Localized Edge Detection in Sensor Fields

Krishna Kant Chintalapudi, Ramesh Govindan
University of Southern California, Los Angeles,
California, USA, 90007.

Abstract

A wireless sensor network for detecting large-scale phenomena (such as a contaminant flow or a seismic disturbance) may be called upon to provide a description of the *boundary* of the phenomenon (either a contour or some bounding box). In such cases, it may be necessary for each node to locally determine whether it lies at (or near) the *edge* of the phenomenon. In this paper, we show that such localized edge detection techniques are non-trivial to design in an arbitrarily deployed sensor network. We define the notion of an edge and develop performance metrics for evaluating localized edge detection algorithms. We propose three different approaches for localized edge detection and present one example scheme for each. In all our approaches, each sensor gathers information from its local neighborhood and determines whether or not it is an *edge sensor*. We evaluate the performance of each of the example schemes and compare them with respect to the developed metrics.

1 Introduction

Several physical phenomena (for instance, contaminant flows [3] and seismic disturbances) can span large geographic extents. Fine-grain sensing of these time-varying phenomena can help scientists understand what factors (*e.g.*, soil density variations) affect the spread of these phenomena. One way to architect an energy-efficient sensor network for studying these phenomena is to store the detections of the phenomena within the network and provide a query interface which enables scientists to understand the temporal and spatial properties of these phenomena.

We anticipate that one common query will ask for the spatial extent of the phenomenon at a given time: for example, "Which sensors saw the primary wave before time T?". For energy-efficiency reasons, it makes more sense for to design the spatial query that returns a *boundary* that captures all or most nodes that satisfy the query predicate. A geometric representation of the boundary has the potential to be more concise (and therefore more energy-efficient) than an enumeration of all nodes. Examples of such representations include contours, hulls, or bounding boxes.

An energy-efficient boundary finding algorithm will need to carefully choose nodes in the sensor network and compute the boundary "in-network". A key component of such an algorithm is a localized *edge detection* scheme: a technique by which each node locally determines (perhaps by gathering information from other nodes within its neighborhood) whether it lies on or near the boundary specified by the query. If a reliable technique existed for localized edge detection, then, conceptually at least, boundary finding is simply a matter of sequentially traversing all nodes that determine themselves to be on the *edge*. Localized edge detection will be an essential component of boundary *tracking* as well; as the phenomenon evolves with time, nodes at the edge may alert neighboring nodes (in a manner similar to target tracking [4]). We think of localized edge detection as a powerful primitive upon which a variety of applications might be built.

Edge detection has been widely studied in the context of digital image processing. Filtering [1] is one of the most common approaches to detecting edges in images. To determine whether an image pixel is at an edge or not, this approach applies a filter to values of a set of neighboring pixels. As such, these techniques can be directly applied to localized edge detection.

However, one fundamental difference between images and a sensor field is the spatial regularity of information. A digital image is a regular grid of pixels, and information is sampled at regular intervals. Almost all standard digital image processing techniques (Fourier transforms, high-pass filtering) rely on having information about the image at regular intervals. Deployment and maintenance of thousands of sensors in a grid like regular fashion over large geographical extents is clearly infeasible. It is expected that sensors will be arbitrarily placed in the sensor field and will be prone to failures or even node displacement. With this relaxation of regularity, it is less clear that digital image filtering can be applied to localized edge detection.

In fact irregular node placement makes it hard to even define precisely whether a node is at an edge or not. In Section 2, we show how to circumvent this difficulty and define some met-

rics for localized edge detection. Because the efficacy of edge detection based on digital image filtering is unclear, we consider two other classes of edge detection schemes in Section 3: a *statistical* scheme and a *classifier-based* scheme (that is suggested by the pattern recognition literature [2]). We find that, over a fairly broad range of operating conditions, the classifier scheme out-performs the other schemes. We present our evaluation results in Section 4, and conclude in Section 5.

To our knowledge, no prior work has considered localized edge detection in sensor networks. Concurrently, Nowak and Mitra [5] describe a scheme for estimating the boundary of a large-scale phenomenon by aggregating readings along a predefined hierarchical structure within the network. Their approach is somewhat complementary to ours, in that our localized edge detection is a primitive that might be used in a variety of boundary estimation applications (not just in their algorithm, but also in applications that estimate more concise, but approximate, boundary descriptions such as ellipses and hulls).

2 Edge Detection

In this section we describe our model of the sensor field and discuss definitions for an "edge". We then develop metrics to evaluate localized edge detection algorithms and discuss the trade-offs involved in the design of localized edge detection

2.1 Assumptions, Models and Terminology

In what follows, we make fairly general assumptions about the capabilities of sensor nodes and the structure of sensor networks. Sensor nodes can be arbitrarily deployed, but each such node knows its location, perhaps using a localization system [6]. For simplicity of exposition, we assume that the deployment of sensors is in the plane, and location can be specified by (x_s, y_s)). We use the term sensor field to both mean the geographical region covered by the deployment, and the set of sensors within the region. Sensors can make measurement errors, and our localized edge detection schemes will need to be robust to these. (Edge detection algorithms may also exhibit significant error due to errors in localization [7]; we do not model such errors, since our interest is in understanding how to compensate for sensor error in edge detection).

We model an edge as follows. Consider a phenomenon that spans some arbitrarily shaped sub-region of the sensor field. Each sensor can, based on locally collected measurements, determine whether it belongs to the sub-region covered by the phenomenon or not. We call the function that makes this decision the *event predicate*, and denote the event predicate at sensor s by η_s. Taking our example in Section 1, if the phenomenon of interest is "the geographical extent covered by the primary seismic disturbance at time *T"*, the event predicate for each sensor is, informally: "Did I see a primary seismic disturbance at or before *T*?".

Given an event predicate, we can then define the *interior* of a phenomenon (I) to be the spatial region $\Re^2$ such that, if a perfectly calibrated error free sensor were placed in this region its predicate function would have evaluated to 1. The *exterior O* of the phenomenon can be similarly defined. Based on these, there exists an *idealized* definition of the *edge* of a phenomenon E: the edge is the set of all points (x,y), such that every non-empty neighborhood of (x,y) intersects with both I and O. We call E the ideal edge, and E represents the ground truth that defines the boundary of the phenomenon.

This idealized definition is descriptive, but does not give us much insight for designing or evaluating localized edge detection schemes. The ideal edge has no "thickness" and therefore constitutes a very restrictive definition of an edge. Intuitively, we would like a sensor to consider itself to be an *edge sensor* if it is closer to the ideal edge than any other sensor. For this reason, we introduce the notion of *tolerance* of an edge detection scheme. We define a sensor to be an edge sensor if it a) is in the interior of the phenomenon, and b) lies within a prespecified distance r of the ideal edge. We call r the *tolerance radius*, and the area around a sensor node covered by a circle of radius r the *tolerance neighborhood*. The tolerance radius roughly measures the "thickness" of the edge that the designer of a localized edge detection scheme is willing to tolerate. For a given tolerance radius, we can define metrics that enable us to compare the efficacy of different edge detection schemes (Section 2.2).

A stronger definition an edge might be to require *continuity* among the set of edge sensors—that is, that there exists a path between every pair of edge sensors that only traverses edge sensors. We have not adopted this definition because it seemed to be beyond the realm of localized techniques to ensure this property. Of course, if we could ensure this property, it would be easy to define energy-efficient boundary finding algorithms that simply traversed edge nodes. As such, whatever boundary finding algorithms that we build on top of localized edge detection will need to traverse some non-edge sensors to determine a continuous boundary for a phenomenon. Such algorithms are beyond the scope of this paper.

Finally, detecting whether a node lies at the edge of a phenomenon is slightly different from detecting whether a node lies on (or near) a *contour* (or iso-lines; *i.e.* continuous curves across the sensor field defined by sensors having the same value of, say, temperature). In the general case, there isn't a well-defined notion of the interior and exterior of a contour, as much as there is a distinction between whether a sensor detects a phenomenon or not.

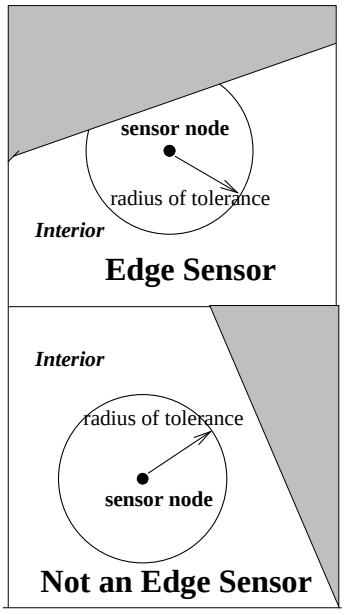

Figure 1: If the edge passes through the radius of tolerance it is deemed an edge sensor.

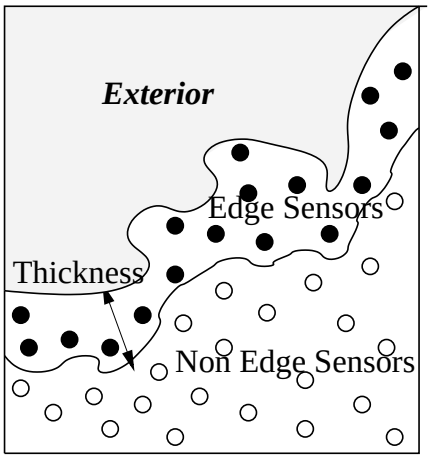

Figure 2: Having a tolerance radius gives a certain thickness to an edge.

2.2 Metrics

There are, broadly speaking, two classes of desirable characteristics of localized edge detection: robustness, and performance. These characteristics inform our choice of metrics for localized edge-detection.

There are several desirable robustness properties of an edge detection algorithm. First, as we shall show later, localized edge detection algorithms can intrinsically exhibit error, failing to detect an edge when there is one, or detecting an edge when there is none. A good algorithm has low intrinsic error. Second, localized edge detection algorithms must be relatively robust to reasonable levels of sensor calibration error. Finally, many localized edge detection schemes employ thresholds to decide on the existence of an edge. We would prefer schemes which are relatively insensitive to the threshold settings over a broad range of operating conditions.

In the performance category, an obvious consideration is energy expended in communication. There exists a trade-off between energy and accuracy in localized edge detection; intuitively, a node can get information from a bigger neighborhood to increase the likelihood of a positive detection. The second performance criterion is the quality of the result, defined by the actual thickness of the edge. Although our definition of an edge above includes a tolerance radius that nominally defines an edge thickness, an actual edge detection scheme might have a thickness that is larger or smaller than this radius.

Based on the above discussion, in this section we use the following metrics to evaluate the performance of localized edge detection algorithms.

Let S be the set of all sensors. Let E be the curve representing the edge (as defined in Section 2.1). Suppose set S_{true} be the set of sensors in the sensor field which are within a distance of r from E. Let S_{det} be the set of sensors marked as edge sensors by the algorithm and let N be the total number of sensor nodes.

Percentage Missed Detection Errors e_m : This represents the fraction of sensors which lie within the radius of tolerance (S_{true}) but were not marked as edge sensors (S_{det}).

$$e_m = \frac{|S_{true} - S_{det}|}{|S_{true}|}. \quad (1)$$

False Detection Errors e_f : This represents the fraction of nodes that declared themselves to be edge sensors but should not have ($S_{det} - S_{true}$) among the *rest* of the ($S - S_{true}$) sensors. For this reason, the denominator for e_f is different from that in (1).

$$e_f = \frac{|S_{det} - S_{true}|}{N - |S_{true}|}. \quad (2)$$

Mean thickness ratio e_t : Let $t(S,E)$ be the mean distance of all the sensors in set S to the edge E. We define,

$$e_t = \frac{t\left(S_{det},E\right) - t\left(S_{true}\right)}{t\left(S_{true},E\right)}. \quad (3)$$

To avoid the effect of random outliers, we consider only the closest 95% edge sensors in the mean.

We are now ready to discuss some localized edge detection schemes that illustrate the trade-offs involved in localized edge detection.

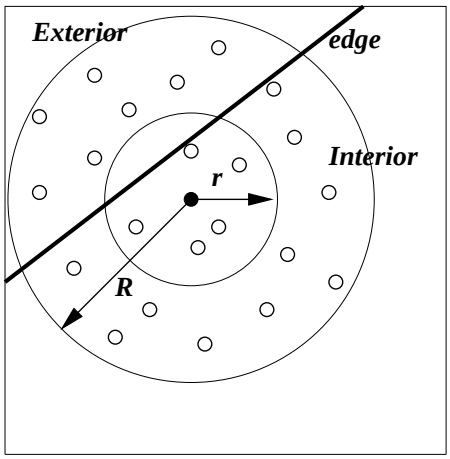

Figure 3: Performance can be improved by gathering information beyond the tolerance radius. Here, the sensor gathers information in a circle of radius R, the probing radius and is able to detect an edge which passes through the area of tolerance more reliably.

3 Three approaches to localized edge detection

In this section we propose three qualitatively different approaches to localized edge detection in a sensor field: a statistical approach, an approach drawn from image processing and an approach drawn from the pattern recognition literature. Each approach can be used to generate a family of algorithms for edge detection.

In all these approaches, each sensor gathers information from sensors in its neighborhood and independently tries to determine if an edge passes within its tolerance radius. Specifically, the sensor gathers the *location* and *the values of the event predicate* (that determines whether the sensor is in the interior or the exterior of a phenomenon) from each node within the neighborhood.

One parameter that determines the performance of all algorithms, to varying extents, is the *size* of this neighborhood. Arbitrary placement of the sensors coupled with sensor errors can result in detection errors. In general, the performance of a scheme improves as we collect information from more sensors (larger neighborhood). This is because the node gets more samples from the interior and the exterior of the phenomenon, and can make more confident estimates even in the presence of sensor errors. However, collecting more information incurs more communication overhead and hence increases the energy usage of the scheme. We have already mentioned this energy accuracy trade-off. We represent this parameter by a circle of radius R centered around the sensor and call it the **probing radius**. Typically, the probing radius is greater than the tolerance radius *i.e.* $R > r$ (see Figure 3). Generally the greater the $\frac{R}{r}$ ratio, the better the performance of the algorithms in terms of errors and thickness ratio, however the communication overhead increases roughly as R^2. In the rest of the paper we shall refer to this neighborhood as the **probing neighborhood**.

3.1 The statistical approach

A general statistical scheme would gather data from the sensors in the probing neighborhood and perform statistical analysis to decide whether or not the sensor is an edge sensor. The advantage in this approach is that statistical methods can be explicitly tailored to be robust to errors, if error characteristics are known. The general algorithm for a statistical scheme then needs three components to be specified.

1. The information to be collected from the neighbors.
2. A set of statistics $\Gamma_1, \Gamma_2, \cdots, \Gamma_n$ based on the information collected from the neighbors.
3. A boolean decision function $\Psi\left(\Gamma_1, \Gamma_2, \cdots, \Gamma_n\right)$ to decide if the sensor is an edge sensor. The decision function usually would involve comparing a value evaluated using $\{\Gamma_i\}_{i=1}^{i=n}$ against a *threshold* which maybe statically or dynamically assigned.

3.1.1 An example scheme

In this paper we evaluate a specific statistical scheme that we designed for edge detection. The key idea behind the scheme is the observation that if one collects the event predicate values

from sensors in the neighborhood, and these values form a bimodal distribution (spikes at 0 and 1) then an edge is present. Let n_+ be the number of 1 valued event predicates and n_- be the number of zero valued event predicates in the neighborhood. We calculate the following statistic:

$$\Gamma = 1 - \frac{|n_+ - n_-|}{n_+ + n_-}. \quad (4)$$

$$\Psi(\Gamma) = \begin{array}{lll} 1 & if & S \geq \gamma_0, \\ 0 & if & S < \gamma_0. \end{array} \quad (5)$$

Our statistical scheme is intuitively simple, and therefore forms a baseline for comparison against other schemes. One salient feature lacking in the statistical scheme is that it does not take the geographical locations of sensors into account when making its decisions. For arbitrarily placed sensors, as we shall see later, this can make a difference.

Designing the statistical scheme to be robust to sensor errors is a bit tricky, as we now explain. If the sensors were perfectly calibrated and error free, the presence of an edge would be indicated by a non-zero value of the statistic Γ and any $\gamma_0 > 0$ would suffice. In a more realistic scenario, with arbitrarily placed sensors having calibration and measurement errors, the statistic would yield non-zero values in absence of edges because of sensor errors. Also if $R > r$, for edges passing in the probing neighborhood which do not lie in the area of tolerance, (4) would give a non-zero value. Then, the choice of an "appropriate" threshold γ_0 would determine the performance of the scheme. In general the choice of $\gamma_0 \in (0,1)$ depends $\frac{R}{r}$, ρ and the performance requirements of the application.

3.1.2 Analysis of the scheme and choice of γ_0

To gain some intuition about the choice of γ_0 and how it relates to the tolerance radius and the probing neighborhood, we analyze the performance of the proposed statistical scheme. For our analysis we assume that nodes are placed in the region at locations drawn from a uniform density function with a density ρ sensors per unit area. Also we assume that the sensors make an error in evaluating the value of the event predicate with a probability p. We hope that this error model encapsulates both calibration and measurement errors. Further, we assume that the probing neighborhood is so "small" in comparison to the area covered by the entire phenomenon that the edge can be approximated by a straight line in this region.

As discussed in Section 2.2, errors can be either false detections e_f or missed detections e_m. False detections can arise in two ways. One cause of false detections is when there is no edge in the probing radius but the algorithm detects an edge due to sensor errors. The second occurs when there is an edge in the probing radius but not within the tolerance radius. We call the former kind of errors *pure false detections* (e_{pf}) and the latter *unwanted detections* (e_{ud}).

$$e_f = e_{pf} + e_{ud}. \quad (6)$$

We make this distinction because a high e_{pf} can result in a large number of sensors being deemed edge sensors even when they are in the "middle" of a phenomenon because of sensor errors and local variations in sensor density. On the other hand, a high e_{ud} simply increases the thickness of the edge. For certain applications, the edge thickness may not be as harmful as identifying a sensor as an edge sensor when it is far from an edge. For fixed values of $\frac{R}{r}$ and ρ, the errors e_m, e_{pf} and e_{ud} depend on the choice of γ_0.

Now, the number of sensors present in an area a can be modeled by a Poisson random variable,

$$P(N = n) = e^{-a\rho} \frac{(a\rho)^n}{n!} \quad (7)$$

and the number of sensors making an error M among N sensors can be modeled by a binomial random variable,

$$P(M = m | N = n) = \left(\begin{array}{c} n \\ k \end{array} \right) (p)^m (1-p)^{n-m}. \quad (8)$$

Based on these assumptions, one can numerically calculate the probability density function for Γ defined in (4), for given values of $\frac{R}{r}$ and ρ. The procedure for calculating the density function is described in Appendix A.

Choosing γ_0: An Example Figure 4, computed from our analysis, shows the variation of the percentage of true, unwanted and false detections as γ_0 varies from 0 to 1, for two different values of $\frac{R}{r}$ (1.0 and 1.5). The sensor density is such that, the expected number of sensors in area of tolerance is 15. The sensor error probability $p = 0.05$.

Suppose an application requires a true detection ratio of at least 80% and also wants false detections to be below 1%. Also suppose the application can do with slightly thicker edges and allows about 20% extra edge detections. For $\frac{R}{r} = 1$, corresponding to an error of 80%, the $S_0 \approx 1.5$. Corresponding to this value of S_0, the false error $e_{pf} \approx 30\%$ and $e_{ud} = 0$. The situation is considerably improved when $\frac{R}{r} = 1.5$. If we choose a threshold of 0.4, one can achieve 80% true detections, but now e_{pf} less than 1% and $e_{ud} \approx 3\%$. Hence, by increasing the probing radius we have improved the performance of our scheme. However, by increasing the probing area we incurred an increase in communication overhead by 125% ($\frac{R^2}{r^2} - 1$).

Clearly since performance depends on the choice of γ_0 and $\frac{R}{r}$, it becomes essential for sensors to be able to figure out a suitable setting for these parameters for "satisfactory" operation. It is conceivable that pre-calculated performance curves

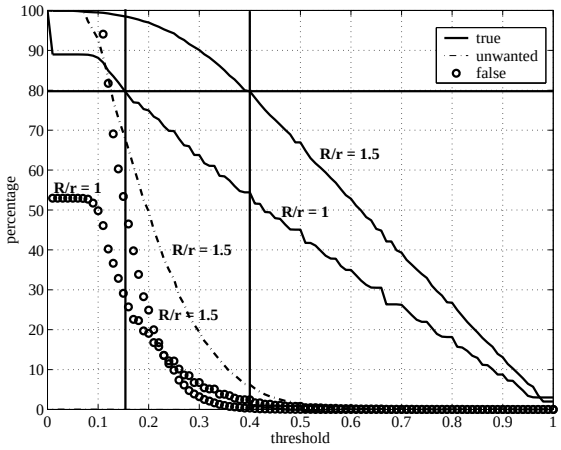

Figure 4: Variation of percentage true, unwanted and false detections as threshold S_0 varies from 0 to 1 for $\frac{R}{r}$ values of 1 and 1.5, an expected value of 15 sensors in the tolerance region and $p = 0.05$.

similar to those in Figure 4 are stored in the sensor nodes *a priori*. Nodes estimate the local sensor density and based on the performance criterion desired, use the performance curves to come up with an operating threshold.

3.2 The image processing approach

Numerous techniques for edge detection have been developed and analyzed in the image processing literature [1]. It is therefore tempting to attempt to apply such techniques to localized edge detection in sensor networks. In this paper we do not exhaustively examine all possible image processing techniques, but simply pick a framework that can incorporate a class of high pass filtering techniques (a standard way of performing edge detection in images *e.g.*. Prewitt, Sobel filters) for localized edge detection.

A high-pass filter retains only the high frequencies (abrupt changes such as edges) present in the image and removes all the uniformities. Designing a filter with a desired frequency response is a mature art and several different techniques exist. In general, if a filter with a frequency response $F(f_x, f_y)$ is desired then a filter $H(x,y)$ can be designed to approximately match the desired frequency response. Here, f_x and f_y represent the frequencies in the image in the x and y axes. To detect edges, the image $P(x,y)$ can be filtered by convolving with the filter $H(x,y)$. Within the context of digital image processing, the x and y are discretized into pixels. The filter and the image are represented by matrices $H(i,j)$ and $P(i,j)$ respectively. The filtered image P' is computed as a convolution of P and H.

$$P'(i,j) = \sum_{m=0}^{m=k} \sum_{n=0}^{n=k} P(i+m-\frac{k}{2}, j+n-\frac{k}{2}) H(m,n) \quad (9)$$

One straight-forward way to map filtering techniques within the context of sensor networks is to treat each sensor as a pixel, and directly apply Equation 9. However, sensors may not exhibit pixel-like regularity in placement. To overcome this, we observe that Equation 9 is essentially a weighted average of all the neighboring values. Our approach is to derive the weights for the sensors based on the continuous version of the filter namely $H(x,y)$. Let PA_s be the set of all the sensors in the probing area of a sensor s_o. Let V_s be the value obtained from a sensor s, and (x_s, y_s) its location. Then the filtering output of sensor s_o is given by,

$$V_{s_o} = \sum_{\forall s \in PA_{s_o}} W(x_s, y_s) H(x_s, y_s) V_s \quad (10)$$

Here, $W(x_s, y_s)$ are weights to compensate for the uneven weighing caused due to arbitrary positioning and variations in number of the sensors. In general $W(x_s, y_s)$ is a function of sensor locations and H.

Unlike the statistical filter, then, our framework for using image processing techniques allows us to take the geographic locations into account. In general, our framework allows for different kinds of H and W functions. We do not explore this space, choosing instead to evaluate one particular localized edge detection scheme that fits into this framework.

3.2.1 The Prewitt filter based scheme

The Prewitt (difference) filter [1] in digital image processing is a set of two matrices,

$$H_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad (11)$$

$$H_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & 1 \end{bmatrix} \quad (12)$$

H_x and H_y are based on the functions,

$$H_x(x,y) = \begin{matrix} 1 & if & x \geq 0, \\ -1 & if & x < 0. \end{matrix} \quad (13)$$

$$H_y(x,y) = \begin{matrix} 1 & if & y \geq 0, \\ -1 & if & y < 0. \end{matrix} \quad (14)$$

σ_x and σ_y are the gradients in the image, along the x and y directions respectively. A high value of say $\sigma = \sqrt{\sigma_x^2 + \sigma_y^2}$ would indicate an edge.

We define V_s as,

$$V_s = \begin{matrix} 1 & if & \eta_s = 1, \\ -1 & if & \eta_s = 0. \end{matrix} \quad (15)$$

Here, η_s is the event predicate of sensor node s.

Suppose we wish to filter at node s_o based on (10), we need to decide $H_x(x_s,y_s)$, $H_y(x_s,y_s)$, $W_x(x_s,y_s)$ and $W_y(x_s,y_s)$ to calculate σ_x and σ_y.

Calculation of H_x and H_y From (10), for calculating σ_x, $H_x(x_s,y_s)$ is -1 if $x_s < x_{s_o}$, 1 if $x_s > x_{s_o}$ and 0 otherwise. For calculating σ_y, $H_y(x_s,y_s)$ is -1 if $y_s < y_{s_o}$, 1 if $y_s > y_{s_o}$ and 0 otherwise.

Selection of W_x and W_y We calculate the weights to make the scheme more tolerant to the varying number of sensors in the region. Consider, the filter H_x for calculating σ_x. Due to arbitrary placement, suppose the number of sensors to the left ($x_s < x_{s_o}$) of s_o are n_{left} and those on the right are n_{right}. Suppose $n_{left} > n_{right}$. Then filtering at s_o will be biased toward the left side. This bias can be avoided if we choose W_x such that V_s from sensors on the left by $\frac{1}{n_{left}}$ and those on the right by $\frac{1}{n_{right}}$. A similar strategy can by used for calculation of σ_y.

Let $\left(n_{i+},n_{i-}\right)_{i=1}^{i=4}$ be the number of sensors with 1 and 0 values of event predicates in the i^{th} quadrant of the probing area around the sensor in question. Quadrants are numbered in the anti-clockwise direction. The weights then become,

$$W_x(x,y) = \begin{cases} \frac{1}{n_{1+}+n_{1-}+n_{4+}+n_{4-}} & if \quad x < x_{s_o}, \\ \frac{1}{n_{2+}+n_{2-}+n_{3+}+n_{3-}} & if \quad x > x_{s_o}. \end{cases} \quad (16)$$

$$W_y(x,y) = \begin{cases} \frac{1}{n_{1+}+n_{1-}+n_{2+}+n_{2-}} & if \quad y > y_{s_o}, \\ \frac{1}{n_{3+}+n_{3-}+n_{4+}+n_{4-}} & if \quad y < y_{s_o}. \end{cases} \quad (17)$$

Based on (15),(16) and (17) we obtain,

$$\sigma_x = \frac{n_{1+}+n_{4+}-n_{1-}-n_{4-}}{n_{1+}+n_{1-}+n_{4+}+n_{4-}} - \frac{n_{2+}+n_{3+}-n_{2-}-n_{3-}}{n_{2+}+n_{2-}+n_{3+}+n_{3-}} \quad (18)$$

$$\sigma_y = \frac{n_{1+}+n_{2+}-n_{1-}-n_{2-}}{n_{1+}+n_{1-}+n_{2+}+n_{2-}} - \frac{n_{3+}+n_{4+}-n_{3-}-n_{4-}}{n_{3+}+n_{3-}+n_{4+}+n_{4-}} \quad (19)$$

The algorithm can now be stated as:

1. Collect the values $\left(n_{i+},n_{i-}\right)_{i=1}^{i=4}$ in the probing area.
2. Calculate σ using (18),(19) and compare it against a threshold σ_0 to decide whether or not the sensor is an edge sensor.

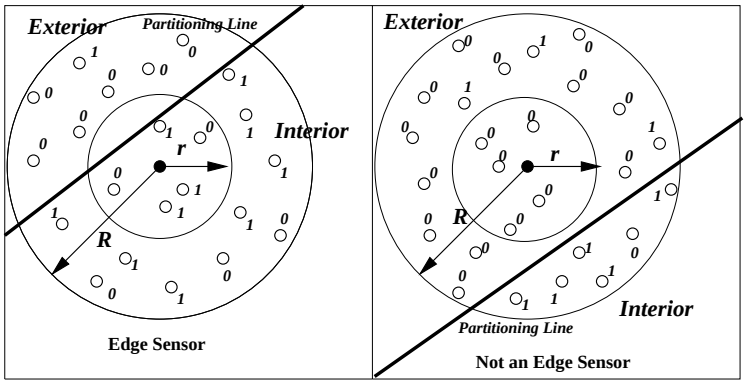

Figure 5: Classifier-based schemes attempt to determine a line which partitions all the event predicates in the probing area into 1s and 0s. If this line passes through the area of tolerance the sensor is deemed an edge sensor.

Analysis for the image processing based scheme can be done similar to the statistical scheme (as in Section 3.1.2). The analysis is provided in Appendix B.

3.3 The classifier-based approach

Our last approach comes from the pattern recognition literature. This *classifier-based* approach relies on the information provided by sensors in the interior *I* being "significantly" different from that by sensors in the exterior *O*. Such a bi-partite data set will allow classification [2] (partitioning) the data into two subsets, such that "similar" data lie in the same subset and "dissimilar" data lie in different subsets. In a classifier, a sensor would attempt to partition data gathered from its neighborhood into two classes. The success of the partition may be assessed by a *partition validity measure* [2]. A successful partition implies the presence of an edge.

The simplest classifier is a linear classifier. This classifier attempts to find a line $L(a,b,c) \equiv ax+by+c = 0$ such that all the sensors (in the probing neighborhood) with $\eta_s = 1$ are on one side of the line and those with $\eta_s = 0$ lie on the other side. A localized edge detection scheme based on a linear classifier is then quite simple. If this line passes within a distance of r from the sensor, the partition is accepted as valid and the edge is deemed as an edge sensor. Figure 5 depicts the scenario.

Two important differences exist between classifier-based edge detection and our two previous approaches. First, the linear classifier explicitly encodes a notion of geography. Second, this classifier does not require any thresholds for operation.

Classifier Definition In the event of sensor errors, an exact partition may not exist. In this case, we try to find a line which maximizes the number of sensors with like values of

event predicate on each side of the line.

Let PA_s be the set of all sensors in the probing area of sensor s. Let s_o be the sensor performing edge detection. Let $L(a,b,c)$ be the line specifying the classifier. Let V_s be as defined in Equation 15. We define **classifier score** J_s as,

$$J_{s_o}(a,b,c) = \left| \sum_{\forall s \in PA_{s_o}} V_s SN(ax_s + by_s + c) \right|. \quad (20)$$

$$SN(x) = \begin{array}{lll} -1 & if & x < 0 \\ 0 & if & x = 0 \\ 1 & if & x > 0 \end{array} \quad (21)$$

In general there can be several methods to find the optimal line based on J_{s_o}. In our implementation, we sample (θ, c) in the region $c \in [-R, R]$ and $\theta \in [0, \pi]$. Here, each sample specifies a line $L(\tan(\theta), 1, c)$, which has a slope $\tan(\theta)$ and intersects the x-axis at c. We evaluate the value of J_{s_o} at all these sample lines; the line $L_{opt}$with the highest value of J_{s_o} is chosen as the partitioning line. We then deem the partition as valid if the optimal line $L_{opt}(a,b,c)$ satisfies $\frac{ax_{s_o}+by_{s_o}+c}{\sqrt{(a^2+b^2)}} \leq r$; that is, the line is within the radius of tolerance r.

The classifier based algorithm can now be summarized as:
1. Collect all the coordinates and event predicate values within the probing radius.
2. Find a line $L_{opt}(a,b,c)$ which gives the maximum value for $J_{s_o}(a,b,c)$.
3. If L_{opt} passes within the radius of tolerance, the sensor is deemed an edge sensor.

4 Results

In this section we compare the performance of the three proposed schemes through extensive simulations. We describe the datasets used, followed by the details of the simulations. We end this section by comparing the three schemes described in the previous section.

4.1 The simulation framework

In all simulations, our sensors are located in a 200m by 200m area, their locations drawn from a uniform distribution over the area. The radio range of all the sensors is 10m and assumed omni-directional. In all simulations, we arbitrarily chose the tolerance radius r equal to the radio range of the sensors. In this context, an $\frac{R}{r} = 2$, roughly implies a 2-hop neighborhood.

The Data Sets Our simulations were conducted for two different data sets. The first, *linear boundary data sets* D_l, comprise of randomly chosen lines $y + mx + c = 0$. c is drawn from a uniform distribution over the entire x-axis within the sensor field. $m = \tan\theta$ is the slope of this line, generated by drawing θ uniformly in $(0, \pi)$. Sensors with $mx_s + y_s + c \leq 0$ belong to the interior region ($\eta_s = 1$) and rest belong to the exterior region ($\eta_s = 0$). The edge (ground truth) is defined by the line $y + mx + c = 0$. The linear boundary forms a baseline for evaluating our scheme; an acceptable edge detection scheme should perform well for this data set.

The second, *elliptical boundary data sets* D_e, consist of ellipses $E(a,b,x_0,y_0,\theta) = 0$ randomly chosen within the sensor field. $2a$ and $2b$ are lengths of the major and minor axes of the ellipse, uniformly drawn over the length of the sensor field. (x_0, y_0) is the center of the ellipse drawn uniformly over the entire sensor field. θ, which is the angle between the major axis of the ellipse and the x-axis is drawn uniformly in $(0, \pi)$. Let (x'_s, y'_s) be the sensor coordinates in a coordinate system "natural" to the ellipse (the major and minor axes of the ellipse form the x and y axes). (x'_s, y'_s) can be obtained by first translating the origin to (x_0, y_0) and then rotating the axes by θ in the anti-clockwise direction. If $1 - \frac{(x'_s)^2}{a^2} - \frac{(y'_s)^2}{b^2} \geq 0$, the sensor is deemed to belong to the interior region ($\eta_s = 1$) and to the exterior ($\eta_s = 0$) otherwise. The edge (ground truth) is defined by the ellipse $E(a,b,x_0,y_0,\theta) = 0$. Ellipses of different eccentricities represent continuously curved edges, and can be serve to distinguish localized edge detection schemes.

Factors To examine the impact of **density**, we chose three values of ρ: $\rho_1 = 1.6 \times 10^{-2}$ sensors/sq.mt (low density - about 5 senors within radio range), $\rho_2 = 3.6 \times 10^{-2}$ sensors/sq.mt (moderate density - about 15 senors within radio range), and $\rho_3 = 7.2 \times 10^{-2}$ sensors/sq.mt (high density - about 30 senors within radio range).

To capture the impact of **sensor errors**, we used a simple bit flipping technique. In this model, a sensor toggles its event predicate value from its true value with a probability p. We used three different choices for p. $p_1 = 1\%$ (low), $p_2 = 5\%$ (moderate) and $p_3 = 10\%$ (high).

Thus, for the linear boundary data set, a single simulation run represents one line chosen randomly, for one value of density and sensor error. For a given density and sensor error probability, we *average* the performance metrics for a localized edge detection scheme over 20 different runs corresponding to different randomly chosen lines. The same is true for the ellipse.

Parameters We chose five different values of $\frac{R}{r}$, namely 1, 1.5, 2, 2.5 and 3. We ran simulations for all the data sets, for each of the three schemes, for the five values of $\frac{R}{r}$.

The statistical and the image processing schemes require choosing $\gamma_0 \in (0,1)$ and $\sigma_0 \in (0,1)$ respectively. This choice

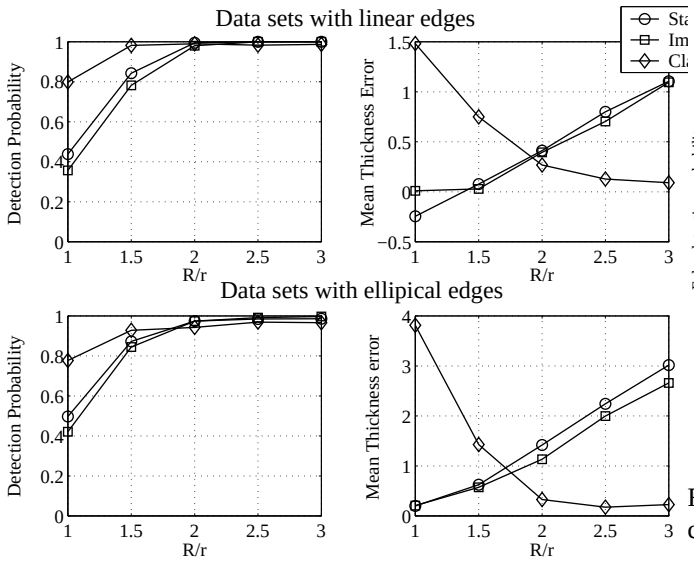

Figure 6: Energy accuracy trade-off : As more and more neighborhood is examined ($\frac{R}{r}$ the detection probability increases for all the three detection schemes.)

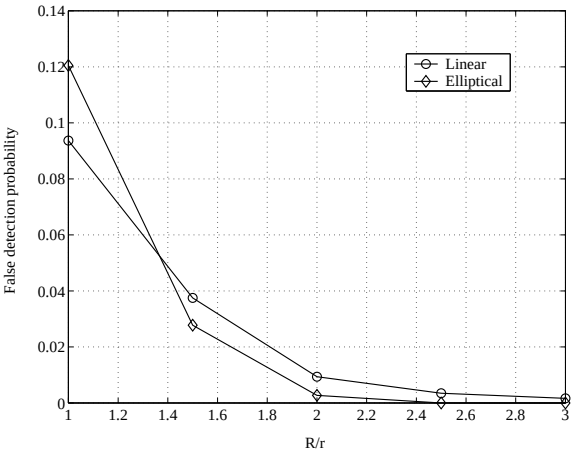

Figure 7: False detections for the classifier based scheme decrease with increase in $\frac{R}{r}$

(as discussed in Section 3.1.2) can impact performance. To be fair to all schemes, for a given simulation run, we chose the best threshold value (using the analysis in Appendix A and B) defined thus: *"Choose the threshold which satisfies $e_{pf} \leq 1\%$ and minimizes e_m"*. Thus, for schemes that require thresholds, our simulations represent the fewest possible missed detections.

We evaluate our schemes with respect to the metrics described in Section 2.2.

4.2 Simulation results

In this section we discuss the results of our simulations. The space of parameters and factors we have explored is large. Rather than exhaustively present all of our results, we selectively describe the simulation results in an effort to give the reader an understanding of the main differences between the schemes.

We start by considering (Figure 6) which shows the variation of e_t (mean thickness error) and $1 - e_m$ (detection probability) for moderate error ($p = 5\%$) and moderate density ($\rho = 3.6 \times 10^{-2}$) with $\frac{R}{r}$. It depicts the basic nature of the *energy accuracy trade-off*. As seen in Figure 6, the edge detection probability increases with increase in $\frac{R}{r}$. However, increasing $\frac{R}{r}$ increases the communication overhead as $o(R^2)$ and hence the energy consumption. For linear data sets all the three schemes give similar detection ratios at $\frac{R}{r} \geq 2$, while the classifier gives a thinner edge. For elliptical data sets, at $\frac{R}{r} \geq 2$, the classifier performs slightly inferior to the other two schemes, however it gives a much thinner edge. The performance of statistical and image processing based schemes is similar.

The statistical and image processing based schemes, allow one to restrict the false detection probability by selecting an "appropriate" choice of threshold (γ_0 and σ_0). In all our simulations, the choices restricted false detection to below 1%. In the classifier based scheme, there is no such direct way to restrict false error probability. Figure 7 shows the variation of false detections made by the classifier scheme with increase in $\frac{R}{r}$ for the moderate density, moderate sensor errors data sets. The false detection probability decreases with increase in $\frac{R}{r}$.

The classifier scheme behaves *qualitatively* differently from the statistical and image processing schemes. For the latter, as the $\frac{R}{r}$ ratio increases, the edge thickness increases while for the classifier based schemes the edge thickness decreases. Edge thickness error results from pure false detections (e_{pf})and unwanted detections (e_{ud}). Since we restricted $e_{pf} < 1\%$ for ths statistical and image processing based schemes, edge thickness error mostly arises out of e_{ud}. In the classifier based scheme e_{ud} is small and does not change significantly with $\frac{R}{r}$. This is depicted in Figure 8. The increase in e_{ud} causes an increase in thickness error for the statistical and image processing based schemes, while a decrease in false detections leads to a *decrease* in thickness error for the classifier based scheme. For this regime, then, the classifier based scheme represents a low-energy technique for achieving thin edges with high likelihood of true detections.

What happens when we change density but keep sensor error constant? Predictably the detection probability increases with increase in sensor density for both kinds of data sets. This is shown in Figure 9. There was no σ_0 which gave an $e_{pf} < 1\%$

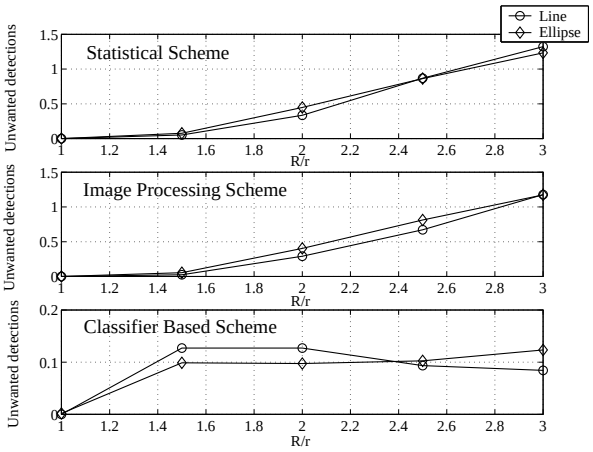

Figure 8: Variation of e_{ud} with increase in $\frac{R}{r}$ for the three schemes for moderately dense sensor fields with moderate errors.

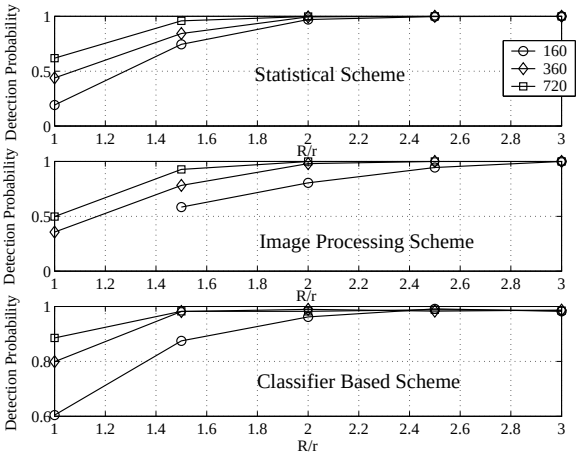

Figure 9: Variation of $1-e_m$ (detection probability), with increase in density for the three schemes for linear edge data sets.

at $\rho = 1.6 \times 10^{-2}$, hence this point is missing. We also found that while the thickness error increases with increase in density for the statistical and image processing based schemes, it decreases for the classifier based scheme. The reason is that, unwanted errors, which dictate thickness error for the statistical and image processing schemes increase with increase in sensor density. However, the false detections which dictate the edge thickness error for the classifier based scheme decrease with increase in density.

How sensitive are the schemes with respect to sensor errors? Keeping density fixed, we notice an expected qualitative trend. The detection probability decreases with increase in sensor errors for all the three schemes. This is shown in Figure 10. We also found that the thickness error increases with increase in sensor error for all the three schemes and both kinds of data

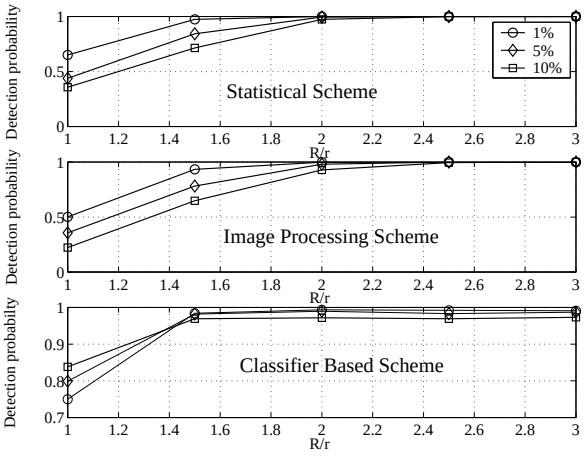

Figure 10: Variation of $1-e_m$ (detection probability), with increase in sensor error p for the three schemes for linear edge data sets.

sets.

Finally, how critical is the choice of appropriate thresholds? The classifier-based scheme does not require selection of a threshold. The other schemes do, and in the results we have presented, we have chosen the best threshold possible for each particular scenario. The threshold based schemes might have been acceptable if there existed one or a small range of thresholds that were acceptable over the density and error ranges we consider. However, we found that the thresholds in the statistical scheme vary very widely with changes in $\frac{R}{r}$ and p, especially at low densities. For instance the optimal value of γ_0 is 0.79 for $\frac{R}{r} = 1$, $p = 0.05$ and $\rho = 1.6 \times 10^{-2}$. The optimal value of γ_0 is 0.17 for $\frac{R}{r} = 3$,$p = 5\%$ and $\rho = 7.2 \times 10^{-2}$. For the image processing based scheme, it turns out that the variation in the choice of σ_0 is very small (within 10%) with respect to $\frac{R}{r}$ and p at low densities. However the scheme exhibits variations similar to the statistical scheme at higher sensor densities for changes in $\frac{R}{r}$.

This discussion leads to the following conclusions. *Over a range of sensor error rates and densities, all the three scheme can achieve true detection rates of 90% or better by using a two-hop probing radius. Among the three schemes the classifier provides the thinnest edges and performs better with increasing probing radius*. The classifier based scheme does not require choosing appropriate thresholds. Thus, from a practical perspective, the classifier-based scheme represents a low energy approach to accurate localized edge detection. Even though the classifier based scheme does not provide a direct control over false detection errors (as thresholds in other two schemes do), one can increase the probing radius (*e.g.*, 3-hop probing) and achieve lower false detections at the cost of more communication cost. Recall that localized edge detection will usually be a component of a larger system that, for example,

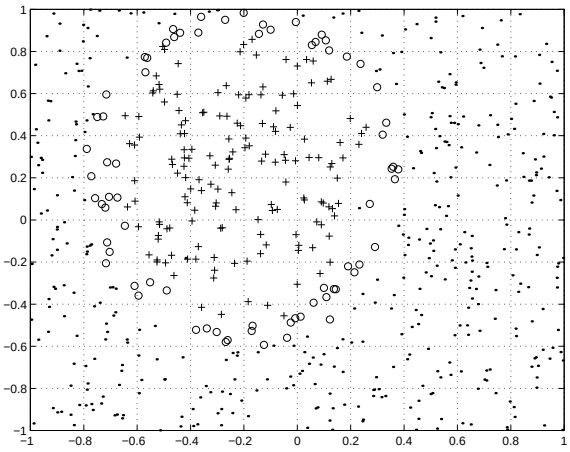

Figure 11: Classifier based edge detection on a low density low error data set. Each unit on x and y axis represent 100m. 'o' represent edge sensors and $+$ the interior.

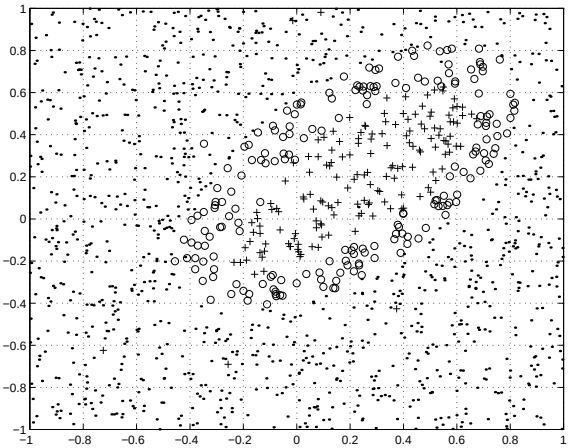

Figure 12: Image processing based edge detection on a moderate density moderate error data set. Each unit on x and y axis represent 100m. 'o' represent edge sensors and $+$ the interior.

computes boundaries of a phenomenon. We observe that false detections can be disambiguated at the level of these boundary finding algorithm. If the algorithm that constructs the boundary from the observed edges *also uses information about each edge sensor's partition line*, it should be able to detect inconsistent partition line orientations and locations among neighboring edge sensors caused by false detections. We believe that a better scheme which relies on edge continuity information will result in fewer false detections.

For this reason, we suggest that, of the three schemes we consider, the classifier is the most promising for localized edge detection. Figures 11,12,13 show three examples of the three edge detection algorithms at work.

5 Conclusion

In this paper we introduced the problem of localized edge detection in a sensor field. We discussed an "edge" and proposed metrics to assess edge detection algorithms. We proposed three qualitatively different approaches to edge detection namely statistical, image processing based and classifier based approaches. We proposed an example scheme for each of these approaches. Through numerous simulations we compared the three schemes with respect to the energy accuracy trade-off, sensitivity to choice of parameters and performance.

Our results indicate that the classifier scheme performs much better than the other schemes. Under higher sensor error conditions, it is susceptible to more false detections than other schemes. These false detections can be reduced at the expense of higher communication cost or can probably be disambiguated by a higher-level boundary finding algorithms. The

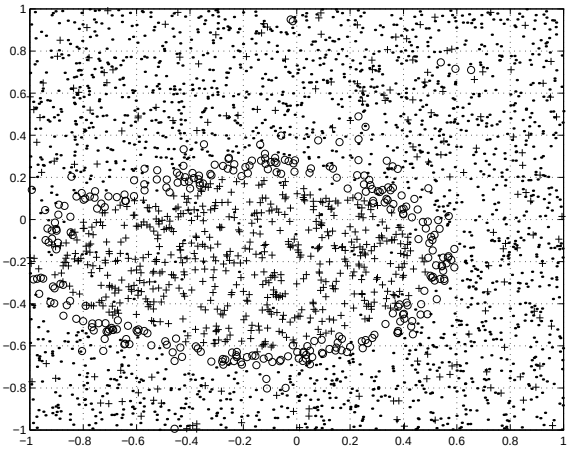

Figure 13: Statistical scheme based edge detection on a high density high error data set. Each unit on x and y axis represent 100m. 'o' represent edge sensors and $+$ the interior.

statistical and image processing based scheme can exhibit similar performance but only if detection thresholds are correctly set. The correct detection thresholds vary widely with density and sensor error and we believe that dynamically setting thresholds by empirically observing densities will be hard to do. As such, then, of the schemes we consider, the classifier based scheme seems to be the most promising for localized edge detection.

References

[1] Bernd Jähne, *Digital Image Processing,*, Springer, 4th edition, 1997.

[2] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification*, John Wiley and Sons, 2nd edition, 2000.

[3] B. K. De La Barre, T. C. Harmon, C. V. Chrysikopoulos, *Measuring and Modeling the Dissolution of a Non-ideally Shaped Dense Non-aqueous Shaped Liquid Pool in a Saturated Porous Medium*, Water Resources Research, 2002.

[4] J. Liu, J. Reich, F. Zhao, *Collaborative In-Network Processing for Target Tracking*, Journal on Applied Signal Processing, 2002.

[5] R. Nowak, U. Mitra, *Boundary Estimation in Sensor Networks: Theory and Methods*, Proceedings of the First International Workshop on Information Processing in Sensor Networks, April 2003.

[6] A. Savvides, A., C.-C. Han, M. Srivastava, Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking* (Rome, Italy, July 2001).

[7] A. Savvides, S. Adlakha, R. Moses, M. Srivastava, *On the Error Characteristics of Localization Algorithms*, Proceedings of the First International Workshop on Information Processing in Sensor Networks, April 2003.

APPENDIX A

In this appendix we calculate e_m, e_{uw} and e_{pf} to analyze the scheme described in Section 3.1.1.
We assume that the phenomenon is "large" enough and the edge can be approximated by a line segment L within the probing neighborhood. Let L be l units distant from the sensor. We assume that line segments at all values of l are equally likely. Suppose an edge passes at a distance l from the sensor (as depicted in Figure 14). The sensor collects

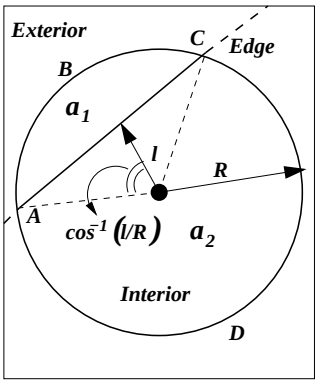

Figure 14:

information about the exterior from area a_1 (ABC) and about the interior from a_2 (ADC).

$$a_1 = R^2 \cos^{-1}\left(\frac{l}{R}\right) - l\sqrt{R^2 - l^2}, \quad \text{(A-1)}$$

$$a_2 = \pi R^2 - a_1, \quad \text{(A-2)}$$

The number of sensors N_1 and N_2 in these regions can be modeled as Poisson random variables.

$$P(N_i = n) = e^{-a_i\rho}\frac{(a_i\rho)^n}{n!} \quad \text{(A-3)}$$

The number of sensor errors K_1 (in ABC) and K_2 (in ADC), can be modeled by a binomial random variable with p as sensor error probability.

$$P(K_i = k | N_i = n) = \left(\begin{array}{c} n \\ k \end{array}\right)(p)^k (1-p)^{n-k}. \quad \text{(A-4)}$$

The value of the statistic in (4) can now can be written in terms of K_i and N_i as,

$$\Gamma = 1 - \frac{\left|N_1 + 2K_2 - N_2 - 2K_1\right|}{N_1 + N_2}. \quad \text{(A-5)}$$

Using equations (A-1)-(A-5), we can numerically calculate the probability density function, $P(\Gamma = \gamma | l = q)$.

Calculation of e_m and e_{uw} : A miss-detection occurs when $l \le r$ and $\Gamma < \gamma_0$.

$$P(l = q | l \le r) = \frac{2q}{r^2} \quad \text{(A-6)}$$

$$P(\Gamma = \gamma | l \le r) = \int_0^r P(\Gamma = \gamma | l = q)\frac{2q}{r^2}\,dq, \quad \text{(A-7)}$$

$$e_m = \int_0^{\gamma_0} P(\Gamma = \gamma | l \le r)\,d\gamma. \quad \text{(A-8)}$$

An unwanted detection occurs when $R \ge l > r$ and $\Gamma \ge \gamma_0$.

$$P(l = q | R \ge l > r) = \frac{2q}{R^2 - r^2} \quad \text{(A-9)}$$

$$P(\Gamma = \gamma | R \geq l > r) = \int_r^R P(\Gamma = \gamma | l = q) \frac{2q}{R^2 - r^2} dq, \quad \text{(A-10)}$$

$$e_{uw} = \frac{R^2 - r^2}{r^2} \int_{\gamma_0}^1 P(\Gamma = \gamma | R \geq l > r) \, d\gamma \quad \text{(A-11)}$$

Calculation of e_{pf} : Now suppose there is no edge with the probing neighborhood ($l > R$). Γ can assume non-zero values only because of sensor errors. The *pdf* of number of sensors in the neighborhood region N can be expressed as a Poisson random variable.

$$P(N = n | l > R) = e^{-\pi R^2 \rho} \frac{(\pi R^2 \rho)^n}{n!}. \quad \text{(A-12)}$$

The number of sensor errors K in the probing neighborhood can be expressed as a binomial random variable.

$$P(K = k | N = n) = \binom{n}{k} (p)^k (1-p)^{n-k}. \quad \text{(A-13)}$$

The value of the statistic in (4) can now can be written in terms of K and N as,

$$\Gamma = 1 - \frac{|N - 2K|}{N}. \quad \text{(A-14)}$$

We numerically calculate the *pdf* $P(\Gamma = \gamma | l > R)$. A pure false error occurs when, $\Gamma > \gamma_0$.

$$e_{pf} = \frac{N}{\rho r^2} \int_{\gamma_0}^1 P(\Gamma = \gamma | l > R) \, d\gamma. \quad \text{(A-15)}$$

N is the total number of sensors in the field.

APPENDIX B

In this appendix we calculate the errors e_f, e_{uw} and e_{pf} for Prewitt filter example in Section 3.2.1. Let the line segment $L(l, \theta)$ approximate the edge (as argued in Appendix A) intersect the probing neighborhood. Here, l is its distance from the sensor and θ is the angle the y-axis makes with the normal from the sensor (see Figure 15). We divide the neighborhood into 4 areas, a_1 (DCHF), a_2 (ABCD), a_3 (ADEG) and a_4 (EDFI) as depicted in Figure 15. The four areas $(a_i)_{i=1}^{i=4}$ for $\theta \in (0, \frac{\pi}{2})$ can be calculated by the equations,

$$a_1 = \begin{cases} \frac{R^2}{2}\left(\cos^{-1}(\frac{r}{R}) - \theta\right) \\ \quad -\frac{r}{2}\left(-r\tan\theta + \sqrt{R^2 - r^2}\right) & \theta < \cos^{-1}(\frac{r}{R}) \quad \text{(B-1)} \\ 0 & \theta \geq \cos^{-1}(\frac{r}{R}). \quad \text{(B-2)} \end{cases}$$

$$a_2 = \begin{cases} \frac{R^2}{2}\left(\cos^{-1}(\frac{r}{R}) + \theta\right) \\ \quad -\frac{r}{2}\left(r\tan\theta + \sqrt{R^2 - r^2}\right) & \theta < \cos^{-1}(\frac{r}{R}) \quad \text{(B-3)} \\ \frac{R^2}{2}\cos^{-1}(\frac{r}{R}) - r\sqrt{R^2 - r^2} & \theta \geq \cos^{-1}(\frac{r}{R}). \end{cases}$$

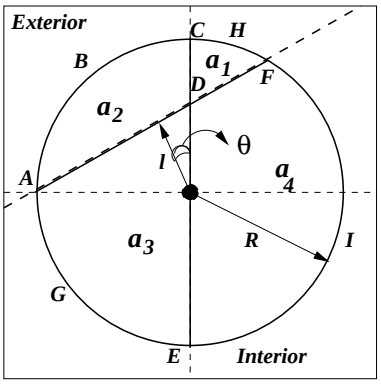

Figure 15: Figure for appendix B

$$a_3 = \pi\frac{R^2}{2} - a_2. \quad \text{(B-4)}$$

$$a_4 = \pi\frac{R^2}{2} - a_1. \quad \text{(B-5)}$$

The number of sensors $(N_i)_{i=1}^{i=4}$ in the four regions and the number of sensor errors $(K_i)_{i=1}^{i=4}$ can be modeled as in A-3 and A-4. The value of σ_x in (18) can now can be written in terms of K_i and N_i as,

$$\sigma_x = \left| \frac{N_2 - 2K_2 + N_3 - 2K_3}{N_2 + N_3} + \frac{N_1 - 2K_1 + N_4 - 2K_4}{N_1 + N_4} \right|. \quad \text{(B-6)}$$

The *pdf* $P(\sigma_x = \mu | l, \theta)$ can be numerically calculated using, (B-1)-(B-6), (A-3) and (A-4).

It turns out that $P(\sigma_y = \mu | l, \theta)$ is same as $P(\sigma_y = \mu | l, \frac{\pi}{2} - \theta)$. The *pdf* $P(\sigma = \mu | l, \theta)$ can be calculated numerically from $\sigma = \sqrt{\sigma_x^2 + \sigma_y^2}$.

Calculation of e_m and e_{uw} : A miss-detection occurs when $l \leq r$ but $\sigma < \sigma_0$.

$$P(\sigma = \mu | l \leq r) = \frac{8}{\pi r^2} \int_0^{\frac{\pi}{4}} \int_0^r P(\sigma = \mu | l, \theta) \, dl \, d\theta \quad \text{(B-7)}$$

$$e_m = \int_0^{\sigma_0} P(\sigma = \mu | l < r) \, d\mu \quad \text{(B-8)}$$

An unwanted detection occurs when $R \geq l > r$ but $\sigma \geq \sigma_0$.

$$P(\sigma = \mu | R > l \geq r) = \frac{8}{\pi(R^2 - r^2)} \int_0^{\frac{\pi}{4}} \int_r^R P(\sigma = \mu | l, \theta) \, dl \, d\theta, \quad \text{(B-9)}$$

$$e_{uw} = \frac{R^2 - r^2}{r^2} \int_{\sigma_0}^1 P(\sigma = \mu | R > l \geq r) \, d\mu. \quad \text{(B-10)}$$

Here, we integrate on θ only over $(0, \frac{\pi}{4})$ since, $P(\sigma = \mu | l, \theta)$ is identical in any section $(\frac{n\pi}{4}, \frac{(n+1)\pi}{4})$.

Calculation of e_{pf} : Now suppose there is no edge with the probing neighborhood ($l > R$) and a non-zero value of σ occurs due to sensor errors. Let N_i be the number of sensors in the i^{th} quadrant (quadrants numbered in anti-clockwise direction). Let K_i be the number of sensor errors in the i^{th} quadrant. Then,

$$P(N_i = n) = e^{-\frac{\pi R^2}{2}\rho} \frac{\left(\frac{\pi R^2}{2}\rho\right)^n}{n!} \quad \text{(B-11)}$$

$$P(K_i = k | N_i = n) = \left(\begin{array}{c} n \\ k \end{array} \right) (p)^k (1-p)^{n-k}. \quad \text{(B-12)}$$

The value of σ can be calculated in terms of $(N_i)_{i=1}^{i=4}$ and $(K_i)_{i=1}^{i=4}$.

$$\sigma_x = \left| \frac{N_2 + 2K_3 - N_3 - 2K_2}{N_2 + N_3} + \frac{N_1 + 2K_4 - N_4 - 2K_1}{N_1 + N_4}, \right| \quad \text{(B-13)}$$

$$\sigma_y = \left| \frac{N_1 + 2K_2 - N_2 - 2K_1}{N_1 + N_4} + \frac{N_3 + 2K_4 - N_3 - 2K_4}{N_3 + N_4}, \right| \quad \text{(B-14)}$$

$$\sigma = \sqrt{\sigma_x^2 + \sigma_y^2} \quad \text{(B-15)}$$

Using (B-11)-(B-15), we can numerically calculate the *pdf* $P(\sigma = \mu | l > R)$. A pure false error occurs when, $\sigma > \sigma_0$.

$$e_{pf} = \frac{N}{\rho r^2} \int_{\sigma_0}^{1} P(\sigma = \mu | l > R)\, d\mu. \quad \text{(B-16)}$$

N is the total number of sensors in the field.