

Tracking Dynamic Fronts using Sensor Network

Subhasri Duttagupta, Krithi Ramamritham and Purushottam Kulkarni
Department of Computer Science and Engineering
Indian Institute of Technology, Bombay, India

Abstract—We examine the problem of tracking dynamic boundaries occurring in natural phenomena using a network of range sensors. Two main challenges of the boundary tracking problem are accurate boundary estimation from noisy observations and continuous tracking of the boundary. We propose Dynamic Boundary Tracking (DBTR), an algorithm that combines the *spatial estimation* and *temporal estimation* techniques to effectively track a dynamic boundary. The regression-based spatial estimation technique determines discrete points on the boundary and estimates a confidence band around the entire boundary. In addition, a Kalman Filter-based temporal estimation technique tracks changes in the boundary and aperiodically updates the spatial estimation to meet accuracy requirements. DBTR, provides a low communication overhead solution to track boundaries without requiring *prior* knowledge about the dynamics. Experimental results demonstrate the effectiveness of our algorithm; estimated confidence bands indicate a *loss of coverage* of less than 2 – 5% for a variety of boundaries with different spatial characteristics.

Index Terms—Boundary Estimation, Kalman Filter, Nonparametric Regression, Range Sensors

I. 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. In many of these applications, front-tracking is an important aspect. In a forest fire tracking application, one of the requirements is to provide the first respondent with continuous updates regarding the *extent* and *direction* of the fire-front [1] and its distance from habitats. As the front moves, the information regarding its latest position should be updated. For instance, at a particular time instant t_0 , the front may be 2 km away with respect to a location while at $t_0 + 30$ minutes, it may be 1.2 km away. In the above situations, the sensing field may be partitioned into two or more regions of distinct behaviors — one set of regions where the condition of fire being present is true and another set of regions where it is false. Strategically deployed sensors can be used to detect and track boundaries associated with such natural phenomena if a well-defined *boundary* separates the two sets (fire present/absent) differing in physical and/or chemical properties.

Previous techniques to estimate boundaries have employed *in-situ* stationary sensors [2], [3] or mobile sensors [4], [5]. In applications like tracking a plume, or predicting trajectory of weather parameters [6], in-situ sensing is not feasible due to difficulty in access to remote areas 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. In range sensing, a sensor estimates distance to a remote location where the phenomenon is present, whereas in in-situ sensing, an observation contains a field value at the sensor location.

There are many possible applications of range sensors. For example, radars used in [6] scan an angular area by sweeping up-to 360 degrees and gather reflectivity and wind velocity information. Recently, Lidars (Light Detection and Ranging) are being used for detecting forest fires [7], [8]. Lidars detect fire by analyzing the energy back-scattered from smoke particles resulting from the fire and measure the distance between lidar sensor and a point on the target(smoke) using simple principles of light. While today lidars are not equipped for wireless communication, we envision in the near future, low power, inexpensive sensors with radar/lidar distance sensing and wireless communication capability will be available. Further, prototypes such as Radar Motes [9] that integrate a Mica2 mote and ultra-wideband radar motion sensor, prove that such sensors are feasible from a technical standpoint. In the rest of the paper, we assume that a network of *wireless sensors with range sensing* is used to track boundaries occurring in natural phenomena.

Remote sensors used for boundary tracking applications may be deployed in regions with limited or no infrastructure support. Assuming sensors meet their power requirements from sources such as solar panels or battery, these sensor nodes operate under a constraint of limited energy consumption. For example, a 10 km radar capable of mapping 20 dBZ reflectivity combined with a low power computing platform that may be operated using a solar powered system is proposed in [10]. Further, battery-operated handheld laser devices are commonly used in many applications such as for measuring local atmospheric visibility [11] or detecting gas, vapor plumes [12]. With such energy-constrained range sensors, a primary consideration in tracking a dynamic boundary is *energy conservation*. We aim to reduce the communication overhead at sensor nodes to increase the lifetime of the sensor network.

Another characteristic of sensor networks is *noisy observations*. Due to inherent inaccuracy of range sensors, individual observations are noisy. On the other hand, accuracy in front estimation is critical since in many front tracking applications, important decisions may be taken based on the current location of the front. The challenge is to achieve high accuracy in boundary estimation in spite of noisy observations. Other challenges in tracking dynamic fronts include *timely updates* of the estimates while keeping *communication overheads* low.

Boundaries occurring in natural phenomenon may or may not have any specific form. Moreover, depending on the environmental conditions or sources of the phenomenon, the fronts occurring in two situations may be very different. Tracking fronts using sensor networks involves estimating its shape and location in a two dimensional field. We refer to this as the *spatial variations* of the front. From the spatial variations, it should be possible to obtain the distance of the front from a specific location of interest (e.g., the nearest habitat). Boundary changes with time are referred to as *temporal variations*. A dynamic front has both

spatial and temporal variations. In order to estimate a boundary accurately both of these variations should be captured. Existing boundary estimation strategies [2],[24], [3] using *in-situ* sensors do not address the issue of tracking dynamic boundaries. While proposed solutions using mobile sensors [27] [4] can be tailored for tracking dynamic boundaries, no optimization in terms of communication overheads is attempted as the focus is on reducing energy and delay due to mobility.

A. Problem Formulation

We assume a set of 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. 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. An observation (x_i, y_i) of the i^{th} sensor represents the location¹ of a boundary point. Figure 1(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 angle θ w.r.t. the y axis and detects a point (x_i, y_i) along the sensing direction. α_i captures the error in observation.

Given n observations $\{(x_i, y_i)\}_{i=1}^n$ with errors, our goal is to estimate a confidence band of a specific width δ (the distance between the estimated boundary and the extent of the band in each direction) as shown in Figure 1(b). The confidence band should cover the dynamic boundary at all times and with high probability. Accuracy of the band is measured in terms of *loss of coverage (LOC)*, the probability of the band not covering the actual boundary. We define *estimation point* as an arbitrary location along the x axis from which distance to the boundary is estimated along the y axis. For an arbitrary estimation point x_i , if $d(x_i)$ is the actual distance to boundary and $\hat{d}(x_i)$ is the estimated distance, then *LOC* over a set of n such estimation points is defined as:

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

where $I(\cdot)$ is an indicator function such that $I(a) = 1$ only if the parameter a is true, otherwise $I(a) = 0$. Minimizing *LOC* implies maximizing accuracy of coverage. In this paper, wherever a band is mentioned we refer to a confidence band.

Since sensors are energy-limited, we aim to reduce communication overheads at sensor nodes to increase the lifetime of the sensor networks. Hence, two important metrics of our solution are the accuracy of estimation (expressed in *LOC*) and communication overheads.

B. Contributions

We propose Dynamic Boundary Tracking (DBTR), a novel technique that intelligently combines both spatial and temporal estimation techniques for accurate dynamic boundary estimation. Specifically our contributions are as follows:

¹ y_i is the y coordinate of the measured boundary point's location and not the relative distance from the sensor

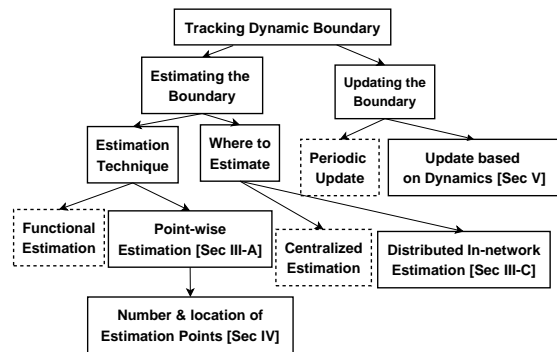


Fig. 2. Solution Aspects of Dynamic Boundary Tracking

- The point-wise approach of DBTR allows handling different sections of a dynamic boundary separately. DBTR is designed to be effective for tracking dynamic boundaries without prior knowledge regarding dynamics of the boundary.
- DBTR's spatial estimation scheme is designed such that it lends itself to in-network aggregation. This strategy reduces the communication overheads as compared to a centralized solution.
- With the model-based prediction of dynamics at individual estimation points, DBTR requires less communication overheads than the best *optimal* periodic update scheme with similar accuracy.
- DBTR consistently estimates boundaries more accurately than the individual Spatial and Temporal estimation techniques. The estimated confidence band around a boundary has *loss of coverage* of less than 2 – 5% for a variety of boundaries with different spatial characteristics.

The rest of the paper is organized as follows. Section II presents the overview of solution approach. Section III describes non-parametric regression-based spatial estimation technique and shows its actual realization in a sensor network. Section IV presents DBTR's approach for selecting the locations and number of estimation points in estimating a boundary. Section V discusses the temporal estimation technique using Kalman Filter and Section VII evaluates spatial and temporal estimations through simulations. Section VI presents the DBTR algorithm as a combination of spatial estimation and temporal estimation. Section VIII discusses the related work and the conclusions are drawn in Section IX.

II. OVERVIEW OF SOLUTION APPROACH

Figure 2 outlines the issues involved in tracking a dynamic boundary. The two main issues are estimating the boundary and updating the estimate as the boundary moves. There are two broad techniques to estimate boundaries: (i) *Functional estimation* and (ii) *Point-wise estimation*. The main assumption of functional estimation is that the entire boundary can be represented using a function. For boundaries occurring in natural phenomenon, a preselected functional form might be too restrictive for reflecting all important features. A common functional form is also likely to have large bias in the estimation. These problems can be overcome in a point-wise estimation technique which assumes that boundaries consist of discrete points. The effectiveness of this technique depends on the number and locations of boundary

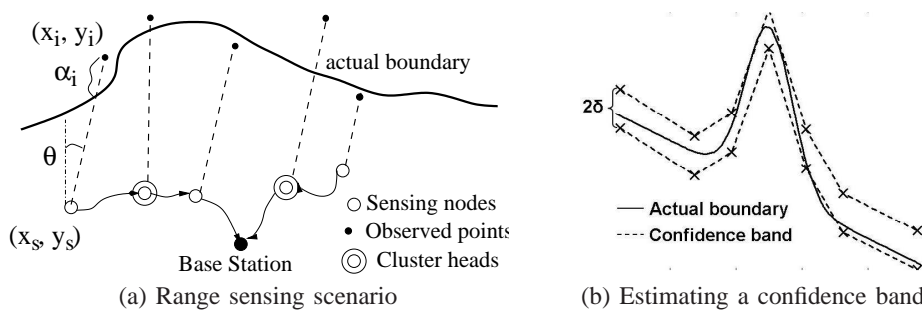


Fig. 1. Tracking dynamic boundary using range sensing observations.

estimation points. Our point-wise technique exploits spatial variations to determine the positions of estimation points and uses only a minimal number of estimation points.

For estimating a boundary, we use a non-parametric regression-based *spatial estimation* technique that exploits spatial correlations of proximate sensor observations to reduce the effect of range sensing errors. A centralized technique for estimating boundaries is to transmit all sensor observations to a central location. But this technique suffers from high communication overheads. We explore a decentralized solution that utilizes sensors' local computation capability and performs *in-network* aggregation at sensors within the network to significantly reduce the communication overheads for boundary estimation.

In order to track a dynamic boundary, the boundary estimates need to be continuously updated. An important question then is, when should the boundary estimates be updated? We show how to estimate and exploit temporal characteristics of the boundary to update its estimate only when required. The time instances when a boundary estimate is updated depend on the dynamics of the boundary. Optimal period for update is the longest time period during which the boundary does not need update without affecting the accuracy. But, unless there is clairvoyance, optimal choice of periodicity at each point is not possible in real-time tracking scenarios. Our Kalman filter-based *temporal estimation* technique predicts the movement of the boundary and updates the estimate only when error in the current estimate exceeds a pre-defined threshold.

III. SPATIAL ESTIMATION TECHNIQUE OF DBTR

In this section we discuss the spatial estimation technique of DBTR. Given a set of noisy observations, we address the question: *how to estimate the current boundary such that the estimate has bounded inaccuracy?* A simple way to estimate the boundary is to apply a polynomial function or a trigonometric function to the observations. Such a *parametric* approach is useful for deriving a sketchy boundary and may be insufficient for extracting important characteristics of the boundary. We use a non-parametric regression based kernel-smoothing technique since it allows deriving a smooth boundary without adhering to a specific functional form.

Initially, the mathematical background of non-parametric regression is presented, then the issues regarding practical implementation of the spatial technique in a sensor network are discussed.

A. Non-parametric Regression: Mathematical Background

A set of noisy observations $\{(x_i, y_i)\}_{i=1}^n$ is the input to DBTR where y_i is taken as the distance to the boundary from location x_i . Due to the presence of noise on observations, the measured distances are not exact. As range sensors find distances to the boundary, it is possible to derive a relationship between a sensor location and the observed distance to the boundary. Given that a regression curve describes a general relationship between an independent variable and dependent variable, a regression relationship is used to obtain the expected distance to boundary from a given location. 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,

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

where d is the regression relation between x_i and y_i , and α_i is the observation error along y axis. For simplicity, the spatial estimation technique of DBTR is described assuming range sensors are aligned along the y axis. Thus, θ , the orientation angle of the sensors, is set to zero. This assumption is relaxed in Section VII-J.

If the error distribution has mean zero, then the expected value of the distance to the boundary at x_i is $d(x_i)$. Though the exact error model depends on the actual range sensing techniques, for simplification,² the error distribution is taken as normal $N(0, \sigma^2)$, where σ^2 is the observation error variance. Assuming a smooth boundary, it is possible to use a local average of the observations near x_p to construct $\hat{d}(x_p)$, an estimate for $d(x_p)$.

The *kernel smoothing* [13] technique that uses observations in the neighborhood of x_p is applied to estimate $\hat{d}(x_p)$. Thus $\hat{d}(x_p)$ is,

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

where $\{W_i(x_p)\}_{i=1}^n$ denotes a sequence of weights defined using a *kernel* function. The following weights proposed by the Nadaraya [14] are used:

$$W_i(x_p) = \frac{K\left(\frac{x_p - x_i}{h}\right)}{\frac{1}{n} \sum_{j=1}^n K\left(\frac{x_p - x_j}{h}\right)} \quad (4)$$

where $K()$ is a kernel function and the weight for x_p is non-zero only if $|x_p - x_i| \leq h$. The parameter h is referred to as the *h-neighborhood* of x_p . There are a number of optimal kernel

²In case of range sensors, observation noise distribution tends to have a non-Gaussian component if there is no line-of-sight sensing. We assume that there is no non line-of-sensing.

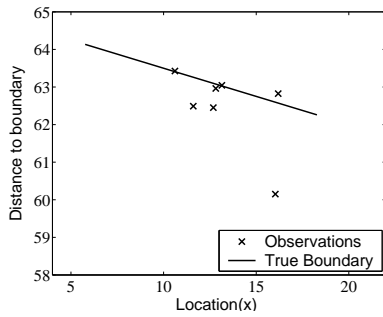


Fig. 3. Sample noisy observations around an estimation point $x = 13.1$

functions suggested in the literature. A simple *Epanechnikov kernel* [13] with parabolic shape is: $K(u) = 0.75(1-u^2) \times I(|u| \leq 1)$. Note that the numerator of the weight $W_i(x_p)$ is a kernel value evaluated at x_p whereas the denominator is the mean of kernel values evaluated at all values of x_i . This mean kernel value reflects the *density* of x_i values at x_p . Such a weight sequence serves two purposes: (1) the weights adapt to the local density at x_p (density of sensors along x axis) (2) weight sequence for x_p sum to one.

After plugging in the weight sequence, Equation (3) becomes,

$$\hat{d}(x_p) = \frac{\frac{1}{n} \sum_{i=1}^n K_h(x_p - x_i) y_i}{\frac{1}{n} \sum_{i=1}^n K_h(x_p - x_i)} \quad (5)$$

where $K_h(u) = K(\frac{u}{h})$ is used for simplification. Further, $\hat{\sigma}^2(x_p)$, the variance of observations at x_p , is estimated as:

$$\hat{\sigma}^2(x_p) = \left(\frac{1}{n} \sum_{i=1}^n W_i(x_p) y_i^2 \right) - \hat{d}^2(x_p) \quad (6)$$

$\hat{\sigma}^2(x_p)$ captures the spatial variation of the boundary and is also an estimate of the observation error variance σ^2 . *Whenever multiple observations with the same variance are averaged, the variance of average reduces with increase in the number of observations.* This is why we expect higher accuracy of estimation for the estimated distance $\hat{d}(x_p)$ compared to just observations which are noisy. This is also verified in Section VII-J.

DBTR uses this variance information for deciding the locations of estimation points which is discussed in Section IV. Figure 3 illustrates how to estimate a boundary from noisy observations using non-parametric regression. Let $x_p = 13.1$ be an estimation point. 7 noisy observations in h -neighborhood of the estimation point are (10.6, 63.4), (11.6, 62.5), (12.6, 62.4), (12.8, 62.9), (13.1, 63.0), (16.0, 60.2), (16.1, 62.8) as shown in Figure 3(a). Applying the kernel function in Equation (4), the weights for these observations are 0.10, 0.17, 0.2, 0.2, 0.2, 0.07, 0.05. Next, applying these weights to observations using Equation (5), we get the estimated boundary at $\hat{d}(x_p)$ as 63.04.

B. Steps in Estimating a Boundary

In a *point-wise* approach based on non-parametric regression, a boundary can be estimated using the following steps:

- 1) Select a set of estimation points where the boundary will be estimated. We assume that the number of estimation points is k . The value of k and locations of the estimation points are discussed in Section IV.

- 2) Find optimal neighborhood for this set of estimation points using the technique mentioned above.
- 3) Estimate $\hat{d}(x_p)$, distance to the boundary using Equation (5) at these estimation points. Thus, we estimate k points that lie approximately on the boundary.
- 4) Estimate the entire boundary using the smoothing spline [13] interpolation scheme from these k point-wise estimates.
- 5) Obtain a confidence band, by adding δ distance on both sides, centered around the estimated boundary as shown in Figure 1(b).

The next relevant question is: *how do we evaluate the distance to the boundary and the variance of observations in a sensor network?* The expressions for $\hat{d}(x_p)$ and $\hat{\sigma}^2(x_p)$ reflect that they are aggregate functions. Hence, we need a suitable aggregation technique to realize these expressions from sensor observations. This is addressed in the following section.

C. Realizing the Regression-based Approach in a Sensor Network

Our system model consists of multi-hop communication network connecting the sensor nodes and the base station. Realizing the spatial estimation technique in such a network involves deciding how the observations are gathered for estimating the boundary. There are two basic approaches for estimating an expression in sensor networks. In a centralized solution, all the observations are sent to a base station that performs the computation. Conversely, in a distributed solution, the observations can be combined through in-network aggregation and computations can be performed in an incremental fashion at intermediate nodes. The distributed solution tends to be more scalable as fewer communication messages are involved. DBTR uses a cluster-based data dissemination scheme. For a distributed implementation, Equations (3) and (6) are amenable to being split into expressions that can be evaluated at the distributed locations in the routing tree.

1) *Cluster-based In-network Aggregation:* In our approach, sensors are organized into clusters and cluster heads are selected. Nodes within a cluster are one or two hops away from the cluster heads so that the observations from individual nodes can easily be aggregated at cluster heads. Any of the cluster formation algorithms suggested in the literature can be used in DBTR. We use the cluster formation algorithm suggested in [15] as it generates minimal number of clusters which can help in minimizing the number of data aggregation messages. It is assumed that the base station is always a cluster head. All cluster heads form a routing tree structure rooted at base station and every cluster head has an intermediate parent node through which it can forward data to base station. In the routing tree, each leaf node sends its observation to the respective cluster head.

2) *Estimating Distance to the Boundary:* Since the regression function, $\hat{d}(x_p)$ and the variance $\hat{\sigma}^2(x_p)$ are duplicate-sensitive aggregates, cluster-based boundary estimation scheme should ensure that observation from a node is included in the estimate at most once. The technique for computing the regression function at x_p is as follows. The observation from a node with x_i coordinate is used for computing boundary point $\hat{d}(x_p)$ if it satisfies the relationship $x_p - h \leq x_i \leq x_p + h$. The expression for $\hat{d}(x_p)$ in Equation (5) can be written in terms of a fraction $\frac{N}{D}$ where the numerator N is the sum of individual observations multiplied by

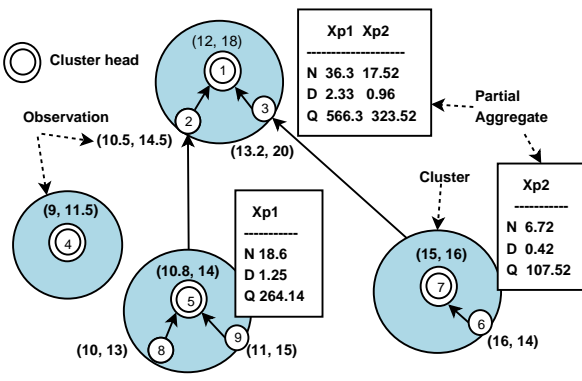


Fig. 4. Cluster heads computing \hat{d} and $\hat{\sigma}^2$ for $x_{p1} = 11.5$ and $x_{p2} = 14$

the kernel evaluated at each x_i and the denominator D is the sum of kernel values evaluated within the h -neighborhood of x_p .

We show that if observations with h -neighborhood of x_p are aggregated separately by a few nodes, by combining individual aggregates, $\hat{d}(x_p)$ is computed. If two terms $N_i(x_p)$, $D_i(x_p)$ are estimated for each observation, then $\hat{d}(x_p)$ can be evaluated as follows:

$$\hat{d}(x_p) = \frac{\sum_{i=1}^n K_h(x_p - x_i) y_i}{\sum_{i=1}^n K_h(x_p - x_i)} = \frac{\sum_{i=1}^n N_i(x_p)}{\sum_{i=1}^n D_i(x_p)} \quad (7)$$

where for $i = 1, \dots, n$:

$$N_i(x_p) = K_h(x_p - x_i) y_i, \quad D_i(x_p) = K_h(x_p - x_i) \quad (8)$$

If a cluster head has multiple observations within h -neighborhood of x_p , it aggregates all the $N_i(x_p)$, $D_i(x_p)$ terms corresponding to these observations and obtains aggregate $N_i(x_p)$, $D_i(x_p)$ terms which we refer to as the *partial aggregates* for $\hat{d}(x_p)$. These partial aggregates are sent through the routing tree to the base station where $\hat{d}(x_p)$ is estimated. If partial aggregates from multiple cluster heads reach an intermediate node in the routing tree, the $N_i(x_p)$, $D_i(x_p)$ terms are separately aggregated to obtain a new partial aggregate. Thus, multiple cluster heads can estimate $\hat{d}(x_p)$ in a distributed manner.

3) *Estimating Observation Variance*: Using a similar technique, multiple nodes in the network can compute the variance of observations in a distributed fashion. The computation of variance involves taking the square of observations from the individual nodes and using the value of final $\hat{d}(x_p)$. If two terms $Q_i(x_p)$, $D_i(x_p)$ are estimated for each observation within h -neighborhood of x_p , then the variance of a boundary point can be computed by aggregating all these terms using the following approach:

$$\begin{aligned} \hat{\sigma}^2(x_p) &= \frac{\sum_{i=1}^n K_h(x_p - x_i) y_i^2}{\sum_{i=1}^n K_h(x_p - x_i)} - \hat{d}^2(x_p) \\ &= \frac{\sum_{i=1}^n Q_i(x_p)}{\sum_{i=1}^n D_i(x_p)} - \hat{d}^2(x_p) \end{aligned} \quad (9)$$

where $D_i(x_p)$ is as given in Equation (8) and $Q_i(x_p)$ is given as:

$$Q_i(x_p) = K((x_p - x_i)/h) y_i^2 \quad (10)$$

Thus, multiple cluster heads compute partial aggregates consisting of (N_i, D_i, Q_i) terms for estimating $\hat{d}(x_p)$ and $\hat{\sigma}^2(x_p)$ in a distributed manner. An example of computing partial aggregates at cluster heads for points $x_{p1} = 11.5$ and $x_{p2} = 14$ with

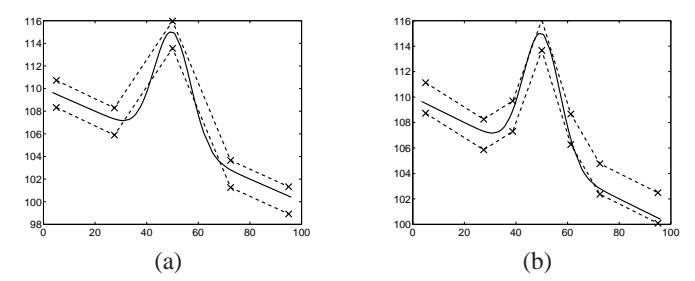


Fig. 5. (a) Confidence Band for 5 estimation points (b) Confidence Band for 7 estimation points

$h = 1.5$ is shown in Figure 4. Cluster head 5 estimates partial aggregate for x_{p1} which is sent to node 2 and cluster head 7 estimates partial aggregate for x_{p2} and is sent to node 3. Cluster head 4 has no partial aggregate as it has no observation inside the h -neighborhood of either x_{p1} or x_{p2} . Node 1 (which is the base station) collects partial aggregates from node 2 and 3 and estimates $\hat{d}(11.5) = 15.5$ and $\hat{\sigma}^2(11.5) = 2.8$, $\hat{d}(14) = 18.3$ and $\hat{\sigma}^2(14) = 3.9$.

IV. TOWARDS MINIMIZING THE NUMBER OF ESTIMATION POINTS

Communication overheads for the spatial estimation technique depend on many factors such as the depth of the routing tree, node density and the number of estimation points. For each estimation point, separate partial aggregates are built and sent to a central location. Because of this, communication are directly proportional to the number of estimation points (verified through an experiment in Section VII-F). This suggests that the boundary should be estimated with low number of estimation points to reduce communication overheads.

On the other hand, more estimation points help in reducing interpolation error of the estimated confidence band. The spatial estimation uses an interpolation scheme over a set of estimated points to derive the entire boundary. Figure 5 (a) and (b) show a confidence band around a boundary and illustrate the effect of adding more estimation points on the interpolated boundary where each 'x' indicates the boundary estimated at an estimation point and the dashed line represents the confidence band. Actual boundary is represented by the solid line. It shows that with 5 points some sections of the boundary are outside the confidence band while with two additional points, the band includes some more sections of the boundary. Thus, a larger number of estimation points may improve the accuracy of estimation but possibly incur higher communication overheads. Our goal is two fold:

- 1) to estimate boundary at a minimal number of points, and
- 2) to ensure that the band obtained through interpolation, covers the actual boundary with high confidence.

The coverage of the entire boundary will be affected if any section of the boundary lies outside the band. This suggests that the band should capture any significant spatial changes. The question arises: *how do we capture the spatial variations of the boundary?*

Properties of a boundary should be utilized in order to determine the number of estimation points. Variance $\hat{\sigma}^2(x_p)$ as given in Equation (6) incorporates two types of information: (i) spatial changes of the boundary and (ii) error in observations. Assuming that all observation errors follow a normal distribution,

the variance of observations at different x_p differ due to spatial variations. A higher value of the variance indicates a larger spatial variation of the boundary within the h -neighborhood. We hypothesize that *the sections (portions between two estimation points) of the boundary with higher variance are the primary contributors to a higher LOC*. Thus, adding more estimation points in the high variance sections of the boundary is likely to reduce the *LOC*.

A. Algorithm for Selecting Estimation Points

We present a heuristic-based approach for selecting a minimal set of estimation points. Our algorithm initially estimates the boundary at a small subset of sensor locations. We observe through experiments that if estimation points are evenly distributed along the boundary, the estimated band provides a good coverage. Hence, the initial set of estimation points consists of points located approximately at every $2h$ interval such that the boundary is estimated at least once in every h -neighborhood. If X_{range} is the range of x values, then the number of estimation points, $k = \lfloor X_{range}/2h \rfloor$. Note that x locations of a set of sensors are taken as estimation points.

The spatial estimation technique also estimates variances at these estimation points. At the base station, different sections of the boundary are sorted according to decreasing order of estimated variance and additional estimation points are added in that order. For example, if three estimation points are 19.9, 24.4, 30.9, and variances at these points are 1.2, 1.8 and 1.72, then an additional estimation point is added after 24.4. Since the base station knows all sensor locations, a sensor location, preferably close to the middle point of two existing estimation points, is chosen as the new estimation point. Based on sensors' x locations, new estimation can be at $x = 27.3$. Thus, locations of sensors residing in high variance sections are likely to be chosen as additional estimation points.

As more number of boundary points are estimated, the interpolation error reduces and so does the loss of coverage (*LOC*). This iterative process *converges* when additional boundary points do not lead to a further reduction in *LOC*. Determining the exact *LOC* requires the knowledge of actual boundary. Consequently we use another metric, *prediction error*, to decide when to stop adding more estimation points.

B. Prediction Error Representing LOC

Prediction error at a specific estimation point is defined as the absolute difference between the observation and the estimated boundary at that location. When estimated over a set of n estimation points, the probability of this difference being greater than $2 \times \delta$ can be used as a representative of *LOC*. We justify the factor of 2 later in this section. Prediction Error function (PE(δ)) for a given δ is approximated as follows:

$$PE(\delta) = \frac{1}{n} \sum_{i=1}^n I(|\hat{d}(x_i) - y_i| > 2\delta) \quad (11)$$

where y_i is the observation and $\hat{d}(x_i)$ is the estimated distance at x_i . Though *LOC* and *prediction error* are defined using indicator functions, if obtained as a mean over a large set of observations, these quantities represent the corresponding probabilities. Assume that *LOC* at x_p is C , i.e., $P[|\hat{d}(x_p) - d(x_p)| > \delta] = C$

Then $P[|\hat{d}(x_p) - d(x_p)| < \delta] = 1 - C$. In Section III-A, we made the assumption that observation noise have a normal distribution $N(0, \sigma^2)$. Given the above probability and if δ is close to the variance of sensor observation noise, then the following statements hold true based on the *three sigma rule* for normal distribution.

$$P[|d(x_p) - y_p| < 3\delta] \approx 1 \Rightarrow P[|\hat{d}(x_p) - y_p| < 2\delta] = 1 - C$$

This implies prediction error at x_p is close to C .

In the experimental section, we verify that the prediction error and *LOC* follow similar trends. As a result, prediction error is used as a metric for the termination condition of the iterative algorithm presented above. Further, we observed that *LOC* does not change based on exact locations of estimation points, it only depends on the value of k provided estimation points are well separated along the boundary.

V. TEMPORAL ESTIMATION TECHNIQUE OF DBTR

The spatial estimation technique described in the last section can be used to track a static boundary. For tracking a dynamic boundary, the estimates from the spatial technique should be refreshed at appropriate periodic intervals to ensure that the boundary is always within the estimated band. But this requires prior knowledge of the boundary dynamics such that the period can be adjusted. The temporal estimation technique of DBTR uses the history of observations to decide *when* to update the boundary. It builds a model to predict the changes in the boundary and aperiodically refreshes the spatial estimates. Further, while tracking a moving boundary, the implicit assumption is that the sensor sampling period is chosen correctly. In this section, we also address the issue of selecting the sensor sampling period according to the dynamics of the boundary.

A. Model-based Temporal Estimation Technique

The temporal estimation technique uses a model for the dynamics, built using a time sequence of observations of the distance to the boundary. In practice, the model is dependent on the exact application scenario. For example, to model plume's mobility from *in situ* measurements of the concentration of plume, Gaussian plume dispersion model or cloud model [16] can be used. But in our problem, the sensor observations are related to locations of the boundary and not directly affected by physical/chemical characteristics of the boundary. Specifically, we are interested in modeling the *velocity* of the boundary. External factors such as the prevailing weather conditions, surrounding topography etc., can affect the dynamics of boundary. If external factors can be modeled as Gaussian error and the underlying physical process changes linearly with time, then simple tracking models like Kalman Filter can be used. Assuming that the boundary at a discrete point changes in a linear fashion with time, we use a Kalman Filter to predict the future boundary locations. Here, a sensor maintains a state representation of distance to the boundary that is updated at each time step.

Kalman Filter state $s(x_p, t_i)$ consists of the actual distance $d(x_p, t_i)$ to the boundary at x_p and the velocity of the boundary along the y axis at time instant t_i .

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

$\dot{d}(x_p, t_i)$ denotes the change in $d(x_p, t_i)$ with respect to time. Irrespective of the actual movement of the boundary, we are interested in knowing only the change in $d(x_p, t_i)$, i.e., the y component of the velocity of the boundary, at x_p . Since a range sensor can find the distance to a dynamic boundary without changing its own location, it helps in reducing the *complexity* of the problem. Sensor at an estimation point tracks changes in distance to the boundary from that point.

For simplicity, we assume that the boundary moves with a *constant mean velocity* having a mean zero and random acceleration. Then the state space equation becomes:

$$s(x_p, t_i) = F \times s(x_p, t_{i-1}) + G \times \alpha_{proc}(x_p, t_{i-1}) \quad (12)$$

where $\alpha_{proc}(x_p, t_{i-1})$ is a Gaussian error with distribution $N[0, \sigma_{proc}^2]$. The matrix F relates the state at time t_i to the state at time t_{i-1} . The term $G \times \alpha_{proc}(x_p, t_{i-1})$ 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, t_s can be same as the sampling period of sensors.

Temporal technique estimates the distance to the boundary at x_p using the observation matrix as $d(x_p, t_i) = H \times s(x_p, t_i)$ where $H = [1 \ 0]$ is the observation matrix. Using this relationship and Equation (2) regarding sensor observation we obtain:

$$y(x_p, t_i) = H \times s(x_p, t_i) + \alpha(x_p, t_i) \quad (13)$$

where $\alpha(x_p, t_i)$ is the error in sensor observations and has distribution $N[0, \sigma^2(x_p)]$. The variance at estimation point x_p , $\sigma^2(x_p)$ (Equation 6), is used as the observation error covariance in the Kalman Filter equations. Note that σ_{proc}^2 is the error variance of process model. Assuming that the model accurately represents the dynamics of the boundary, σ_{proc}^2 can be taken as a small quantity as compared to the observation noise variance σ^2 .

B. Estimating Dynamic Boundary

While spatial estimation for an estimation point may be performed by multiple cluster heads, the temporal estimation has to be associated with a specific sensor and its observations. By default, the sensor with x location same as x_p is selected for performing temporal estimation. Assuming that the boundary has similar temporal variations within the *h-neighborhood* of an estimation point, any sensor having observations within the neighborhood of x_p can perform the temporal estimation for x_p . 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.

Estimates from a few selected sensor nodes are sent to the base station where a confidence band of width δ is obtained around the entire boundary. As the boundary is dynamic, estimates at different estimation points must be updated. Estimate at x_p is updated only if it is expected to have changed beyond a certain threshold since the last update. This *update threshold* is specified by $c \times \delta$ where $c (\leq 1)$ is a constant. Whenever any latest estimate reaches the base station, the confidence band is updated to reflect the changes. Therefore, if the boundary at a section is changing

faster than other sections, then that section is updated more frequently.

C. Comparison of Spatial and Temporal Estimations

So far, we have described the spatial estimation component and temporal estimation components of DBTR. These techniques are further analyzed and different options for estimating a dynamic boundary are explored.

In the temporal technique, estimates of k estimation points are sent to the base station. For k estimation points, communication overhead for this technique is of the order of $k \times l$ where l is the average number of hops of a node from the base station. Alternatively, communication overheads of the spatial estimation technique include messages required for aggregating observations from neighboring sensors and sending partial aggregates to the base station. Thus, communication overheads are proportional to the total number of nodes sending observations to the respective cluster heads. As discussed in Section IV, the total number of nodes sending observations is more than the number of estimations points. Consequently, the spatial estimation technique incurs additional overheads due to gathering of neighboring sensor observations.

The spatial estimation technique can be used in two ways for tracking a *dynamic* boundary:

- 1) Periodic Update: if spatial estimation is performed periodically to update boundary estimates, it is referred to as the *Spatial -Periodic* scheme.
- 2) Aperiodic Update - if the spatial estimation is performed whenever based on the changes in the boundary, it is referred to as the *Spatial -Aperiodic* scheme.

These two versions of the spatial estimation are used to compare the accuracy of the temporal estimation technique. Further, we assume that the spatial-aperiodic scheme updates its estimates whenever the temporal estimation technique updates its estimates. The following experiments are performed to compare the accuracy of two techniques.

1) *Effect of width of the confidence band*: Accuracy of estimation is affected by mainly two factors — dynamics of the boundary and δ . In this experiment, these two factors are varied and *LOC* of the spatial and temporal estimation techniques are observed. Figure 6(a) plots the accuracy of both aperiodic and periodic versions of these techniques when δ is varied. As δ is increased, lower *LOC* is observed for both the techniques. Higher values of δ increase the probability of coverage of the confidence band. However, for $\delta = 1.0$, *LOC* of spatial estimation is lower than temporal estimation by 2%, whereas *LOC* are similar for $\delta = 1.1$ and $\delta = 1.2$. Due to the aggregation of observations from multiple sensors during spatial estimation, the variance of the estimated distance to the boundary is low and it offers lower *LOC* than temporal estimation.

For $\delta > 1.2$, temporal estimation performs better than spatial estimation. The temporal estimation technique tracks $0.5 \times \delta$ change in the boundary and with higher values of δ , this change happens slower. This also means that the corresponding sensor node observes and estimates the distance to the boundary for longer duration before the update takes place. Hence, higher δ results in higher accuracy of estimation for the temporal technique.

The accuracy of aperiodic update schemes are also compared with the periodic versions of the temporal and spatial estimation

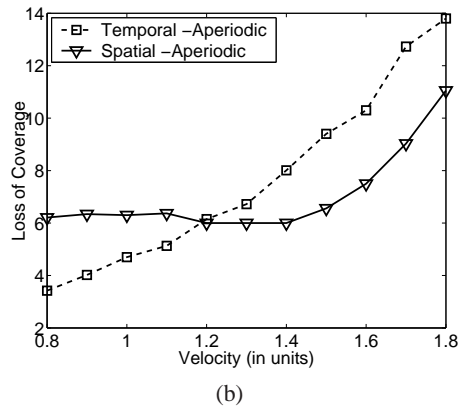
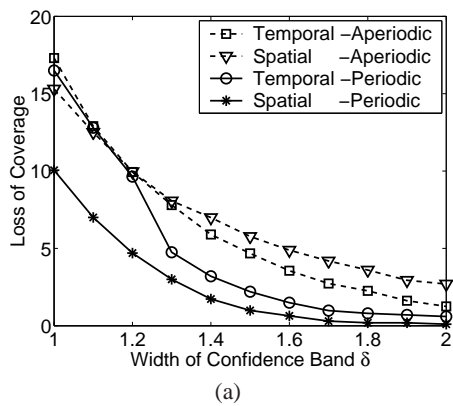


Fig. 6. Accuracy of Temporal and Spatial Estimation techniques (a) as δ Varies (b) as Velocity Varies

techniques. It is assumed that the constant period for update is the same as the sensor sampling period which is 1 second by default. It can be observed that for $\delta = 1.0$, LOC of the spatial-periodic update scheme is 10% whereas it is 15.3% and 17.3% for the spatial-aperiodic and temporal-aperiodic schemes respectively. Similar improvement in LOC can be seen also for all values of δ . The reason for better performance can be explained using the number of updates of the boundary estimates. For periodic updates, the number of updates for the boundary at an estimation point is 300 in 300 seconds. But for aperiodic schemes, the update happens only after a change in the boundary of $0.5 \times \delta$ is detected. In aperiodic schemes, the number of updates is 269 and 214 for $\delta = 1.0$ and 2.0 respectively. Due to higher number of updates, the periodic update schemes provide better accuracy.

2) *Effect of velocity*: Figure 6(b) plots the accuracy of the temporal and spatial estimation techniques when the velocity of the boundary is varied between 1 unit/second to 2 units/second. At higher velocity, the boundary moves out of the estimated confidence band faster and this reduces the coverage of the estimated confidence band. Hence, we observe that LOC of both the techniques increase with increase in velocity.

LOC of temporal estimation is lower than spatial estimation until the velocity is 1.2. For higher velocity, the accuracy of the temporal technique suffers due to faster change in the boundary. For velocity = 1.4 units/sec, the amount of change within one sampling period (1 second) is close to $\delta = 1.5$. Since the temporal technique tracks $0.5 \times \delta$ change, the accuracy of prediction starts to degrade.

From these experiments, we conclude that *spatial estimation provides better accuracy than temporal estimation except for higher values of δ or slow moving boundaries*.

VI. DBTR: COMBINING SPATIAL AND TEMPORAL ESTIMATIONS

The advantage of the spatial estimation technique is that it can estimate the boundary with high accuracy by aggregating multiple sensor observations. But this technique incurs high communication overheads due to gathering of latest observations. On the other hand, the temporal estimation can estimate a boundary with less communication overheads. It offers a technique for refreshing estimates aperiodically depending on the dynamics of the boundary. Considering merits and demerits of two techniques, we hypothesize that an estimation strategy can utilize benefits of both the techniques. While the spatial estimation incurs high

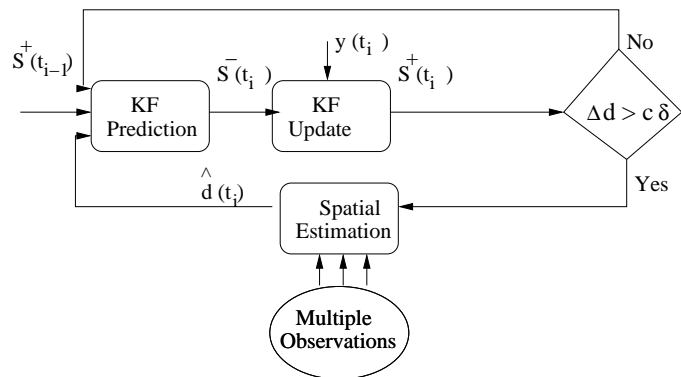


Fig. 7. Details of combining the Spatial and Temporal Estimation Techniques (x_p is omitted from all terms for clarity).

communication overheads, it should be performed less frequently. But in spatial estimation, cluster heads do not track dynamics of the boundary. Naturally, to decide when to update the estimates of a dynamic boundary, the temporal technique should be used. To utilize benefits of both of the techniques, DBTR combines both the spatial estimation as well as the Kalman Filter-based temporal estimation. The combined technique uses the spatial estimation technique for obtaining latest estimates and uses the temporal estimation technique to aperiodically refresh the estimates based on boundary dynamics.

A block diagram in Figure 7 shows all the components of DBTR for estimating the boundary at an estimation point.. The Kalman Filter consists of two stages: state *prediction* and state *update*. We use superscripts of ‘-’ and ‘+’ to indicate whether the state variable $s(x_p, t_i)$ corresponds to prediction stage or update stage respectively. In the figure, states corresponding to two consecutive sampling instants t_{i-1} and t_i are shown. Let $s^+(x_p, t_{i-1})$ denote the estimated state at t_{i-1} . From this state, the prediction stage predicts the state $s^-(x_p, t_i)$ at time t_i . This predicted state and the latest observation, $y(x_p, t_i)$ are used to obtain the updated output $s^+(x_p, t_i)$. Both the prediction and update stages together maintain the model of estimated distance.

Remaining part of the block diagram deals with how the state $s^+(x_p, t_i)$ is used to decide whether to trigger the spatial estimation. The temporal estimation remembers the last estimated distance to the boundary from the spatial estimation technique and compares with its own estimate of the current distance. Using

Kalman Filter observation Equation (13), distance to the boundary at x_p is $H \times s^+(x_p, t_i)$. Let $\hat{d}(x_p, t_{last})$, be the last updated estimate obtained using the spatial technique. The difference between the current estimated distance and $\hat{d}(x_p, t_{last})$ provides the change in the estimated distance since the last update by the spatial estimation technique. The difference Δd is estimated as:

$$\Delta d = H \times s^+(x_p, t_i) - \hat{d}(x_p, t_{last}) \quad (14)$$

This difference is compared with the *update threshold* which is $c \times \delta$ (where c is a constant and δ is the user specified width of the confidence band). If the difference is more than the update threshold, it implies that distance to the boundary at x_p has changed by larger than $c \times \delta$. As a result, the boundary is updated with the latest observations from all sensors in the *h-neighborhood* of x_p .

The estimate from the spatial technique is taken as the latest best estimate of the distance to the boundary at x_p and is used by future temporal estimations for more accurate prediction. In Figure 7, $\hat{d}(x_p, t_i)$, the output from spatial estimation is used to update the distance information in state $s^+(x_p, t_i)$. Assuming the delay in spatial estimation to be negligible, $\hat{d}(x_p, t_i)$ can update the Kalman Filter state in the same time instant t_i . *The intelligent combination of spatial and temporal estimation techniques not only reduces wasted boundary updates but also avoids updates to sections of the boundary that have not moved significantly.*

VII. 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 compare the performance of both the spatial and temporal estimation techniques, (ii) to test sensitivity of these techniques to parameters such as specified width of band, number of estimation points and dynamics of the boundary, (iii) to verify the practical applicability of spatial estimation using a testbed.

A. Experimental Setup

We implement a discrete-event simulator to evaluate DBTR in MATLAB. Sensors are randomly deployed in a two-dimensional field with dimension 100 units \times 50 units. The communication range of each sensor is 10 – 15 units. The base station is located in the middle of sensor field. The maximum number of hops from the base station to sensors in a multi-hop network varies between 7 and 12 hops. We assume each sensor message contains a single observation and a single partial aggregate (explained in Section III-C). Transmission of messages is assumed to be error-free. We also use a TOSSIM [18] based simulator to generate randomly distributed networks of different sizes and the underlying communication framework of the simulator is used to generate clusters in the network. These networks with clusters are used as input to DBTR implementation in MATLAB.

Performance of DBTR is evaluated with several boundaries generated using mathematical functions and real data traces from sensors. Boundaries with different spatial variations are used. For example, boundaries in Figure 8(a) and 8(b) are smooth (continuous higher derivatives) while those in 8(c) and 8(d) are non-smooth. In addition, we also use a boundary obtained based on a real oil-slick³ (Figure 8(e)). Dynamic boundaries are

| Parameter | Description | Default Value |
|------------|-----------------------------|---------------|
| n | Number of sensors | 100 |
| σ^2 | Observation Error Variance | 1.0 |
| δ | Width of band | 1.5 |
| h | <i>h-neighborhood</i> | 3.6 |
| k | Number of estimation points | 16 |
| t_s | Sampling Period | 1 sec |

TABLE I
PARAMETERS USED IN EXPERIMENTS

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.

Important experimental parameters and their nominal values are specified in Table I. These default values and the boundary Smooth 1, are used in the experiments unless specified otherwise.

B. Evaluation Metrics

DBTR is evaluated using two metrics: (i) *accuracy of estimated boundary* and (ii) *communication overheads*. Accuracy of the estimated boundary is measured in terms of *LOC* that is defined in Section I-A. At a time instant, there is a loss of coverage for an estimation point if the actual boundary is outside the estimated confidence band at that point. *LOC* is calculated as the total length of such time instances normalized by the total length of observations. The reported *LOC* for the boundary is estimated as the mean *LOC* for all estimation points and is expressed as percentage. The overall communication overhead is the cumulative number of transmissions required for the spatial estimation technique and updates needed by the temporal estimation technique. This reflects the energy expenditure of our solution.

C. Effect of Update Threshold

The temporal estimation technique updates the estimate based on its changes at an estimation point. Whenever the boundary at an estimation point changes by $c \times \delta$ where δ is the width of confidence band, it updates the estimate (refer to Figure 7). Here, experimentally we derive a suitable value for update threshold c . Figure 9 (a) depicts the *LOC* of the temporal estimation technique for different values of constant c . With $c = 0.2$, the boundary is updated if the temporal technique predicts that it has changed by $0.2 \times \delta$ whereas with $c = 1.0$, the estimates are updated only after the boundary has changed by δ .

We observe that *LOC* increases with increase in value of c but it increases slowly until $c = 0.5$ with *LOC* being less than 5%. For $c > 0.5$, *LOC* increases rapidly in a non-linear fashion. For $c = 0.6-1.0$, *LOC* increases from 6.01% to 14.9% with the temporal estimation technique. Though actual *LOC* depends on the dynamics of the boundary and value of δ , we have noticed that for $c > 0.5$, the rate of increase in *LOC* is high irrespective of the initial *LOC* at $c = 0.5$. The later the boundary estimates are updated, the higher is the *LOC*. This also happens because, due to inaccuracy in temporal estimation, the boundary changes by δ before the same is detected by the Kalman-filter. Hence, we conclude that good choice of c is 0.5 and *the boundary estimates should be updated at the base station whenever the boundary is expected to have changed by $0.5 \times \delta$ since the last update.*

³Data for Lake Maracaibo <http://modis.marine.usf.edu/index.html>

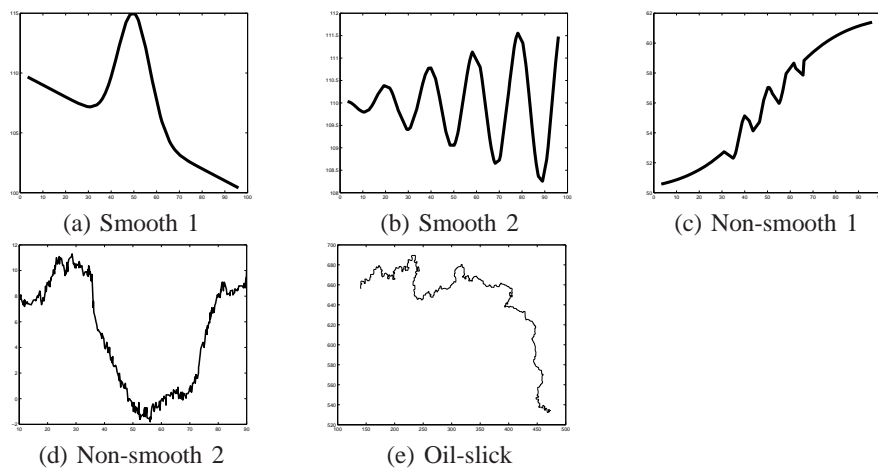


Fig. 8. Boundaries used to evaluate DBTR.

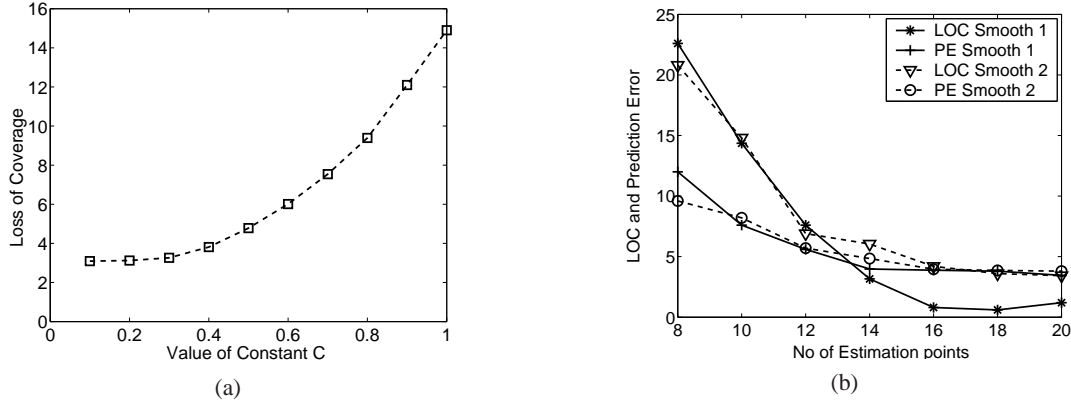


Fig. 9. (a) Effect of Update Threshold on Accuracy of Estimation (b) *LOC* and Prediction Error for Different Number of Estimation Points

D. Effect of the Number of Estimation Points on Accuracy

This experiment verifies that the *prediction error* and *LOC* (Refer to Section IV for relevant discussion) follow similar trend. Figure 9(b) shows how these two quantities change as the number of estimation points is varied. We observe that with increase in the number of estimation points, the prediction error function and *LOC* both decrease and finally stabilize when the number of estimation points is greater than 14. Initially both *LOC* and prediction error decrease sharply (50 – 70%) as the number of estimation points increases from 8 to 12. But, for the number of estimation points > 14 , both of these decrease by a small amount (0.5 – 0.2)%. The number of estimation points, for which prediction error stabilizes can be used as a good choice for k , the minimized number of estimation points. Since prediction error stabilizes earlier, a few more boundary points can be added in order to further reduce the *LOC*. Thus, our technique estimates a boundary using only minimal number of estimation points and ensures that the estimated band also achieves low *LOC*.

E. Accuracy of DBTR and the Spatial, Temporal Estimation Components

In this section, the performance of DBTR is compared with both its components—the temporal and spatial estimation techniques. In all of these techniques, an estimate is updated whenever a boundary point moves beyond $0.5 \times \delta$.

Figure 10(a) plots the *LOC* as the velocity of the boundary changes between 0.8 unit/second to 1.8 units/second. It shows that DBTR performs better than the temporal and spatial estimation techniques. For a velocity = 1 unit/second, *LOC* of DBTR is 3.42% whereas the spatial and temporal techniques achieve *LOC* of 6.3% and 4.7%.

It was observed earlier in Section VII-G that for a velocity less than 1.2 units/second, the temporal estimation provides better accuracy than spatial estimation. However, DBTR outperforms the temporal estimation technique. Accuracy of estimation depends on how often the boundary estimates are updated. In DBTR, whenever the boundary changes by more than $0.5 \times \delta$, the output of spatial estimation is used to update the current state in temporal estimation. Hence, the Kalman Filter predicts the future change in the boundary more accurately. Due to the feedback from spatial estimation, in DBTR boundary estimates are updated more frequently than temporal estimation.

F. Communication Overheads of Spatial Estimation Technique

The goal of this experiment is to find out how the communication overhead of the spatial estimation component of DBTR, varies with network size. We also compare the total messages required by this technique with a solution where all the observations are sent to a central server for estimation of the confidence band. As observed in Figure 11(a), when network size changes from 100 to 250, the communication overhead of

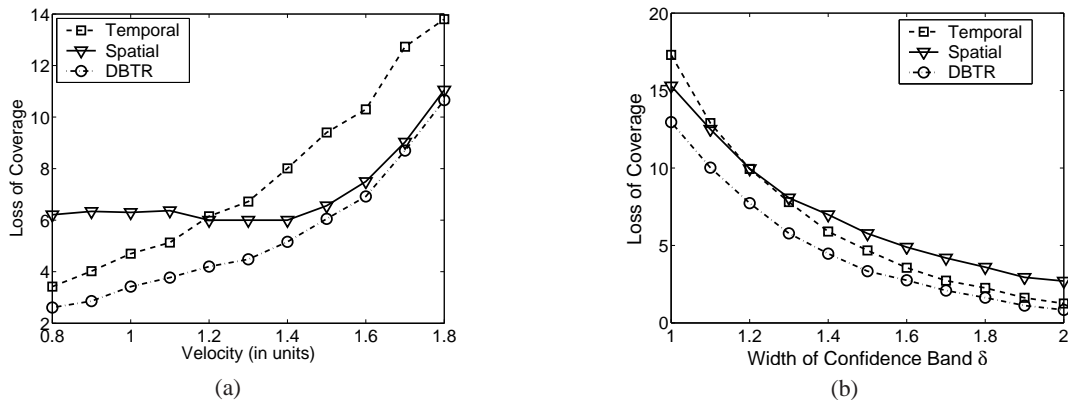


Fig. 10. (a) Accuracy of Temporal and Spatial Estimation techniques for Periodic Updates (b) Accuracy of DBTR compared with individual Spatial and Temporal Estimation Techniques

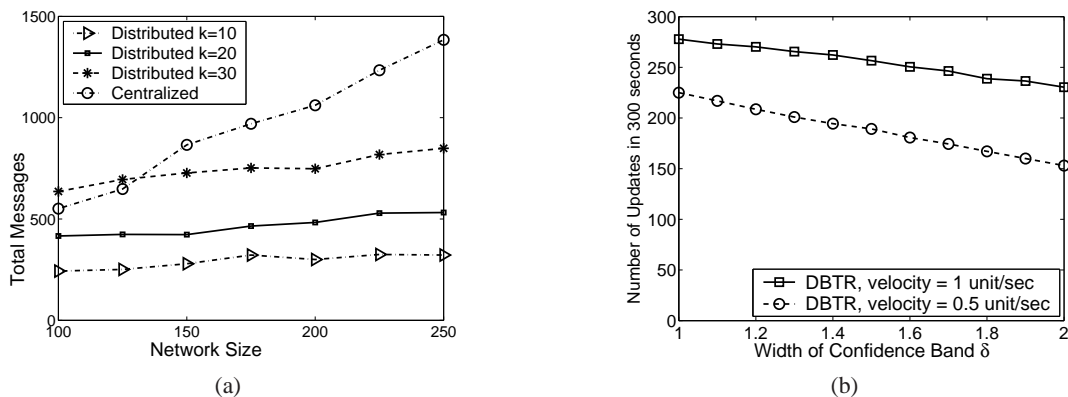


Fig. 11. (a) Communication overheads of DBTR compared with Centralized approach (b) Effect of Boundary Dynamics on communication overheads

the centralized scheme increases by a factor of 2.5 but in case of DBTR, it increases by 1.27 with 20 estimation points. The communication overheads for the spatial estimation depend on the h -neighborhood and as the optimal h is inversely proportional to the fifth root of n [13], with increase in network size, the value of h reduces. Thus, the distributed solution of DBTR is easily scalable to larger networks.

DBTR's communication overhead is lower than the centralized approach by a factor of 2.2 for 200 nodes. In addition, the proportional increase in communication overheads for larger networks is less. This shows that in-network aggregation is helpful in reducing communication overheads. However, as k increases from 10 to 30, communication overheads of DBTR increase from 243 to 636, by a factor of 2.6. The partial aggregates of each of the boundary points are estimated and routed to the Base Station separately, leading to an increase in communication overheads with k . The results motivates the need for reducing the number of estimation points to reduce communication overheads.

G. Effect of Boundary Dynamics on Communication Overheads

In this experiment, we observe how the communication overhead of DBTR varies with different velocities of the boundary. The communication overhead is captured using the number of times boundary estimates are updated. It depends mainly on two factors: (i) the width of the estimated band and (ii) velocity of the boundary along the y axis. We use two constant mean velocity of 1 unit/sec and 0.5 unit/sec. All the boundary points are assumed to be changing at the same velocity. Figure 11(b) depicts the

number of updates required for two different velocities in an interval of 300 seconds for different values of δ . We observe that as δ increases from 1.0 to 2.0, the number of updates reduces by 10% for velocity = 1 unit/sec and 30% for velocity = 0.5 unit/sec. With higher δ , the estimates are updated less frequently. A faster moving boundary requires more updates since the boundary moves out of confidence band more frequently. As velocity changes from 0.5 to 1 unit/sec, the number of updates increases from 225 to 278 for $\delta = 1.0$.

We also conducted experiments with different sections of the boundary changing at different velocities and observed that the communication overheads for updating of estimates increase in proportion to the velocity of the boundary and importantly DBTR is able to capture boundary dynamics for adaptive updates. Due to lack of space, we do not report this experiment.

H. Effectiveness of Aperiodic Update Scheme

In this experiment we compare the performance of DBTR with a technique that periodically updates the boundary. The experiment is conducted with a boundary that is assumed to be having constant mean velocity of 1 unit/sec during the first 150 seconds. Then the velocity changes linearly from 1.0 unit/second to 1.5 unit/second along the $+ve$ x axis during the next 150 seconds. Thus, the velocity of individual sections of the boundary is varied over time.

The solid line in Figure 12(a) plots *LOC* of periodic update scheme with update period varying between 0.75 seconds to 1.75 seconds. We observe that to achieve *LOC* < 5%, update schemes

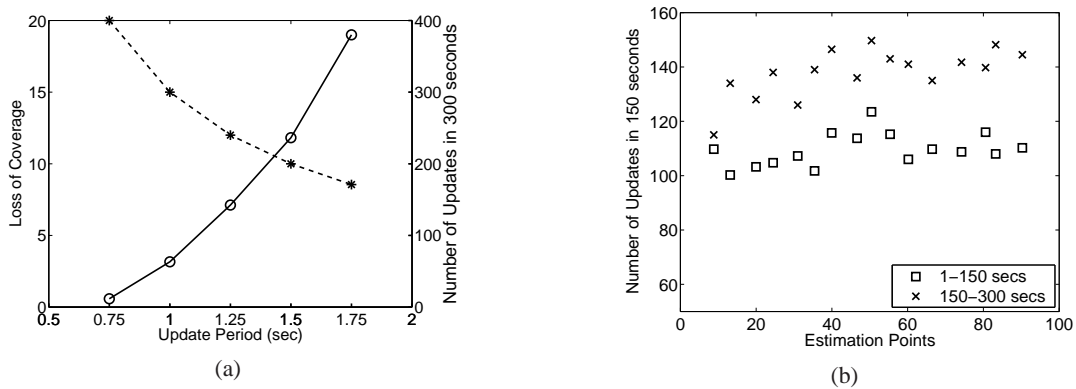


Fig. 12. (a) Performance of Periodic Update Schemes with Different Periods of Update. (b) Number of Updates using DBTR at 16 Estimation Points.

with period 0.75 seconds or 1 second can be used. Further, the dotted line in Figure 12(a) also plots communication overheads of periodic update scheme expressed in terms of the number of updates in 300 seconds. This shows that the number of updates for an estimation point reduces by 20% as the update period increases from 1 second to 1.25 seconds. Thus, to reduce communication overheads, higher update period is preferred.

Optimal periodic scheme is the periodic scheme that uses the longest time period for updating the boundary but ensures low $LOC (< 5\%)$. The optimality is decided based on the number of periodic updates. In this case, the optimal scheme is the one using 1 second as the period of updates. The optimal scheme achieves LOC of 3.16%. LOC increases to 7.16% if the update period increases to 1.25 seconds.

Next, we show the number of updates using DBTR for 16 estimation points in Figure 12(b). As the velocity changes after 150 seconds, we show the number of updates between 0–150 seconds and 150–300 seconds. In this case, LOC of DBTR is 4.45% and the average number of updates per estimation point is 247 in 300 seconds. DBTR updates the boundary based on the velocity and the specified δ , width of confidence band. During the first 150 seconds, the velocity is 1 unit/sec and we observe that DBTR requires 109 updates on an average whereas a periodic update scheme with 1 second period would use 150 updates. In the next 150 seconds, the mean number of updates in DBTR is 138.

The results of the experiment demonstrate that DBTR requires less communication overheads than an optimal periodic update scheme while guaranteeing a low LOC . It should be noted that DBTR achieves this performance improvement in spite of having no prior knowledge about the dynamics of the boundary.

I. Boundary Estimation using Infrared Sensors

To verify practical applicability of the spatial estimation technique of DBTR, we perform an experiment with real sensors. We use a robot equipped with linear infrared distance measuring sensors. Further, a non-linear boundary is created using a wall-like obstacle. The robot moves along a defined path and distances to the target are measured. The actual distances to the boundary are also noted for reference. Using the observations, the boundary is estimated. It is observed that for $\delta = 1.2$ and with the total number of observations less than 50, the confidence band provides very high LOC (35%). But LOC improves to 6% when the number of observations is increased to more than 100. In both cases,

observations are taken over the same range of x values. Figure 13(a) shows the estimated observation variance for different x values. Note that the variance is 1.0–2.0 for most of the locations. Further, the variance is maximum when the boundary exhibits large spatial variation near $x = 50$. In Figure 13(b), the estimated confidence bands for $\delta = 1.2$ and 2.0 are shown. LOC for $\delta = 1.2$ and 2.0 are 6% and 2% respectively. We observe that the actual boundary is covered by the band if δ is similar to the mean observation error variance. Thus, the practical experiment shows the efficacy of the spatial technique in estimating a boundary.

J. Impact of Sensing Angle on Accuracy of Estimation

In this experiment, we relax the assumption made in Section III-A that all the sensors align their sensing antennas along y axis to detect points on the boundary. The goal is to determine the impact of non-zero sensing angles on estimation accuracy. For this experiment, only the spatial estimation technique of DBTR is considered. Further, sensing angles are chosen from a normal distribution with mean zero. θ is used to denote the angle between sensing direction and the y axis. Figure 14(a) shows the impact of the variance of angle θ on LOC for two smooth boundaries when $\delta = 1.0$. We observe that LOC is about 2% for $\theta = 0$. But as variance of θ increases to 12 degrees, LOC increases by a factor of 8 for Smooth 1 and by a factor of 2 for Smooth 2. We observe that there are two reasons for higher LOC (i) due to non-zero sensing angles, there is an error component in the x coordinates of observations. This implies in Equation (2), the error α_i not only affects y_i but also x_i . In fact x_i is having error component $\alpha_i \times \sin(\theta)$. Due to error in x values, there is an increase in variance $\hat{\sigma}^2$. This causes higher LOC . (ii) Due to different sensing angles, the distribution of observations along the boundary is affected. Whenever the number of observations in h -neighborhood of an estimation point reduces, a higher LOC is expected for that estimation point. Because of reduction in variance (mentioned earlier), we expect lower LOC for more number of observations at an estimation point.

We verify this from the scatter plot in Figure 14(b) which shows LOC for a number of estimation points. The x axis gives the number of observations in h -neighborhood of estimation points. While selecting these points, sections of the boundary with high spatial variations are avoided so that point-wise LOC is not high due to high variance and it shows the effect of number of observations in h -neighborhood. We conclude that non-zero sensing angles do not affect LOC if all sections of a boundary have the required number of observations.

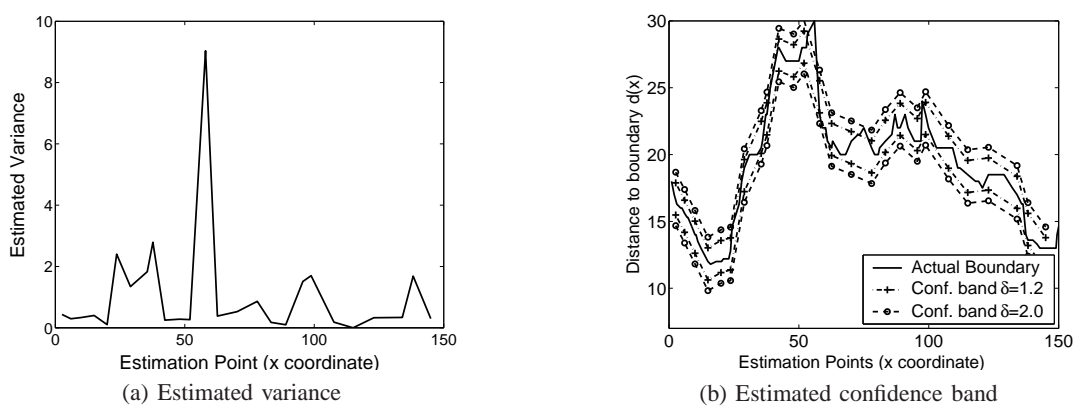


Fig. 13. Boundary estimation using observations from a testbed.

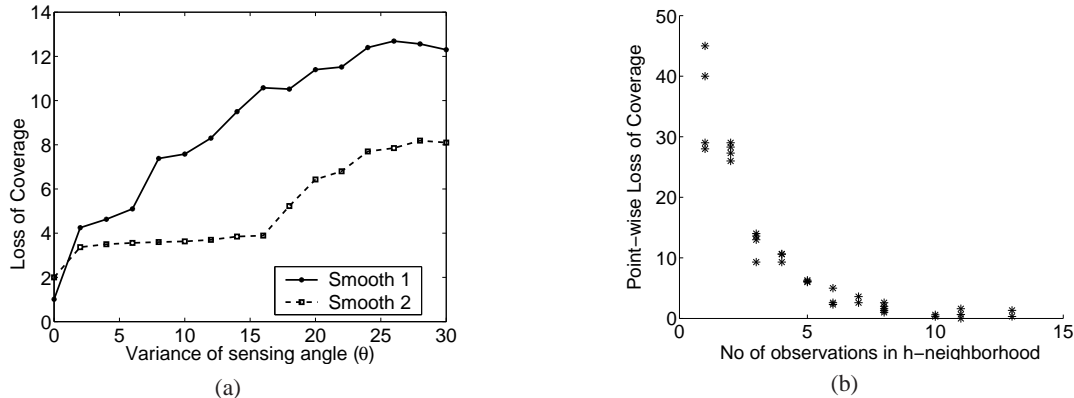


Fig. 14. (a) Effect of variance of sensing angle on accuracy (b) Effect of number of observations in h -neighborhood on accuracy.

K. Summary of Results

Experimental evaluation of the spatial and temporal estimation techniques reveals the following: (i) DBTR consistently estimates boundaries more accurately than the Spatial-only and Temporal-only estimation techniques. The accuracy improvement is by 2–4%. (ii) The distributed in-network estimation strategy reduces the communication overhead as compared to the centralized solution by a factor about 1.3 to 2.6. (iii) The Kalman Filter-based scheme can capture boundary dynamics with no prior knowledge of the boundary. When velocity of a boundary point changes with time, DBTR can adjust to it, whereas a periodic update scheme cannot. (iv) The heuristic for simultaneously minimizing LOC and the number of estimation points achieves a LOC of less than 2% for smooth boundaries.

Additionally, our experiments reveal the following which are not anticipated: (i) Whereas as expected, LOC of a confidence band decreases with δ , the number of estimation points minimizing LOC also reduces by a factor of 3 as δ changes from 1 to 1.5 and more. (ii) On an average a k value between 12 and 20 (where total number of sensors $n = 100$) gives a low LOC . This implies with point-wise spatial estimation, it is possible to obtain entire boundary front even when it is estimated at a few selected locations. (iii) The spatial estimation technique using non-parametric regression can estimate a boundary even if all sensors do not align their antennas in the same direction and for non-smooth boundaries. This itself is a new extension to non-parametric regression-based applications.

VIII. 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 [19] 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 coefficients of basis functions are computed using observations to obtain an estimation with minimized mean square error (MSE). Using correlations among sensor observations for achieving energy-accuracy trade-off is also explored in [20]. The system BBQ [21], exploits correlation among sensor attributes and a probabilistic model to answer queries.

An alternative of Kalman Filter-based approach for tracking discrete points on the boundary is particle filter [22] that can be used for non-linear dynamics or non-gaussian error. Switching Kalman Filters can be used to monitor boundaries with non-stationary dynamics (e.g., a storm) as discussed in [23].

The goals of DBTR are most similar to the boundary estimation problem solved by [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. DBTR provides a confidence band around the actual boundary using range sensor observations whereas in [2], the approximate boundary using in-situ measurements provides a lower bound on MSE. Several other techniques for event boundary detection using in-situ sensors are proposed in [24], [3]. While we assume range sensors always find a point on the boundary, deciding whether a

sensor is an edge sensor or not [25], [26] is an important aspect in event boundary detection using in-situ measurements.

DBTR's adaptive selection of estimation points is similar to the adaptive sampling method [27], where after a preview phase, additional observations are collected in regions containing the boundary. Similar approach of using higher network granularity in regions with high variability is proposed in [28].

DBTR makes use of sensors with range/remote capabilities for detecting a moving boundary. Another application using remote sensing is CASA [6], where a network of radars scan an angular sector up to 360 degrees for meteorological monitoring to detect tornadoes. Using their approach, we could detect multiple boundary points using one range sensor. Recently developed ultrawideband Radar motes [9] are another example of such sensors which can be used to detect moving objects like people and vehicles.

IX. CONCLUSIONS

We have proposed the DBTR algorithm for dynamic boundary estimation in sensor networks where observations from range sensors are aggregated and a confidence band around the actual boundary is obtained from estimates at a few selected locations. We demonstrated how to estimate a boundary with high accuracy using only minimal number of estimation points. The spatial estimation technique of DBTR uses non-parametric regression relationship to estimate distances to boundary and utilizes in-network aggregation of sensor observations to reduce communication overheads as compared to a centralized solution. Note that DBTR's input is a discrete set of observed points that lie approximately on the boundary. Irrespective of the actual sensing mechanisms used for gathering observations, using DBTR a confidence band can *always* be obtained. The temporal estimation technique of DBTR uses a Kalman filter-based prediction technique to track changes in the boundary. This strategy updates the estimates before the boundary is expected to move out of confidence band. Thus, DBTR tracks both the spatial and temporal variations of a boundary and provides a confidence band with high accuracy around the actual boundary at all times with low communication overheads without *a priori* knowledge about the dynamics of the boundary.

We are currently extending our work to take into account delays in message communications and account for non-Gaussian errors. As part of future work, we propose to study how DBTR can be applied to track a moving contour instead of only a boundary front. If a contour can be viewed as consisting of four fronts, one in each quadrant, then this may require deployment of range sensors in four patches. Other directions for extending our work are considering fronts following complex dynamic models such as constant acceleration model or analyzing the effect of different MAC protocols and network routing protocols on the accuracy of estimation.

REFERENCES

- [1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Elsevier Journal of Computer Networks*, vol. 38, pp. 393–422, 2002.
- [2] R. Nowak and U. Mitra, "Boundary Estimation in Sensor Networks: Theory and Methods," in *Proceedings of Information Processing on Sensor Networks (IPSN)*, April 2003.
- [3] M. Ding, D. Chen, K. Xing, and X. Cheng, "Localized Fault-Tolerant Event Boundary Detection in Sensor Networks," in *IEEE INFOCOM*, vol. 2, pp. 902–913, March 2005.

- [4] C. H. Hsieh, Z. Jin, B. Q. Nguyen, D. J. Tung, A. L. Bertozzi, and R. M. Murray, "Experimental validation of an algorithm for cooperative boundary tracking," in *Proceedings of American Control Conference*, 2005.
- [5] S. Srinivasan, K. Ramamritham, and P. Kulkarni, "ACE in the Hole: Adaptive Contour Estimation Using Collaborating Mobile Sensors," in *Proceedings of Information Processing on Sensor Networks (IPSN)*, April 2008.
- [6] M. Zink, D. Westbook, S. Abdallah, and B. Horling, "Meteorological command and control: An end-to-end architecture for a hazardous weather detection sensor network," in *Workshop on End-to-End, Sense-and-Respond Systems, Applications, and Services (EESR)*, 2005.
- [7] A. Utkin, A. Lavrov, L. Costa, F. Simoes, and R. Vilar, "Detection of Small Forest Fires by Lidar," *Applied Physics B: Lasers and Optics*, vol. 74, pp. 77–83, 2002.
- [8] A. Lavrov and R. Vilar, "Application of lidar at 1.54 μm for forest fire detection," *Remote sensing for earth science, ocean, and sea ice applications*, vol. 3868, pp. 473–477, 1999.
- [9] P. Dutta, A. Arora, and S. Bibyk, "Towards Radar-Enabled Sensor Networks," in *Proceedings of Information Processing on Sensor Networks (IPSN)*, April 2006.
- [10] B. C. Donovan, D. McLaughlin, and J. Kurose, "Principles and Design Considerations for Short-Range Energy Balanced Radar Networks," in *Proceedings of IEEE Geoscience and Remote Sensing Symposium*, 2005.
- [11] W. J. Lentz, "The Visioceilometer: A Portable Visibility and Cloud Ceiling Height Lidar," in *Oceanography and Atmospheric Science, Storming Media, Pentagon Report*, Jan 1982.
- [12] R. Wainner, B. Green, and M. Allen, "Handheld, battery-powered near-IR TDL sensor for stand-off detection of gas and vapor plumes," *Applied Physics B: Laser and Optics*, vol. 75, pp. 249–254, 2002.
- [13] W. Härdle, *Applied Nonparametric Regression*. Cambridge University Press, 1990.
- [14] E. A. Nadaraya, "On estimating regression," *Theory Prob. Appl.*, vol. 10, pp. 186–90, 1964.
- [15] B. Deb, S. Bhatnagar, and B. Nath, "A topology discovery algorithm for sensor networks with applications to network management," in *IEEE CAS Workshop on Wireless Communication and Networking*, Sept 2002.
- [16] A. Meka and A. Singh, "DIST: A Distributed Spatio-temporal Index Structure for Sensor Networks," in *Proceedings of ACM Conf. on Information and Knowledge Management*, November 2005.
- [17] K. M. Moudgalya, *Digital Control*. John Wiley & Sons Ltd., Chichester, 2007.
- [18] P. Levis, N. Lee, A. Woo, M. Welsh, and D. Culler, "TOSSIM: Accurate and scalable simulation of entire tinyos applications," in *Proceedings of ACM Conference on Embedded Networked Sensor Systems (Sensys)*, Nov 2003.
- [19] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed Regression: an Efficient Framework for Modeling Sensor Network Data," in *Proceedings of Information Processing on Sensor Networks (IPSN)*, April 2004.
- [20] M. Sharaf, J. Beaver, A. Labrinidis, and P. Chrysanthis, "Balancing energy efficiency and quality of aggregate data in sensor networks," *The VLDB Journal*, vol. 13, no. 4, Dec 2004.
- [21] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong, "Model-based Approximate Querying in Sensor Networks," *The VLDB Journal*, vol. 14, no. 4, pp. 417–43, 2005.
- [22] M. Coates, "Distributed Particle Filters for Sensor Networks," in *Proceedings of Information Processing on Sensor Networks (IPSN)*, April 2004.
- [23] V. Manfredi, S. Mahadevan, and J. Kurose, "Switching Kalman Filters for Prediction and Tracking in an Adaptive Meteorological Sensing Network," in *IEEE Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, 2005.
- [24] K. Ren, K. Zeng, and W. Lou, "Fault-tolerant Event Boundary Detection in Wireless Sensor Networks," in *Proceedings of IEEE GLOBECOM*, 2006.
- [25] K. Chintalapudi and R. Govindan, "Localized edge detection in sensor fields," in *IEEE Sensor Network Protocols and Applications Workshop*, pp. 59–70, May 2003.
- [26] P. Liao, M. Chang, and C. J. Kuo, "Distributed edge detection with composite hypothesis test in wireless sensor networks," in *IEEE Globecom*, 2004.
- [27] A. Singh, R. Nowak, and P. Ramanathan, "Active Learning for Adaptive Mobile Sensing Networks," in *Proceedings of Information Processing on Sensor Networks (IPSN)*, April 2006.

- [28] S. N. M. Duckham and M. Worboys, "Monitoring dynamic spatial fields using responsive geosensor networks ;" in *Proceedings of the 13th annual ACM international workshop on Geographic information systems*, 2005.