

Tracking Dynamic Boundary Fronts Using Range Sensors

Subhasri Duttagupta, Krithi Ramamritham,
Purushottam Kulkarni¹, and Kannan M. Moudgalya²

¹ Dept. of Computer Science and Engineering
Indian Institute of Technology, Bombay
{subhasri,krithi,puru}@cse.iitb.ac.in

² Dept. of Chemical Engineering
kannan@che.iitb.ac.in

Abstract. We examine the problem of tracking dynamic boundaries occurring in natural phenomena using range sensors. Two main challenges of the boundary tracking problem are energy-efficient boundary estimations from noisy observations and continuous tracking of the boundary. We propose a novel approach which uses a regression-based *spatial estimation* technique to determine discrete points on the boundary and estimates a confidence band around the entire boundary. In addition, a Kalman Filter-based *temporal estimation* technique is used to selectively refresh the estimated boundary to meet the accuracy requirements. Our algorithm for dynamic boundary tracking (DBTR) combines *temporal* estimation with an aperiodically updated *spatial* estimation and provides a low overhead solution to track boundaries without requiring *prior* knowledge about the dynamics of the boundary. Experimental results demonstrate the effectiveness of our algorithm and estimated confidence bands achieve *loss of coverage* of less than 2 – 5% for a variety of boundaries with different spatial characteristics.

1 Introduction

Large scale sensor networks are being deployed for real-time monitoring applications, such as detecting leakage of hazardous material, tracking forest fires and environmental phenomena. Consider a poisonous gas or plume monitoring application [1], tracking a spreading plume requires continuous updates regarding extent of the plume, its direction and its distance from habitats. The plume can be considered to be delineated by a boundary such that tracking the movement of the plume involves estimating a dynamically changing boundary. Strategically deployed range sensors can coordinate to track boundaries associated with such natural phenomena.

The solution space for boundary estimation using a sensor network can be examined along four orthogonal dimensions: (i) the *characteristic* of sensors - static [2], [3] or mobile [4]; (ii) *sensing capabilities* - in-situ sensing or range/remote sensing; (iii) the *accuracy* of estimation; and (iv) the *nature* of the boundary - static or

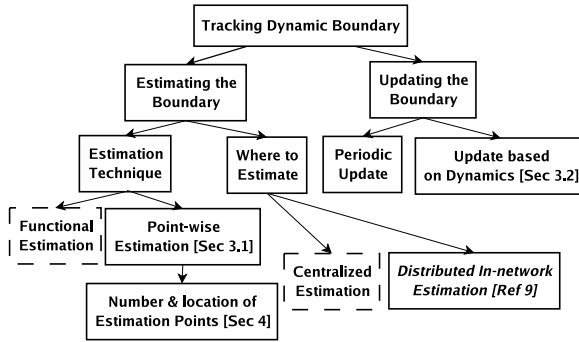


Fig. 1. Issues related to tracking dynamic boundaries. (Solid boxes indicate problems addressed in this paper).

dynamic. In this paper, we address the problem of estimating dynamic boundaries using static sensors with range/remote sensing capabilities. Previous techniques to estimate boundaries have employed *in-situ* [2] static or mobile sensors. In applications like tracking a plume, or predicting trajectory of weather parameters [5], in-situ sensing is not feasible due to difficulty in remote access or requirement of a large-scale deployment of sensors. In such situations, techniques based on range or remote sensing using radar or laser pulses are better suited. The basic difference between in-situ and range sensing is that, in the former approach a sensor measures the value of the field at its current location whereas in the latter approach a sensor finds approximate distance to a remote location where the field value equals some specific threshold. Radars used in [5] scan an angular area by swiping upto 360 degrees and gather reflectivity and wind velocity information. Lidars (Light Detection and Ranging) are being used for detecting forest fires [6], [7] in the last few years. Lidars detect fire by analysing the energy back-scattered from smoke particles resulting from fire and measure the distance between lidar sensor and a point on the target(smoke) using simple principle of light. While today lidars are not capable of wireless communication, we envision in near future low power, inexpensive sensors with radar/lidar distance sensing and wireless/optical communication capability will be available. In the rest of the paper, we assume such sensors are used to detect boundaries occurring in natural phenomena.

Figure 1 is a pictorial representation of the issues involved in tracking a dynamic boundary. The two main issues are estimating the boundary and updating the estimates as the boundary moves. There are two broad techniques to estimate boundaries, (i) *Functional estimation* and (ii) *Point-wise estimation*. Unlike a functional estimation technique, a point-wise estimation technique assumes that boundaries consist of discrete points and individual points are estimated without reference to any specific functional form. The effectiveness of this technique depends on the number and locations of boundary estimation points. Our proposed point-wise technique exploits spatial variations to determine locations of estimation points and minimizes the number of estimation points.

Error in range estimation due to inherent inaccuracy of sensors introduces error in estimating points on the boundary. The challenge is to estimate an accurate boundary in the presence of noisy observations. In this paper, we use a *kernel smoothing* technique that exploits spatial correlation between proximate sensors so as to reduce the effect of range sensing errors. Further, a centralized technique for estimating boundaries suffers from high communication overheads. We explore a decentralized solution that utilizes local computation capability and performs *in-network* aggregation at sensors within the network to significantly reduce the communication overhead for boundary estimation.

In order to track a dynamic boundary, the boundary estimates need to be updated periodically. The ability to use the temporal characteristics of the boundary to update its estimate only when required is another challenge we address in this paper. The instances when a boundary estimate is updated depend on the dynamics of the boundary. But, unless there is clairvoyance, optimal choice of periodicity at each point is not possible in real-time tracking scenarios. Our approach uses a Kalman Filter based mechanism to *predict the movement* of the boundary and updates estimates only when error in the current estimate exceeds a pre-defined threshold.

We address the problem of accurate dynamic boundary estimation with observations from range sensors incurring low communication overhead. By way of *contributions*,

- We propose DBTR, a novel technique that intelligently combines both spatial and temporal estimation techniques for accurate dynamic boundary estimation. Our spatial estimation scheme is designed such that it lends itself to in-network aggregation.
- We demonstrate the effectiveness of DBTR for tracking a dynamic boundary without *prior* knowledge about the dynamics of the boundary. The performance of DBTR in terms of communication overheads and accuracy is comparable with the best *optimal* periodic update scheme.
- We experimentally show that the estimated confidence band around a boundary has *loss of coverage* (defined in Section 2) less than 2–5% for a spectrum of boundaries with different spatial characteristics.

2 System Model and Problem Formulation

We assume n sensor nodes distributed randomly over a two dimensional field measuring a phenomenon (e.g., viscosity or reflectivity). Further, each sensor has directional range sensing capability to estimate the closest point whose field value matches the definition of a boundary. An observation (x_i, y_i) of the i^{th} sensor represents the location of a boundary point. We assume that sensors can align their sensing antennas at any angle to locate a point on the boundary. Further, all sensors are located on one side of the boundary tracking the *front* of a phenomenon. Figure 2(a) shows a typical scenario of sensors detecting various points on the boundary. A sensor at location (x_s, y_s) positions its beam at an

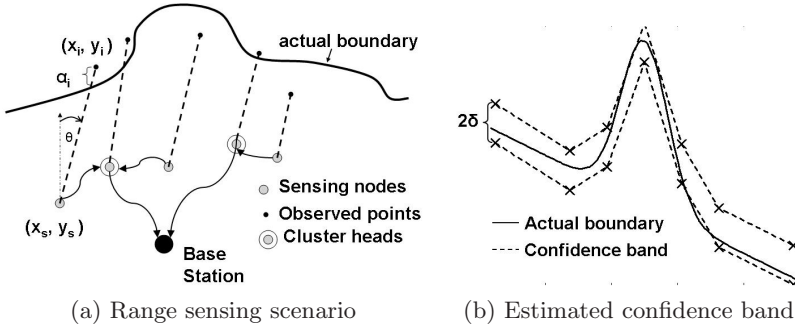


Fig. 2. Tracking dynamic boundary using range sensing observations

angle θ ¹ w.r.t. the y axis and detects a point (x_i, y_i) on the boundary with error α_i along the sensing direction.

Given n observations $\{(x_i, y_i)\}_{i=1}^n$ with errors, we address the problem of estimating a confidence band of a specific width δ (the distance between the estimated boundary and limit of the band) as shown in Figure 2(b). The confidence band should cover the dynamic boundary at all times and with high probability. We measure the accuracy of coverage in terms of *loss of coverage (LOC)*, the probability of the band not covering the actual boundary. If $(x_i, d(x_i))$ is a point on the actual boundary, *LOC* over a set of n sensors is defined as:

$$LOC(\delta) = \frac{1}{n} \sum_{i=1}^n I(|\hat{d}(x_i) - d(x_i)| > \delta) \quad (1)$$

where $I(a)$ is an indicator function, i.e., $I(a) = 1$ if a is true, $I(a) = 0$ otherwise and $d(x_i)$ is the actual distance from estimation point x_i and $\hat{d}(x)$ is its estimate. Minimizing *LOC* helps maximize accuracy of coverage.

Our model assumes that sensor nodes are equipped with wireless radios. Further, these nodes use clustering for aggregation and multi-hop routing techniques for communication with a base station. Finally, since sensors are energy-limited, we aim to minimize the communication overhead at nodes to increase lifetime of sensor networks.

3 DBTR: Dynamic Boundary Tracking Algorithm

A dynamic boundary has mainly two types of variations: spatial and temporal. Effective tracking of dynamic boundaries requires handling both of these variations. In this section, we describe DBTR, a point-wise algorithm for dynamic boundary tracking which combines a *spatial estimation* technique and a *temporal estimation* technique to effectively track a dynamic boundary.

¹ In this paper, we assume the antennas are aligned to the y axis i.e., $\theta = 0$. Please refer to [8] for a discussion on using non-zero sensing angles.

The first step of DBTR is to estimate the boundary at a location x_{pj} using a spatial estimation technique. It uses spatial correlations among observations at a given time by sensors within a small *neighborhood* of x_{pj} . Cluster heads perform aggregation operations on sensor observations to estimate a number of boundary points. Partial information of the boundary from cluster heads is then sent to the base station where the final estimates are computed. A confidence band is estimated from multiple boundary points around the entire boundary using an interpolation scheme.

The second component of DBTR is a temporal estimation technique which ensures that the estimates are updated whenever due to changes in the boundary the confidence band does not cover the boundary with a desired accuracy. DBTR uses a Kalman Filter based technique to predict future boundary locations based on its model of the boundary dynamics. Once the boundary has moved by more than a certain threshold, DBTR invokes the spatial estimation technique to get an accurate estimate of the boundary. As a result, boundary estimates are updated based on only the local dynamics of the boundary and partial estimates track changes in sections of the boundary. Both of these lead to reduction in communication overhead for accurate boundary estimation.

3.1 Regression-Based Spatial Estimation Technique

This section briefly sketches the non-parametric regression method used by the spatial estimation technique as discussed in [9].

For each sensor observation (x_i, y_i) , the independent variable x_i and the dependent variable y_i can be modeled as a non-parametric regression relation. For n observations at the n sensors, the regression relation is stated as,

$$y_i = d(x_i) + \alpha_i, \quad i = 1, \dots, n \quad (2)$$

where d is the regression relation between x_i and y_i , and α_i the observation error. If the error distribution has mean zero, then the expected value of the distance to the boundary at x_i is $d(x_i)$. We assume the error distribution to be normal $N(0, \sigma^2)$, where σ^2 is the observation error variance. Note that in reality, observations from range sensors may not satisfy this assumption but in experiments with real sensors [9], we verify that the mathematical technique is applicable even when the assumption does not hold.

For a point on the boundary estimated at location x_{pj} , $d(x_{pj})$ is the actual distance of boundary from x_{pj} and $\hat{d}(x_{pj})$ is the estimated distance. Assuming a smooth boundary, it is possible to use a local average of the observations near x_{pj} to construct an estimate for $d(x_{pj})$. The *kernel smoothing* [10] technique that uses observations in the neighborhood of x_{pj} is applied to estimate $\hat{d}(x_{pj})$. Thus $\hat{d}(x_{pj})$ is,

$$\hat{d}(x_{pj}) = \frac{1}{n} \sum_{i=1}^n W_i(x_{pj}) y_i \quad (3)$$

where $\{W_i(x_{pj})\}_{i=1}^n$ denotes a sequence of weights defined using a *kernel* function. The weight for x_{pj} is non-zero in the neighborhood from $(x_{pj} - h)$ to

$(x_{pj} + h)$, referred to as the h -neighborhood of x_{pj} . Using the observations in the h -neighborhood of x_{pj} , the variance of the y -component of the observations is also estimated. This variance $\hat{\sigma}^2(x_{pj})$ captures the spatial variations of the boundary and is an estimate of the actual observation error variance σ^2 .

$$\hat{\sigma}^2(x_{pj}) = \left(\frac{1}{n} \sum_{i=1}^n W_i(x_{pj}) y_i^2 \right) - \hat{d}^2(x_{pj}) \quad (4)$$

A crucial step in estimating the boundary is choosing the parameter h that controls how much of the neighborhood around x_{pj} has to be considered. An iterative *plug-in* approach (refer [11] for details) is used to estimate the optimal value of h that minimizes the error in estimation. Evaluation of optimal h involves estimation of the boundary for all x values of the sensors. This is the reason we recommend a centralized approach for evaluating the optimal h initially.

Since both expressions, Equation (3) and Equation (4), are summations, they are amenable to distributed evaluation. All observations contributing to the estimation of $\hat{d}(x_{pj})$ and $\hat{\sigma}^2(x_{pj})$ may not be available at a single cluster head. Each cluster head computes partial expressions for $\hat{d}(x_{pj})$ and $\hat{\sigma}^2(x_{pj})$, referred to as *partial aggregates*. Whenever possible, partial aggregates from multiple cluster heads for a specific x_{pj} are combined at intermediate nodes and forwarded to the base station. The base station collects all partial aggregates and estimates $\hat{d}(x_{pj})$ and $\hat{\sigma}^2(x_{pj})$ for all estimation points x_{pj} . Using the above technique, k distinct points x_{pj} , $j = 1, \dots, k$ along a boundary are estimated. These k points on the boundary are used as input to an interpolation scheme that estimates the confidence band at δ distance around the entire boundary. DBTR uses smoothing spline [10] interpolation to estimate the boundary.

3.2 Model-Based Temporal Estimation Technique

The temporal estimation technique uses a model for the dynamics built using a time sequence of observations of the distances to the boundary. Typically the model is dependent on the exact application scenario. But the distinction in our approach is that here the sensors are not performing *in situ* measurements. Specifically, we are interested in modeling the *velocity* of the boundary which may be affected by factors such as the prevailing weather conditions, surrounding topography etc. If the combined effect of these factors can be modeled as a Gaussian error, and the actual physical process has a linear dynamics, then traditional tracking models like Kalman Filter can be used. A sensor maintains a state representation of distances to the boundary that is updated at each time step. Assuming that the boundary at a discrete point changes in a linear fashion with time, we use Kalman Filters to predict future boundary locations.

Process state $s(x_{pj}, t_i)$ consists of the actual distance $d(x_{pj}, t_i)$ to the boundary at x_{pj} and the velocity of the boundary along the y axis at time instant t_i .

$$s(x_{pj}, t_i) = \begin{bmatrix} d(x_{pj}, t_i) \\ \dot{d}(x_{pj}, t_i) \end{bmatrix}$$

$\dot{d}(x_{pj}, t_i)$ denotes the change in $d(x_{pj}, t_i)$ with respect to time. Irrespective of the actual movement of the boundary, we are interested in knowing only the change in $d(x_{pj}, t_i)$, i.e., the y component of the velocity of the boundary at x_{pj} . The principle of remote or range sensing helps in reducing the *dimensionality* of the problem because a range sensor can always find the distance to a moving boundary irrespective of its own location. For simplicity, here we assume that the boundary moves with a *constant mean velocity* having a mean zero random acceleration. Then the state space equation becomes:

$$s(x_{pj}, t_i) = F \times s(x_{pj}, t_{i-1}) + G \times \alpha_p(x_{pj}, t_{i-1}) \quad (5)$$

where $\alpha_p(x_{pj}, t_{i-1})$ is a Gaussian error with distribution $N[0, \sigma_p^2]$. The matrix F relates the state at time t_i to the state at time t_{i-1} . The term $G \times \alpha_p(x_{pj}, t_{i-1})$ represents the noise component in the process model and matrices F and G can be obtained using simple laws of motion:

$$F = \begin{bmatrix} 1 & t_s \\ 0 & 1 \end{bmatrix} \text{ and } G = \begin{bmatrix} \frac{t_s^2}{2} \\ t_s \end{bmatrix}$$

where t_s is the duration between time instant t_i and t_{i-1} . In this case, it can be same as the sampling period of sensors. Assuming that the model accurately represents the dynamics of the boundary, σ_p^2 can be taken as a small quantity as compared to the observation error variance σ^2 . If $\alpha(x_{pj}, t_i)$ is the error in sensor observations as given in Equation (2), the observation $y(t_i)$ at x_{pj} is linearly related to the state using the observation matrix as:

$$y(t_i) = H \times s(x_{pj}, t_i) + \alpha(x_{pj}, t_i) \quad (6)$$

where $H = [1 \ 0]$ is the observation matrix. This relationship is helpful to derive the distance $d(x_{pj}, t_i)$ information from the current state. The observation error covariance for the Kalman Filter at estimation point x_{pj} is obtained from observation error variance $\sigma^2(x_{pj})$ estimated in Equation (4).

While the boundary can be estimated at any k points using the spatial estimation technique irrespective of whether sensors are located at those points, the temporal estimation has to be associated with a specific sensor and its observations. Assuming that the boundary has similar temporal variation within the h -neighborhood of a location, any sensor having observations within h distance from x_{pj} can perform the temporal estimation for x_{pj} . Moreover, by applying distinct Kalman Filter-based estimates for each of the k points, it is possible to track a boundary that has different sections moving at different velocities.

3.3 DBTR – Combining Spatial and Temporal Estimations

The proposed algorithm combines both spatial estimation as well as Kalman Filter-based temporal estimation and is illustrated in Figure 3 for a specific estimation point x_{pj} (x_{pj} is omitted from all terms for clarity). In this block diagram, two stages of Kalman Filter, state *prediction* and state *update*, are

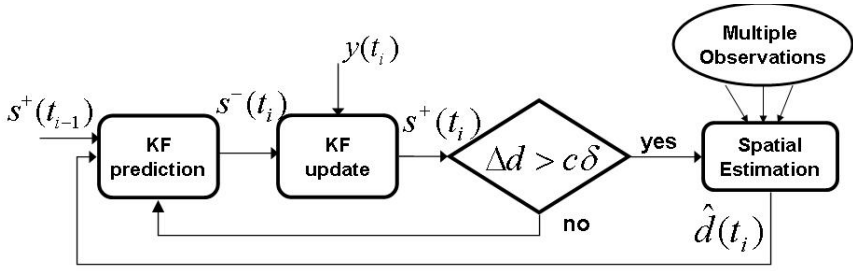


Fig. 3. Details of combining the Spatial and Temporal Estimation Techniques

shown separately. The prediction stage is used to predict the $s^-(t_i)$ from the state $s^+(t_{i-1})$ at previous time instant t_{i-1} . The output from the prediction stage and the new sensor observation are used to obtain the updated output $s^+(t_i)$. Both the prediction and update stages are needed for maintaining the current distance information. From the state information, the distance to the boundary as predicted by the Kalman Filter is obtained using $H \times s^+(t_i)$. This is compared with $\hat{d}(t_{Last})$, the last updated estimate obtained using the regression technique. Then, the difference Δd is estimated as:

$$\Delta d = H \times s^+(t_i) - \hat{d}(t_{Last}) \quad (7)$$

If the difference is more than $c \times \delta$ (where c is a constant and δ is the user specified width of the confidence band), it implies the boundary at x_{pj} has moved a distance larger than $c \times \delta$. Then the boundary is updated with the latest observations from all sensors in the h -neighborhood of x_{pj} . The estimate from the spatial technique is taken as the latest best estimate of the distance to the boundary at x_{pj} and is used by future temporal estimations for more accurate prediction. In Figure 3, $\hat{d}(t_i)$, the output from spatial estimation is used to update the distance information in state $s^+(t_i)$. The intelligent combination of spatial and temporal estimation techniques not only minimizes wasted boundary updates but also avoids updates to sections of the boundary that have not moved significantly.

4 Minimizing Number of Estimation Points

Our technique uses an interpolation scheme over a finite set of boundary points to estimate a confidence band around the boundary. The interpolation error of the confidence band reduces as the number of estimation points is increased. However, this will lead to an increased communication overhead. Our goal is two fold: (i) to estimate boundary at a minimal number of points, and (ii) to ensure that the *interpolated band* as mentioned in Section 2 covers the actual boundary with high confidence. We use k to denote the number of estimation points.

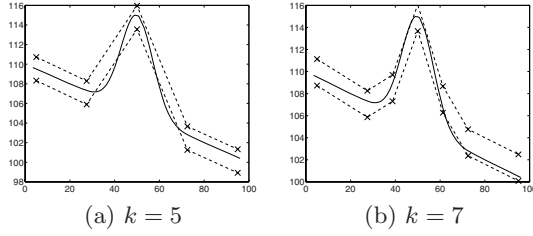


Fig. 4. Confidence band with different number of estimation points

Since the variance $\hat{\sigma}^2(x)$ captures the local spatial changes within the h -neighborhood, a higher value of the variance indicates a larger spatial variation of the boundary. We hypothesize that the sections of the boundary with higher variance contribute primarily to a higher LOC . Thus adding more estimation points in the high variance sections of the boundary is likely to reduce the LOC .

Our algorithm initially estimates the boundary at a small number of equidistant points. We can set $k = \lceil X_{range}/2h \rceil$ such that the boundary is estimated at every $2h$ interval, where X_{range} is the range of x values over which the boundary is being observed. Next, the sections of the boundary are sorted according to decreasing order of spatial variation and estimation points are incrementally added in that order. As more boundary points are estimated, the interpolation error reduces and LOC , the probability of the band not covering the boundary, is lower. This iterative process *converges* when additional boundary points do not lead to a further reduction in LOC . In absence of knowledge of the actual LOC , the heuristic uses another metric, *prediction error*, to decide the termination criterion for additional estimation points.

The prediction error at a specific location is the absolute difference between the observation and the estimated boundary at that location. When estimated over a set of n sensors, the probability of the prediction error being greater than δ can be used as a representative of LOC . This probability is evaluated as follows:

$$prediction\ error(\delta) = \frac{1}{n} \sum_{i=1}^n I(|\hat{d}(x_i) - y_i| > \delta) \quad (8)$$

Figure 4 illustrates the main aspect of our algorithm. It shows that with 5 points some sections of the boundary is outside the confidence band but with two additional points in sections of high variance, portions of the boundary outside the band reduces. We experimentally find that the trend of *prediction error* is similar to the LOC of boundaries (see Section 5.4 for details).

5 Experimental Evaluation of DBTR

In this section, we evaluate the performance of DBTR and its sensitivity to various parameters. The goals of our experimental evaluation are as follows: (i) to verify the effectiveness of DBTR as a combination of both spatial and

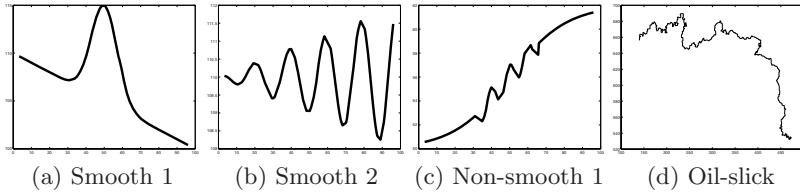


Fig. 5. Boundaries used to evaluate DBTR

temporal estimation techniques, (ii) to test its sensitivity to parameters such as specified width of band, number of estimation points and dynamics of the boundary, and (iii) to verify the effect of the adaptive update policy of DBTR on communication overheads.

5.1 Experimental Setup

Extensive simulation-based experiments are used to evaluate DBTR. DBTR algorithm is implemented in a MATLAB-based simulator. In addition, the spatial estimation technique is verified in a TOSSIM [12] based simulator and the results from these two simulators are similar. Sensors are randomly deployed in a two-dimensional field with dimension 100 units $\times$ 50 units. The communication range of sensors is 10–12 units. The maximum number of hops from the base station to sensors in a multi-hop network varies between 7–12 for different networks. We assume each sensor message contains a single observation and a single partial aggregate (explained in Section 3.1). Transmission of messages is assumed to be error-free. The error in sensor observations is assumed to be a Gaussian distribution $N(0, \sigma^2)$, where σ^2 is the error variance.

The performance of DBTR is evaluated with several boundaries generated using mathematical functions and real data traces from sensors. The boundaries in Figure 5 having different spatial variations are used as a representative set to evaluate DBTR. For example, the boundaries in Figure 5(a) and 5(b) are smooth while that in 5(c) is non-smooth. In addition, we also use a boundary (Figure 5(d)) obtained based on a real oil-slick². The boundary Smooth 1 is used as the default boundary in all experiments unless specified otherwise. Dynamic boundaries are generated using a constant mean velocity model. Assuming a continuous boundary consists of several discrete points, at every time instant, each of the boundary points is displaced by a finite distance based on the model. We consider two scenarios: (i) all points on the boundary move with the same velocity and (ii) different points move with different velocities.

5.2 Evaluation Metrics

DBTR is evaluated using two metrics: (i) *communication overhead* and (ii) *accuracy of estimated boundary*. The overall communication overhead is the cumulative number of transmissions required for the spatial estimation technique

² Data for Lake Maracaibo <http://modis.marine.usf.edu/index.html>

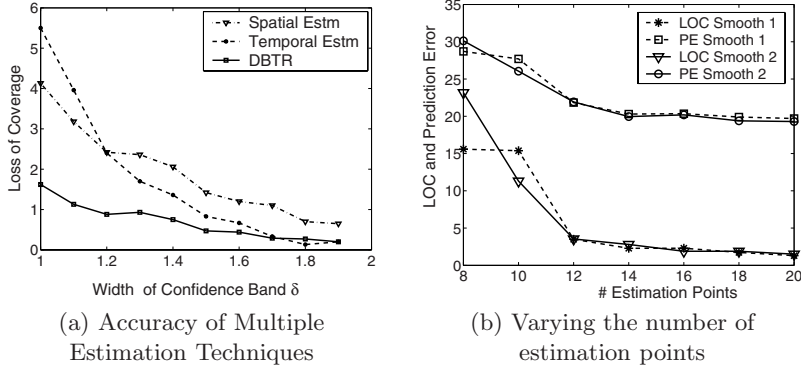


Fig. 6. Effect of the width of band and the number of estimation points on *LOC*

and the temporal estimation technique. This reflects the energy expenditure of our solution. Accuracy of the estimated boundary is measured in terms of *LOC* that is defined in Section 2. *LOC* reported is the mean value over at least 100 sets of observations.

5.3 Comparison of Boundary Estimation Techniques

In this section, we compare the performance of DBTR with both temporal and spatial estimation techniques. In the temporal *only* and spatial *only* scenarios, the confidence intervals around the boundary are updated at time instants when the boundary is expected to move by greater than $0.5 \times \delta$, as predicted by the temporal estimation. Figure 6(a) plots the *LOC* with varying δ , the width of the confidence band (*LOC* for this experiment is the mean over 20 estimated boundary points). Total duration of the experiment is 100 seconds. The velocity of boundary is 1 unit/sec and sampling interval for the sensors is 0.5 sec. We observe that for $\delta = 1 - 1.2$, DBTR performs better than both Temporal and Spatial techniques by a factor 2.8 – 2.5. For $\delta < 1.2$, DBTR as well as spatial estimation provide better performance as compared to temporal estimation. This is because the error in sensor observations is reduced due to aggregation from multiple sensors. For $\delta > 1.2$, the temporal technique provides better performance than the spatial technique. This is due to the fact that the accuracy of the temporal estimation improves if the boundary changes by $0.5 \times \delta$ less frequently. However, we observe that DBTR performs best for *all values* of δ . This is attributed to the feedback from the spatial estimation to the temporal estimation, due to which DBTR predicts the future boundary changes most accurately. This experiment demonstrates the effectiveness of combining the temporal and spatial estimation techniques in DBTR.

5.4 Impact of Estimation Points on Prediction Error and *LOC*

This experiment verifies correctness of the heuristic-based algorithm used to select k , the number of estimation points. The goal is to ensure that the prediction error

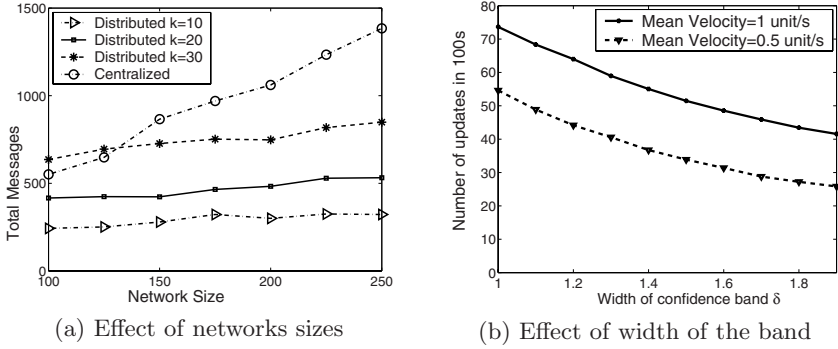


Fig. 7. Sensitivity of communication overhead to DBTR parameters

can accurately capture the trend in *LOC*. Figure 6(b) plots the prediction error function and the *LOC* for boundaries Smooth 1 and Smooth 2. We observe that with increase in k , the prediction error function (see Section 4) and *LOC* both reduce and finally stabilize for $k > 14$. Initially *LOC* decreases much more sharply (85%) as compared to prediction error (29%), but for $k > 14$, both the prediction error and *LOC* reduces by a small amount (0.5 – 0.2). The value of k for which prediction error stabilizes can be used as a good choice for the number of estimation points. Since prediction error stabilizes earlier, a few more boundary points can be further added in order to achieve the minimized *LOC*. This experiment shows that the prediction error function represents *LOC* with high fidelity.

5.5 Communication Overhead of DBTR

The communication overhead of DBTR has two components: the number of messages required by the spatial estimation technique and the number of updates as indicated by the temporal estimation technique.

Overhead due to Spatial Estimation Technique. The result of this experiment is included from [9] to show the communication overhead of the spatial estimation component of DBTR. This overhead is compared with a solution where all the observations are sent to a central server for the estimation of the confidence band. Figure 7(a) plots the total number of messages for different sizes of the network. For 20 estimation points, the communication overhead for DBTR is lower than that of the centralized solution by a factor 3.3 – 2.6. The communication overhead for the spatial estimation depends on the h -neighborhood and with increase in network size, the value of h reduces. Thus, the distributed solution of DBTR is easily scalable to larger networks. However, the communication overhead of DBTR increases in proportion with the number of estimation points which justifies reducing the number of estimation points to minimize communication overheads.

Overhead Due to Boundary Dynamics. In this experiment, we observe how the communication overhead of DBTR varies with different velocities of the

boundary. The communication overhead depends on how frequently the estimation is updated. The number of updates depend mainly on two factors: (i) the width of the estimated band and (ii) velocity of the boundary along the y axis. Figure 7(b) depicts the number of updates required for two different velocities in an interval of 100 seconds as δ is varied. All the boundary points are assumed to be changing at the same velocity. The boundary is updated only when it is expected to have changed by more than $0.5 \times \delta$. We observe that the number of updates reduce by a factor of half as δ doubles, allowing for the boundary to be updated less frequently. As expected, a faster moving boundary requires more updates and as velocity changes from 0.5 to 1 unit/s, the number of updates increase by a factor more than 1.35. The experiment is also conducted with *different* portions of the boundary changing at different velocities and it shows that DBTR is able to capture boundary dynamics for adaptive updates.

5.6 Effect of Update Policies on Accuracy

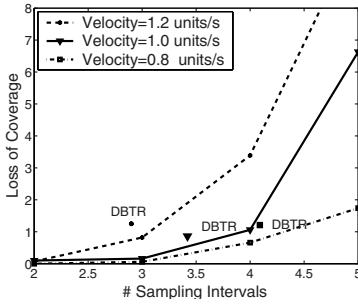


Fig. 8. Comparison of update schemes

The goal of this experiment is to compare the adaptive estimation technique of DBTR with a technique that periodically estimates the boundary. In such a scheme, the estimation based on regression is performed periodically rather than being based on the continuously predicted changes in the boundary (DBTR). The experiment is conducted for three scenarios—boundaries changing at velocities of 0.8, 1.0, 1.2 units/s. Figure 8(b) plots *LOC* versus periodicity of boundary updates for schemes with different periods. The period of updates is stated in terms of the number of sensor sampling intervals. We observe that for a certain velocity, there is an optimal period

that should be used to obtain a *LOC* of 1% or less. For example, when velocity=1.0 unit/s, *LOC* is 1.06% for a period of 4 sampling intervals. For each of the boundaries, the performance of DBTR is also shown. The period of DBTR is obtained by dividing the total duration with the number of times DBTR updates the boundary. The *LOC* is different in all three scenarios. For velocity of 1.0 unit/s, on an average DBTR updates the boundary at every 3.42 sampling intervals and achieves *LOC* of 0.86%. We note that DBTR may require lower communication overhead as it uses aperiodic updates without sacrificing accuracy obtained in a periodic scheme. Thus, the performance of DBTR is comparable with a periodic update scheme while not requiring *prior* knowledge about the dynamics of the boundary.

5.7 Summary of Results

Experimental evaluation of DBTR reveals the following important results: (i) DBTR consistently estimates boundaries more accurately than the Spatial-only

and the Temporal-only estimation techniques. (ii) The distributed in-network estimation strategy significantly reduces the communication overhead as compared to the centralized solution by a factor about 2.6 to 3.3. (iii) The accuracy and communication overhead of DBTR are similar to the optimal periodic update scheme. (iv) The heuristic for simultaneously minimizing *LOC* and the number estimation points achieves a *LOC* of less than 2% for smooth boundaries.

Additional results from a more comprehensive experimental evaluation reported in [8] are: (i) A good choice of threshold for amount of change in the boundary (in Figure 3) is $0.5 \times \delta$. It implies that for low *LOC*, the boundary should be updated before the temporal estimation technique indicates that the boundary is changed by $0.5 \times \delta$. (ii) While non-smooth boundaries require higher number of estimation points, the efficacy of DBTR is demonstrated for smooth as well as non-smooth boundaries. (iii) Initial evaluation suggests that DBTR is applicable for non-zero sensing angles provided sensors detect an adequate set of points on the boundary.

6 Related Work

DBTR uses the *spatial-temporal* correlations among sensor readings to estimate the boundary efficiently. An alternative to non-parametric regression based technique is to use parametric regression as in [13] where sensor network data is modeled in terms of basis functions. The non-parametric approach reduces the effect of observation errors by aggregation. In the parametric case, the observations are taken to be the actual values of the sensed quantity and the coefficients of basis functions are computed to obtain an estimation with minimized mean square error (MSE). The system BBQ [14] exploits correlation among sensor attributes and a probabilistic model to answer queries. While our approach works for a boundary of arbitrary shape, the multivariate gaussian distribution used by BBQ may not be applicable.

An alternative to the model-based approach using Kalman Filter is simple state space models [15]. If the individual boundary points follow non-linear dynamics or have non-Gaussian errors more advanced techniques like particle filter [16] can be used. Switching Kalman Filters can be used to monitor boundaries with non-stationary dynamics (e.g., a storm) as discussed in [17].

There is a large amount of work dealing with contour extraction [18] using sensor networks. While a boundary detection technique is useful in detecting the presence of a phenomenon (either plume or fire), a contour extraction technique can provide more detailed information about the phenomenon. DBTR is most similar to the boundary estimation technique proposed by Nowak et al. [2]. The main difference is that DBTR tracks a dynamic boundary without incurring significant communication overhead, but there is no easy way for extending their technique to track dynamics apart from periodically recomputing the boundary. While DBTR provides a non-parametric estimation of the boundary, their technique provides a staircase-like approximation of the boundary. DBTR's adaptive selection of estimation points is also similar to the adaptive sampling method [19]

which consists of two phases- a preview phase of collecting observations followed by a refinement phase in regions containing the boundary. DBTR attempts to minimize the locations for estimation points whereas their approach achieves a minimax bound on MSE.

DBTR makes use of sensors with range/remote capabilities for detecting a moving boundary. Another application using remote sensing is CASA [5], where a network of radars is used for meteorological monitoring to detect tornadoes.

7 Conclusion

We have developed a technique for dynamic boundary estimation in sensor networks where observations from range sensors are aggregated and a confidence band around the true boundary is obtained from estimates at a few selected locations. In addition, the temporal correlation among observations at certain points is utilized to develop a Kalman Filter based technique for estimating the changes in the boundary. This strategy updates the estimates before the boundary is expected to move out of the confidence band. Thus, our solution provides confidence band with high accuracy around the actual boundary at all times with low communication overheads that a suitable periodic scheme cannot achieve without prior knowledge about the dynamics of the boundary.

As part of future work, we propose to study the parametric regression technique for estimating boundaries in sensor networks. We propose to explore the impact of non-zero sensing angles on the accuracy of estimation. We also plan to extend our strategy to include in-situ measurements for detecting a boundary. Another way to extend our work is to consider more complex models for the dynamics of the boundary.

Acknowledgment. We would like to thank Parmesh Ramanathan of University of Wisconsin, Madison for his valuable suggestions.

References

- [1] Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: Wireless sensor networks: a survey. Elsevier Journal of Computer Networks 38, 393–422 (2002)
- [2] Nowak, R., Mitra, U.: Boundary Estimation in Sensor Networks: Theory and Methods. In: Zhao, F., Guibas, L.J. (eds.) IPSN 2003. LNCS, vol. 2634, Springer, Heidelberg (2003)
- [3] Ding, M., Chen, D., Xing, K., Cheng, X.: Localized Fault-Tolerant Event Boundary Detection in Sensor Networks. IEEE INFOCOM 2, 902–913 (2005)
- [4] Hsieh, C.H., Jin, Z., et al.: Experimental validation of an algorithm for cooperative boundary tracking. In: Proc. of American Control Conference (2005)
- [5] Zink, M., Westbook, D., Abdallah, S., Horling, B.: Meteorological command and control: An end-to-end architecture for a hazardous weather detection sensor network. In: EESR. Workshop on End-to-End, Sense-and-Respond Systems, Applications, and Services (2005)

- [6] Utkin, A.B., Lavrov, A.V., Costa, L., Simoes, F., Vilar, R.: Detection of Small Forest Fires by Lidar. *Applied Physics B* 74, 77–83 (2002)
- [7] Lavrov, A., Vilar, R.: Application of lidar at 1.54 μm for forest fire detection. In: *Remote sensing for earth science, ocean, and sea ice applications*, vol. 3868, pp. 473–477 (1999)
- [8] Duttagupta, S., Ramamritham, K., Kulkarni, P.: Tracking dynamic boundaries using sensor networks. In: *IIT Bombay, CSE department Technical Report TR-CSE-2007-7* (2007)
- [9] Duttagupta, S., Ramamritham, K., Ramanathan, P.: Distributed Boundary Estimation using Sensor Networks. In: *The 3rd IEEE Conf. on Mobile Ad-hoc and Sensor Systems* (2006)
- [10] Härdle, W.: *Applied Nonparametric Regression*. Cambridge University Press, Cambridge (1990)
- [11] Gasser, T., Kneip, A., Kohler, W.: A Flexible and Fast Method for Automatic Smoothing. *Journal of the American Statistical Association* 86(415), 643–652 (1991)
- [12] Levis, P., Lee, N., Woo, A., Welsh, M., Culler, D.: TOSSIM: Accurate and scalable simulation of entire tinyos applications. In: *Sensys. Proc. of 1st ACM Conf. on Embedded Networked Sensor Systems* (November 2003)
- [13] Guestrin, C., Bodik, P., Thibaux, R., Paskin, M., Madden, S.: Distributed Regression: an Efficient Framework for Modeling Sensor Network Data. In: *IPSN* (2004)
- [14] Deshpande, A., Guestrin, C., Madden, S., Hellerstein, J.M., Hong, W.: Model-based Approximate Querying in Sensor Networks. *VLDB Journal* (2005)
- [15] Liu, J., Chu, M., Liu, J.: Distributed State Representation for Tracking problems in Sensor Networks. In: *IPSN* (2004)
- [16] Coates, M.: Distributed Particle Filtes for Sensor Networks. In: *IPSN* (2004)
- [17] Manfredi, V., Mahadevan, S., Kurose, J.: Switching Kalman Filters for Prediction and Tracking in an Adaptive Meteorological Sensing Network. In: *SECON. IEEE Conference on Sensor and Ad Hoc Communications and Networks* (2005)
- [18] Liao, P., Chang, M., Kuo, C.J.: Contour Line Extraction with Wireless Sensor Networks. In: *IEEE Conference on Communications* (2005)
- [19] Singh, A., Nowak, R., Ramanathan, P.: Active Learning for Adaptive Mobile Sensing Networks. In: *IPSN* (2006)