed architectures which attempt to integrate recent sensor measurements and simulation based predictive modeling into closed loop systems. Most of these works rely on remote sensing, i.e. measured data (e.g. satellite spectral images) are used to calibrate periodically simulation models in order to minimize model prediction errors. These methods, also known as Dynamic Data Driven Assimilation, have recently drawn the attention of the scientific community due to their expected high societal impact [1-4].

Unfortunately, in many cases, satellite images, or image data in general, is not available, or may be inappropriate for detecting a certain diffusing hazard. In such cases, Wireless Sensor Networks (WSNs), that are becoming a mature state of the art technology due to their rapidly dropping cost, may provide a viable alternative for environmental monitoring applications, especially if the quality of the results they provide does not heavily depend on their node density (sensors per square kilometer).

Recently, a number of collaborative WSN-based methods have been proposed for detecting the boundary line of a diffusing hazard [5-7]. Their main objective is to identify at each time step the sensor nodes located closest to the evolving hazard's front line. Despite their demonstrated capability to delineate the area affected by a hazard these schemes suffer by construction from the severe limitation that their achieved accuracy is proportional to the WSN nodes density (requiring thousands of sensors per square kilometer), which renders them impractical today even for small-scale environmental monitoring applications. Furthermore, since they do not provide any information about the evolution characteristics of the hazard (e.g. direction and speed) they cannot be used for decision support based on predictive modeling.

A hazard's front line can be approximated as a piecewise linear function. Each line segment of this curve (local front) can be adequately characterized using a small number of parameters (location of segment's end points, orientation angle and propagation speed). In this work we model the spatiotemporal evolution of a front's line segment by a modified 2D Gaussian function. This approach allows us to treat the estimation of local front parameters as a model updating problem which can be solved analytically [8]. We show here how this updating can be implemented by an in-network processing scheme which forms dynamically small clusters (triplets) of cooperating sensor nodes. Processing at each sensor node is "light" and fast since it amounts to computing parameter updates based on closed form algebraic expressions. We show through extensive simulations, that the proposed scheme can estimate accurately the time varying parameters (angle and speed) of the local front even if the WSN is not very dense and/or a large percentage of its sensor nodes have failed during the passage of the hazard. To the best of our knowledge this is the first attempt to use a fully decentralized WSN to implement in-network predictive modeling of the spatiotemporal evolution of a hazard's local front line. Our work shows that it is possible to make accurate local predictions even by using *low density* WSNs, and forms the basis for developing WSN-supported Dynamic Data Driven Application Systems [1,2] for large-scale environmental monitoring and hazard response management.

This work is partially supported by an "Hrakleitos II" grant co-financed by national (Greek General Secretariat for Research and Technology) and EU funds. The authors acknowledge the support of the funding agency.

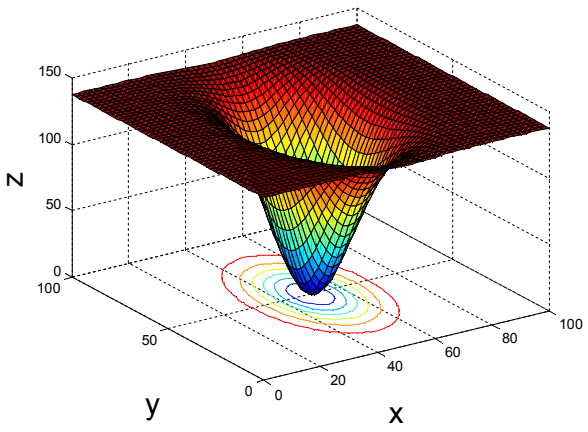

Figure 1. A modified 2D Gaussian and its contour plot.

The rest of the paper is organized as follows: In section II we present the justification for using modified 2D Gaussian functions to model the spatiotemporal evolution of a diffusing phenomenon and present an in-network processing algorithm for estimating their parameters. Experimental validation results are presented and discussed in section III. Finally our findings are summarized and work in progress is outlined in section IV.

II. Modeling and Algorithm Design

A. Elliptical spatiotemporal evolution model for hazards

Many researchers support the notion that the spatiotemporal evolution of diffusing phenomena initiated from a single point source can be approximated by an evolving ellipsoid with a principal axes ratio depending on area prevailing conditions [9-11] (e.g. wind direction and wind speed for wildfires, water stream speed and direction for oil slicks etc.).

For simulating elliptical spatiotemporal evolution we are introducing the use of a modified 2D Gaussian function whose shape and orientation can be controlled by assigning appropriate values to the covariance matrix elements. Figure 1 shows the spatiotemporal evolution of a hazard initiated from single point as this is modeled by the proposed function:

$$f_\theta(x) = \frac{A}{2\pi\,|\Sigma|^{\frac{1}{2}}} - \frac{A}{2\pi\,|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{(x-\mu)^{\mathrm{T}}\Sigma^{-1}(x-\mu)}{2}\right) \qquad (1)$$

where the parameters $\theta = \{A, \mu, \Sigma\}$ are: a predetermined amplitude related constant A, the mean value μ and the 2x2 covariance matrix $\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 \end{bmatrix}$. In this modeling, $f_\theta(x)$ corresponds to time and the larger the time values (vertical z-axis) the larger the area that is covered by the corresponding ellipsoidal contours on the x-y plain (see Figure 1). The factor $A / 2\pi|\Sigma|^{\frac{1}{2}}$ corresponds to the largest value that $f_\theta(x)$ can take (modeled time span). The modified 2D Gaussian function (1) has all the properties needed to models spatiotemporal evolution, while also keeping intact all the advantages that 2D Gaussians offer i.e. adjustable shape, simple parameterization and ability to obtain analytical results.

B. Sensor network assumptions

Before proceeding with the presentation of the collaborative algorithm let us state the assumptions regarding to WSNs adopted in this work:

Deployed sensors are assumed to be stationary with their positions known. A sensor *S*, is able to communicate directly only with its neighbors i.e. the sensor nodes located within a circular region of radius *R* from its location. In a valid deployment each sensor node is assumed to have at least 2 neighbors. The clocks of the sensor nodes *do not* need to be synchronized. Finally we assume that a sensor node *may fail* at any given time due to the hazard's propagation. Once a node fails it stops participating in the collaborative algorithm. Failures are considered permanent.

C. Key elements of the proposed collaborative approach

The main idea of the in-network processing algorithm is as follows: During the evolution of a diffusive phenomenon the deployed sensor nodes are dynamically organized into ad-hoc local clusters of three nodes each (triplet). The sensor nodes of a formed cluster collaborate to update the local front evolution belief model of the Master i.e. the node who initiated the formation of the cluster. The updated model is then propagated forward to new nodes in the same direction as the phenomenon that evolves.

How the model of the Master (prior model) is updated depends on the solution of a KL-divergence [12] minimization problem. The KL-divergence is commonly used to assess dissimilarity between two probability density functions. In essence the proposed model updating procedure corresponds to finding among the modified 2D Gaussian models the one that has minimum KL-divergence (minimum dissimilarity) from the prior model and explains best the most recent sensor field measurements. Minimizing the KL-divergence to the prior model is justified since it is expected (for diffusing hazardous phenomena) that for short time periods the local model parameters change smoothly. We have formulated and solved this optimization problem analytically in [8]. The resulting algebraic expressions can be used to find the updated model parameters without the need for using floating point arithmetic, a fact that respects the energy constraints of WSNs. We provide below the details of the distributed in-network processing steps for model parameter estimation.

D. The collaborative in-network processing algorithm

Each sensor keeps locally the following information about itself: {ID, Location, Detection Status Flag, Sensor Status Flag, Prior Model Parameters}. The Prior Model is represented by the covariance matrix Σ and the direction parameter δ which indicates the evolution direction of the local front (see text below for details). In addition, each sensor keeps locally the following information about each one of its

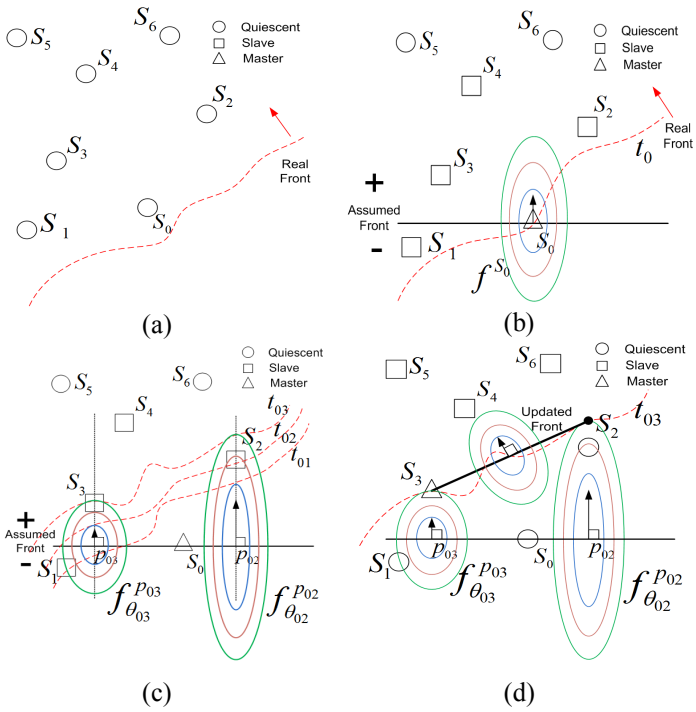

Figure 2. Master's S_0 local front model updating procedure.

neighbors: {ID, Location, Time of (hazard) Detection, Detection Status Flag, Sensor Status Flag}.

Sensor states, self-organization into clusters*:* A sensor may assume one of the following three states:

Quiescent: Default state. The Sensor Status Flag (SSF) has value 0.

Master: It is responsible for the estimation and the updating of the local front parameters. The SSF of a Master has value 1.

Slave: It is responsible for monitoring the phenomenon upon request from a Master. The SSF value for Slave sensors is 2. Upon detection of the phenomenon, a slave's Detection Status Flag (DSF) becomes 1 (default value is 0) and is broadcasted to its neighbors. A slave may serve more than one Masters at any given time.

To facilitate the presentation of the proposed distributed algorithm we will use a simple running example. In Figure 2(a) we assume *w.l.o.g.* that the real local front is the red dashed curve and its local speed and direction are indicated by the length and direction of the corresponding red vector. In this example we assume (*w.l.o.g.*) that a sensor is detecting the phenomenon when it is reached by the evolving front.

When the front line reaches sensor S_0 say at time t_0 it detects the phenomenon, sets its DSF = 1, initializes an internal timer and checks its SSF flag. If SSF = 0 (state = Quiescent) and at least two of its neighbors (sensors inside its communication region, i.e. S_1, S_2, S_3 and S_4 in Figure 2(a)) have not detected the phenomenon yet (DSF = 0) S_0 starts the following procedure to check if it satisfies the necessary conditions to become a Master and forms a cluster.

Forming a local cluster: Sensor node S_0 uses its current (prior) spatiotemporal evolution model (a modified 2D Gaussian) and its location's coordinates (x_0, y_0) and derives the equation of a line that is perpendicular to its prior model's major axis and passes from its location (see figure 2(b)). This line separates the plane in two half planes (positive and negative). After the line derivation, S_0 checks the value of the direction evolution parameter δ_0. The default value of this parameter is 0 and indicates that the front's evolution direction is unknown.

- if $\delta_0 = 0$ (example's case): S_0 checks if it has at least two neighbors that have not detected the phenomenon and belong to the same half plane. If it does, it decides to become a Master. It changes its SSF to 1 and broadcasts a "Master Declaration Message (MDM)" {ID = S_0, SSF = 1} which sets the SSF of its neighbors to 2 (see Figure 2(b)).
- if $\delta_0 = 1$ (-1): S_0 checks if it has at least two neighbors within the *positive (negative)* half plane. If it does, it decides to become a Master. It changes its SSF to 1 and broadcasts a "Master Declaration Message (MDM)" {ID = S_0, SSF = 1} which sets the SSF of its neighbors to 2.

If S_0 does not satisfy any of the aforementioned conditions it decides not to become a Master. It broadcasts a hazardous event "Detection Message (DM)" {with ID = S_0} and leaves its SSF unchanged. S_0's neighbors when they receive the DM they update the information related to S_0 in their neighborhood local tables.

In the presented example S_0 satisfies the conditions to become a Master and broadcasts an MDM. Its neighbors (S_1, S_2, S_3 and S_4 in Figure 2(b)) when they receive this message change their SSFs from 0 to 2 and become S_0's slaves.

Now S_0 waits until it receives two hazardous event Detection Messages (DM) from two neighbors. Let's assume that at Master's (S_0) internal clock times t_{01} and t_{02} ($t_{02} > t_{01}$ *w.l.o.g.*), S_0 receives the DMs from S_1 and S_2 respectively. Then Master S_0 checks if these sensors belong within the same half plane as determined by the aforementioned line.

- If they do, the Master is ready to start the local front parameters updating procedure (to be described below).
- If they do not (as in the example), the Master waits to receive one more DM from one of its neighbors.

By the time (t_{03}) let's say that Master S_0 receives a DM from S_3 which enables him to start its prior model updating procedure.

The Master uses its current (prior) spatiotemporal evolution model (a modified 2D Gaussian) and assumes that the local front at its neighborhood can be approximated by a line segment that is perpendicular to the major axis of its available prior model (see Figure 2(b)) and its length is equal with S_0's communication diameter (2*R*). The speed of motion and the direction, which are determined from the prior model, are denoted by the vector that is perpendicular to the assumed front. Each point on this front evolves as indicated by the prior elliptical evolution model of the Master. This assumption is well justified and has been extensively used by many researchers [9-11] .

Model updating procedure: *Finding the intersection points.* Master S_0 uses the location of slaves S_2 and S_3 (stored in its neighborhood table) and calculates the two points (p_{02}, p_{03}) where these two sensor locations project on the assumed local front's line segment (see Figure 2(c)). It is assumed that the phenomenon is best captured by the Master's prior modified 2D Gaussian model at each point on the assumed front segment, and therefore also for projection points p_{02} , p_{03} as well. The Master S_0 computes the distances between sensor S_2 , S_3 locations and the corresponding projection points (p_{02}, p_{03}) , to be called d_2 and d_3 respectively. Subsequently, Master S_0 estimates two new 2D modified Gaussians, namely $f_{\theta_{02}}^{p_{02}}$ and $f_{\theta_{03}}^{p_{03}}$ (see Figure 2(c)) which: (i) are centered at the projection points (p_{02}, p_{03}) respectively, (ii) have minimum Kullback-Leibler divergence from the Master's prior model, and (iii) assign at the corresponding slave locations, S_2 and S_3, time values equal to the measured by the Master time differences t_{02} and t_{03} respectively. The direction of the major axes of the two new 2D modified Gaussians are the same to that of the prior model of the Master but their rate of spread may be different, as indicated by the size of the two small arrows originating at the two projection points (p_{02}, p_{03}) in Figure 2(c). In [8] we present the formulation and solution of the KL optimization problem leading to closed form expressions for updating the model parameters.

Estimating the new local front's direction: Master S_0 checks its table to find out which one of the two used Slaves (S_2, S_3) has detected the phenomenon most recently (it is S_3 in our example *w.l.o.g.*). The location of this sensor is for sure one point from where the local front line passes at time t_{03} (see green contour in Figure 2(c)). A second point of the local front is estimated as follows: Master S_0 calculates, using the updated modified Gaussian model $f_{\theta_{02}}^{p_{02}}$, the point on that model's major axis reached after t_{03} (marked by a black dot in Figure 2(d)). This point and the location point of S_3 are equi-temporal (both are reached by the front at time t_{03}) and thus define an estimated local front line segment at the specific time instant.

***Model selection*:** One of the two models, either $f_{\theta_{02}}^{p_{02}}$ or $f_{\theta_{03}}^{p_{03}}$, will be used as the new spatiotemporal evolution model for the newly formed local front line segment (hard decision). Assuming smooth model changes, the algorithm selects among them the one with the smaller KL-divergence from the prior model of the Master f^{S_0} , rotated so that its major axis is perpendicular to the new local front's line segment (in our example the selected model is $f_{\theta_{03}}^{p_{03}}$). The new rate of spread and direction (same as major axis) of this new front are determined from the updated model parameters.

To update the evolution direction parameter δ_0 , Master S_0 checks in which half plane (positive or negative) belong the slaves which "helped" him to update the model (S_2 and S_3 in our example) and assigns the appropriate value (1, -1) to δ_0 .

Model propagation: After updating the prior model, Master S_0 sends the new model information to the helper-slave which detected the phenomenon most recently (it was S_3 in our example), and asks it to become the new Master. If S_3 satisfies the aforementioned necessary conditions (see paragraph *Forming a local cluster*) it accepts the responsibility and returns a confirmation message to S_0 . At this point, S_0 broadcasts a "release slaves" message to all its neighbors and if the recipients are not "enslaved" by some other Master(s) they change in turn their SSF to 0, so that they may become Masters themselves in the future. If S_3 does not satisfy the necessary conditions to become the new Master, it rejects the current Master's offer forcing S_0 to try the same exact negotiation with the second helper (node S_2 in our example). If S_2 also fails to become the new Master, S_0 gives up with the helpers and asks all its slaves to propagate the updated model information to their neighbors, releases them, and the algorithm repeats from the beginning.

III. Evaluation Method and Results

We present in this section simulation results that demonstrate the ability of the proposed distributed WSN algorithm to estimate accurately the direction and speed of a propagating hazard's local line front. Specifically, we have conducted the following experiments:

Experiment 1: This experiment was designed to demonstrate how the algorithm performs under different sensor node densities and node failure probabilities (equal to 0, 0.1, 0.2, 0.3 respectively). As node failure we consider the inability of a sensor node to participate in the distributed algorithm. In this

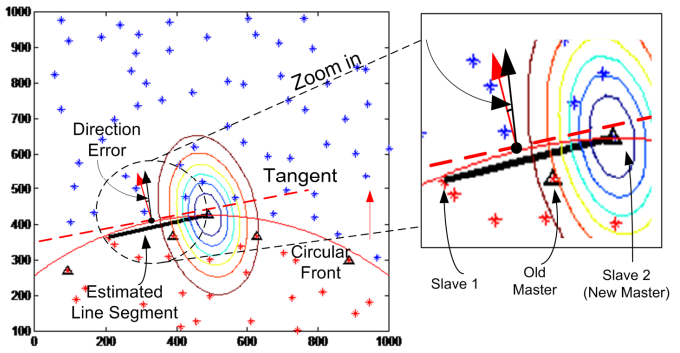

Figure 3. Matlab simulator snapshot. Red (blue) crosses denote sensors which have (have not) detected the evolving front (red circle). The line segment is updated through the cooperation of the three sensors indicated by the arrows in the right side panel. The ellipsoid is the contour plot of the updated 2D Gaussian Model which describes the spatiotemporal evolution behavior of the specific line segment (local front).

experiment the diffusing phenomenon front line was modeled as a circle of fixed center but with a radius that is increasing at a predetermined constant rate (equal to 1.5 meters/min).

Experiment 2: This experiment was designed to demonstrate the ability of the proposed algorithm to estimate the evolution parameters (speed and direction) of a time varying front, under different sensor node densities and failure probabilities. The front line of the diffusing phenomenon is again modeled as a circle with fixed center and increasing radius, but of time varying rate. More specifically, the speed at which the radius of the circle is increasing is initially 0.5 meters/min and then starts becoming larger with a constant rate until it reaches its maximum value (3.5meters/min) in the middle of the deployment area. Then the speed value starts to decrease at the same rate until it comes back to its original value. In that manner, when the front reaches the edge of the deployment area its speed will be again equal to its initial value (0.5 meters/min).

To simulate the behavior of our algorithm we have developed a fully parameterized Matlab based WSN simulator which allows us to simulate scenarios with different: sensor node densities, deployments strategies, sensor node failure probabilities and front evolution characteristics (different shapes and speeds).

The network densities used in both experiments were $5x10^{-5}$, $7.5x10^{-5}$ and 10^{-4} sensor nodes per square meter which correspond to 50, 75 and 100 pseudo-randomly deployed sensor nodes within a square area of $1km^2$ respectively. For each density value we run 30 simulations differing only in terms of the sensor nodes deployment. The same 30 sensor node deployments per density were used in both experiments. Furthermore, in order to guarantee that we have a fully connected network for each density scenario, the sensor communication radius was set to R=150m and was assumed to be the same for all sensor nodes.

In order to evaluate the accuracy of the proposed algorithm, we compared the local front estimates (in terms of direction and speed) to the known ground truth. To estimate

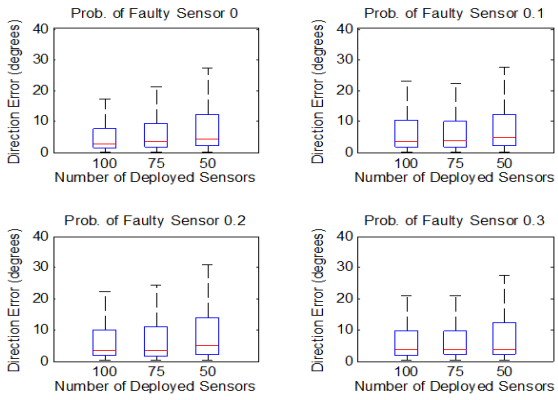

Figure 4. Experiment 1 results. Distribution of the direction angle error under different scenarios. Each panel corresponds to a different probability of faulty sensors.

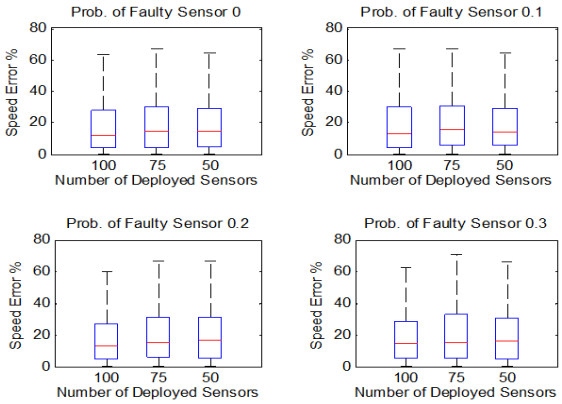

Figure 5. Experiment 1 results. Distribution of the speed error under different scenarios. Each panel corresponds to a different probability of faulty sensors.

the direction error we calculated the angle of two vectors: the vector that is perpendicular to the estimated front line segment and the vector perpendicular to the tangent of the circle (modeling the hazard) at the middle point of the corresponding arc (see Figure 3). For a model the speed is estimated as the ratio $\Delta s / \Delta t$, where $\Delta s = 2\sigma'_{11i}$ and σ'^{2}_{11i} is the variance element of the diagonal covariance matrix which results after rotating Σ_i such as the major axis of the corresponding 2D modified Gaussian is aligned with the horizontal x-axis. If we call the rotated model f'_θ then $\Delta t = f'_\theta(\Delta s)$.

Figures 4 and 5 present the boxplots summarizing the distribution of the angle and speed estimation errors respectively for experiment 1. For the generation of each boxplot we considered as sample points all front line segment updates for the 30 runs of the corresponding scenario (i.e. with a specific number of sensor nodes deployed and probability of node failures). As observed from the figures the distribution of errors seems to be insensitive to the density and failure probability variations. This was also confirmed by comparing pairwise the means of the boxploted densities using Student's t-test. For all cases the differences of the means were deemed insignificant at the 0.05 significance level.

Table 1 provides for each simulation scenario (30 runs) the total number of updates and the mean number of messages per

TABLE I. EXPERIMENT 1: TOTAL NUMBER OF UPDATES AND MEAN NUMBER OF MESSAGES PER MODEL UPDATE FOR EACH SIMULATION SCENARIO CONSIDERED (30 RUNS).

Probability of Faulty Sensor	Number of Deployed Sensor Node					
	50		75		100	
	Total Updates	Mean number of Messages Per Model Update	Total Updates	Mean number of Messages Per Model Update	Total Updates	Mean number of Messages Per Model Update
0	302	10.09	686	7.15	1046	6.21
0.1	225	11.75	537	8.27	812	7.03
0.2	190	12.93	420	9.22	645	7.86
0.3	144	15.73	326	10.41	512	8.98

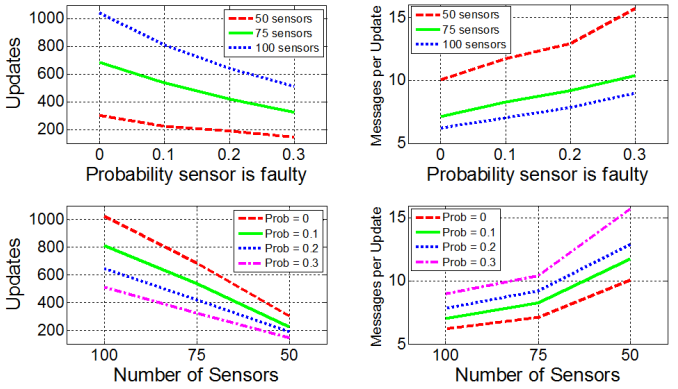

Figure 6. Experiment 1: Total number of updates and mean number of messages per model update as a function of the faulty sensor probability (first row of panels) and as a function of the deployed sensor nodes (second row)

model update required by the proposed algorithm. Figure 6 presents a graphical representation of the information in Table 1 which helps us visualize the trends of the total number of updates and the mean number of messages per model update as the number of the deployed sensor nodes decreases and the faulty sensor probability increases.

As observed from Figure 6, for each density scenario as the number of faulty node probability increases the total number of model updates decreases (top left panel). This behavior could be easily explained if we consider that: a) increasing the faulty sensor probability implies a reduction of the functional sensor nodes which participate in the in-network algorithm in the same area and therefore implies a reduction of the network's density, b) faulty sensors may lead to the formation of "dummy" clusters, i.e. clusters where sensor node malfunctions occurring within render the Master unable to update its model parameters. Another important observation is that as the network's density decreases so does the number of model updates for all simulated probability of faulty sensor scenarios (bottom left panel). This can be justified since a smaller number of sensor nodes in a given area implies fewer neighbors within a sensor's communication range. The reduced number of neighbors in turn implies that it becomes more difficult for a Master to find at least one of its slaves (which participate to its model updating procedure) to satisfy the necessary conditions for becoming a Master (see section II *Forming a local cluster*). When a Master cannot find a new qualified Master, it is forced to broadcast a message to its slaves so that they propagate its updated model to their neighbors (see section II *Model propagation*). This model propagation procedure increases the mean number of

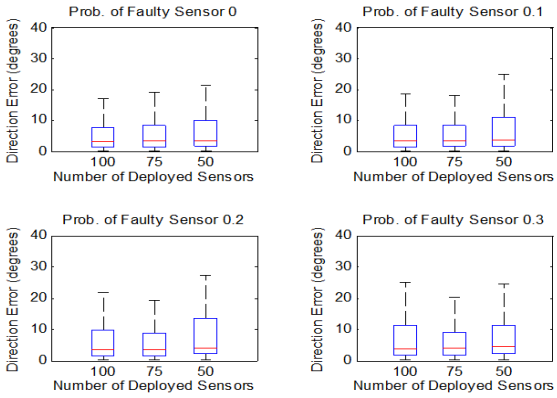

Figure 7. Experiment 2 results. Distribution of the direction angle error under different scenarios. Each panel corresponds to a different probability of faulty sensors.

Figure 8. Experiment 2 results. Distribution of the speed error under different scenarios. Each panel corresponds to a different probability of faulty sensors.

messages per model update (bottom right panel). Finally, the number of messages exchanged (wasted) within the formed "dummy" clusters explains why the mean number of messages per update becomes larger as the probability a sensor is faulty increases (top right panel).

Figures 7 and 8 summarize the distribution of the estimation error for the direction angle and the speed for each simulation scenario (30 runs) for experiment 2. As we observe from the provided boxplots direction angle and the speed errors do not seem to change significantly between scenarios. This observation was confirmed also by applying pairwise Student t-tests for the means at the 0.05 level of significance.

Finally Table 2 and Figure 9 (in correspondence to Table 1 and Figure 6) provide for experiment 2 the data and the graphical representation respectively of the total number of updates and the mean number of messages per model update. The interpretation of these results is similar to that of experiment 1 results.

In summary, the presented results suggest that the proposed WSN collaborative algorithm manages to provide good quality local estimates of the evolving hazard's front parameters (direction and speed) under a variety of different simulation scenarios. Furthermore its accuracy is insensitive to changes in sensor density and sensor failure probability making it a

TABLE II. EXPERIMENT 2: TOTAL NUMBER OF UPDATES AND MEAN NUMBER OF MESSAGES PER MODEL UPDATE FOR EACH SIMULATION SCENARIO CONSIDERED (30 RUNS).

Probability of Faulty Sensor	Number of Deployed Sensor Node					
	50		75		100	
	Total Updates	Mean number of Messages Per Model Update	Total Updates	Mean number of Messages Per Model Update	Total Updates	Mean number of Messages Per Model Update
0	301	10.07	672	7.22	1042	6.22
0.1	231	11.44	521	8.04	812	7.09
0.2	181	13.49	409	9.45	669	7.60
0.3	139	16	314	10.81	510	9.30

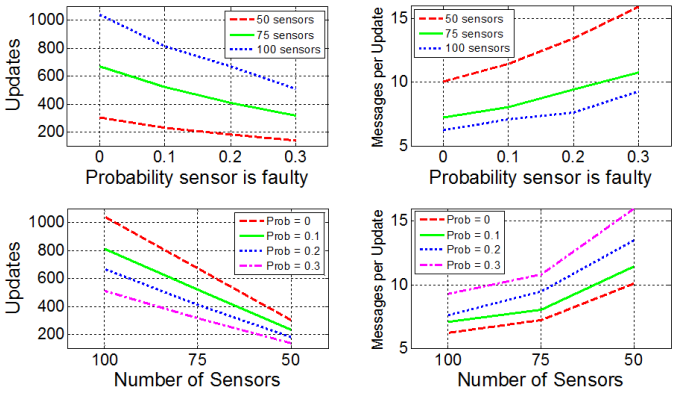

Figure 9. Experiment 2: Total number of updates and mean number of messages per model update as a function of the faulty sensor probability (first row of panels) and as a function of the deployed sensor nodes (second row).

realistic approach for large scale environmental monitoring applications.

As we have mentioned in the introduction there are recently several interesting in-network processing schemes proposed for estimating and tracking the boundaries of an evolving "object" [5-7]. Apart from using orders of magnitude more sensors per square kilometer, these schemes do not intend to estimate the local front parameters, and their target is not predictive modeling (of local evolution's direction and speed). Therefore, since the objectives of the two lines of work are quite different we have not attempted a performance comparison to such methods, which would be inherently difficult to perform anyway due to the different setup and WSN related assumptions.

We should mention however, that in contrast to related works [5-7] our approach works with reasonable sensor node densities, allows for sensor node failures (that are certainly expected in extreme environmental conditions) and does not require the clocks of sensor nodes to be synchronized. It is known that clock synchronization is difficult to achieve even in medium scale WSNs and under normal conditions.

IV. CONCLUSIONS

We have presented a novel collaborative WSN algorithm that can estimate effectively the parameters (direction and speed) of the evolving front of a hazardous diffusing phenomenon. We characterize the spatiotemporal evolution of a front's line segment by a modified 2D Gaussian function which serves as an adaptive local predictive model. The proposed WSN collaborative algorithm estimates the time varying direction and speed of the phenomenon by dynamically forming small-size clusters of sensor nodes (triplets). Furthermore it updates the local model parameters and propagates them in the direction of the front's movement in a fully decentralized manner. The experimental results show that the estimation accuracy of the proposed algorithm is insensitive to changes in sensor density and sensor failure probability, making it a very realistic approach for real-world scenarios. Moreover, the results indicate that the proposed scheme estimates equally well and with reasonable accuracy the parameters of fronts which evolve with constant or time-varying speed.

We are currently designing a new version of the algorithm in which the speed parameter is updated using a soft decision scheme. We believe that with this modification we will be able to reduce the speed estimation error without increasing the computational load. Furthermore, we are porting the algorithm to a real sensor network platform for in-field testing and validation. Finally, we are working on an algorithm for combining the derived local estimates to reconstruct an overall (global) front line estimate of the hazard (e.g. front line of an extended wildfire). The capability to update dynamically the parameters of local front models in conjunction with this new global front reconstruction algorithm will allow us to make predictions for the overall front line's movement which of course will become more accurate dynamically in areas where more sensors are available.

REFERENCES

[1] A. Ononye, A. Vodacek, Saber, "Automated extraction of fire line parameters from multispectral infrared images". Remote Sens. Environ. 108:179-188,2007.

[2] J.Mandel, J. Beezley, J. Coen, M. Kim, "Data Assimilation for Wildland Fires: Ensemble Kalman filters in coupled atmosphere-surface models", IEEE Control Systems Megazine vol.29, issu 3, pp. 47-65, 2009.

[3] E.S. Manolakos, D. V. Manatakis, G, Xanthopoulos, "Temperature Field "Modeling and Simulation of Wireless Sensor Network Behaviour During a Spreading Wildfire", Proc. 16th EUSIPCO, Aug. 2008.

[4] O. Sekkas, D.V. Manatakis, E. S. Manolakos and S. Hadjieuthimiades, "Sensor and Computing Infrastructure for Environmental Risks – The SCIER System" chapter 16, in Advanced ICTs for Disaster Management and Threat Detection: Collaborative and Distributed Frameworks, pp. 262-278 ISBN: 9781615209873 l, 2010.

[5] J.Kim, K. Kim, S.Chauhdary, W. Yang, M. Park, "DEMOCO: Energy-Efficient Detection and Monitoring for Continuous Objects in WSN", IEICE Trans.on Comm, vol.E91-B, pp.3648-3656.

[6] W. Chang, H. Lin, Z. Cheng: "CODA: A Continuous Object Detection and Tracking Algorithm for Wireless Ad Hoc Sensor Networks". Consumer Communications and Networking Conf, pp. 168-174, 2008

[7] C. Zhong, M. Worboys, "Energy Efficient Continuous Boundary Monitoring in Sensor Networks" Technical Report, 2007. Available: http://ilab1.korea.ac.kr/papers/ref2.pdf.

[8] D.V. Manatakis, E.S. Manolakos, "Predictive Modeling of the Spatiotemproal Evolution of an Envirnmental Hazard and its Sensor Network Implementation." In Proc. ICASSP 2011, in press.

[9] H. Anderson,"Predicting wind-driven wildland fire size and shape". USDA Forest. Service. Research. Paper. INT-305, Feb.1983.

[10] M. Alexander, "Estimating the length-to-breadth ratio of elliptical forest fire patterns". Proc. 8th Conf. on Fire and Forest Meteorology, pp. 287-304. 1985

[11] M. Marghany,"RADARSAT for oil spill trajectory model", Journal of Env. Modeling and Software, Vol.19, May 2004, pp. 473-483.

[12] S. Kullback, Information Theory and Statistics, Dover Publ. 1968