Stateless and Efficient Boundary Simplification of Phenomena in Sensor Networks

Sai Hin Tse*, Lars Kulik†, Egemen Tanin‡ and Anthony Wirth§
National ICT Australia
Department of Computer Science and Software Engineering
University of Melbourne, Victoria, 3010, Australia
{*shtse,†lkulik,‡etanin,§awirth}@unimelb.edu.au

***Abstract*—Object tracking has been a prominent application domain of sensor networks. For many applications, it is sufficient to monitor the boundary of a physical phenomenon instead of the full extent of the phenomenon in return for significant energy savings. However, continuous monitoring of a large phenomena with irregular boundaries will still incur a significant energy expenditure in a sensor network. We propose a novel distributed algorithm, SpatialZip, to efficiently simplify the boundary resulting in less energy expenditure.**

I. Introduction

Wireless sensor networks (WSNs) can be used to monitor a wide variety of physical phenomena such as toxic gas plumes, soil wetness in vineyards or oil spills. In many cases, we can decompose the region we are monitoring into two distinct regions, one with some physical property above a threshold value and the other below this threshold value, with a boundary separating these two regions. For example, a WSN may be deployed to monitor the soil wetness in a vineyard and will trigger water sprinklers in different areas of the vineyard once the soil wetness falls below a threshold value.

Communication within a WSN is achieved by relaying information between sensors that are close to each other using a communication protocol. A message may have to be forwarded many times before it reaches its intended location, which is usually the sink that disseminated the initial query. The relevant nodes that answer the query will have to transmit the results of the query through a series of nodes before reaching the sink. This means that even though a particular node need not respond to a particular query, it may have to take part in the transmission process of the results of the query.

A well-known limitation of WSNs is the amount of energy available to each sensor. Since sensors are usually battery powered, when a battery runs out of energy, the sensor will no longer be able to broadcast and relay signals. When too many sensors in the network run out of power, the network may become segregated and sensors on one side of the network may not be able to send messages to the other side if there is no routing path across the sensors that ran out of power. Therefore it is important to develop energy-saving techniques for applications in sensor networks.

Energy-efficient boundary detection and monitoring has been a vital field of research [1], [2]. For example, DEMOCO [3] serve this task. However, ultimately these algorithms have to report the boundary back to the sink by routing the boundary information through the sensor network. In this paper, we propose to improve on the energy efficiency of these algorithms by simplifying the boundary of the detected phenomenon. The saving in energy comes from two sources: fewer sensors on the boundary having to actively detect changes, and less information about the boundary being reported.

Boundary simplification by overestimation is acceptable in many applications where the phenomenon is known to be growing, as areas close to the phenomenon are likely to be affected soon. This means that the regions that are overestimated have a high chance of being inside the phenomenon later.

Although there has been some work done on line simplification [4], feature identification and simplification [5], [6] and Spatial Approximation Algorithms in Sensor Networks [7], [8], these existing algorithms are all centralized. An important feature is to be able to locally compute a simplified boundary. An approach using a central source (a sink) requires additional energy as it requires additional messages between the source and the boundary nodes. The savings from the simplified boundary may not be able to justify the energy initially invested in *simplifying* the boundary, which prompts the need for a decentralized simplification algorithm.

The main contribution of this work is an algorithm called SpatialZip. SpatialZip is a stateless, decentralized and efficient algorithm for simplifying a boundary. To simplify the boundary, we identify concavities in the boundary (inlets) and zip them up. This is done recursively until no more inlets can be identified. Efficiency is achieved by using only information local (one-hop) to sensors when identifying inlets hence requiring little extra communication overhead when simplifying the boundary. Statelessness is assured by not requiring non-local information to be stored.

In summary, advantages of the SpatialZip algorithm over existing centralized boundary simplification algorithms are:

- identify concavities locally for energy efficiency;
- efficient simplification of a growing boundary; and
- produces a simplified boundary that can be easily monitored for change.

II. Preliminaries

In this paper, we focus on phenomena without holes, whose boundaries are connected and contains no dangling nodes. In

We acknowledge Archana Sathivelu of The University of Melbourne whose master's thesis titled *Efficient Boundary Monitoring of Spatial Phenomena in Sensor Networks* was a key preliminary research that led to this paper.

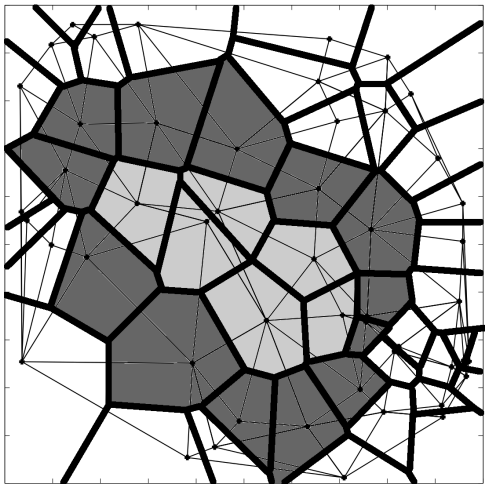

Fig. 1. This image illustrates our definition of a boundary. The light gray cells indicate the Voronoi cells that are inside the boundary. The dark gray cells show the cells monitored by BSs. All the BSs are connected to the next BS by a Delaunay edge, which is represented by a thin black line.

future work, we drop one or more of these assumptions.

We use a Voronoi Diagram [9] as our underlying data structure as it captures the notion of neighborhoods. A Voronoi diagram partitions a plane containing a set of sites *S*, the sensor nodes, into a collection of *cells*. It associates a cell with a site p so that all points in cell *Cell(p)* are closer to p than to any other site in *S*. In our case, each Voronoi cell is the node's area of responsibility. When a node detects the phenomenon, we assume, as an approximation, that the phenomenon covers the entire Voronoi cell.

We assume that the Voronoi Diagram is computed prior to any boundary detection and simplification. Energy efficient and distributed methods for computing Voronoi diagrams that are suitable for applications in WSN have been studied in [10] where each node is able to generate its Voronoi cell by just considering its immediate neighbors. Since the Voronoi diagram only needs to be calculated once, the cost of calculating it is included into the initial setup cost of the deployment. Furthermore, only the sink needs the entire Voronoi diagram for reconstructing the phenomenon using boundary information; other nodes only require information about their local Voronoi neighbors for our algorithm to work.

We define the Region of Interest (ROI) as the area covered by the phenomenon and assume that its boundary can be approximated by a discretization defined by the locations of *boundary sensors* (BSs). We define a BS as a node inside the ROI that has at least one neighboring node outside the ROI. The neighborhood of a node p comprises the nodes that are Voronoi neighbors of p (nodes whose Voronoi cells share an edge with p's cell). The boundary is a series of BSs that are neighbors of each other. The dual of the Voronoi Diagram is the Delaunay Triangulation. If two nodes are neighbors of each other, they will be connected by a Delaunay edge. A Delaunay edge must connect each BS to the next BS on the boundary as shown in Figure 1.

As highlighted in the introduction, simplification of the boundary is achieved by removing inlets and concavities from the boundary. To remove an inlet from the boundary, the node causing the inlet should be included in the ROI as the concavity gets filled up.

To formalize the above process, we will call nodes that are near the boundary *near boundary sensors* (NBSs). NBSs are nodes not inside the ROI that have at least one neighboring BS. If a NBS causes an inlet, we include it in the ROI by promoting it to a *virtual boundary sensor* (VBS). A VBS is a NBS that is considered to be in the ROI for boundary simplification. Once a NBS is promoted to a VBS, it is treated like the other BSs in its neighborhood as it delineates the sensors within the region from those outside the region.

We will evaluate the performance of our algorithm by first measuring the decrease in the number of nodes on the boundary. This reduction will also decrease the energy expended when transmitting the boundary to the sink. Second, we will measure the increase in area enclosed by the boundary. A larger increased enclosed area is a loss of resemblance of the phenomenon. Furthermore, Spatialzip should not introduce bulges to the boundary as they increase its length and the area inside the boundary.

III. SpatialZip

The key idea of our algorithm is to identify the NBSs that can be chosen as inlet cells and then promote them to VBSs. Repeated promotion of NBSs into VBSs will result in the simplification of the inlet. It is important to be able to identify inlets using only local knowledge because non-local information requires additional energy expenditure, defeating the purpose of boundary simplification.

Limited work has been done on identifying and simplifying certain features such as inlets on a boundary. Therefore, we have developed geometric criteria for identifying inlet nodes on the boundary.

The criteria depend on the identification of extreme nodes of NBSs which we will present first.

A. Extreme Nodes

Extreme nodes are important because they determine if a node causes an inlet on the boundary. Let x be an NBS. We define the extreme nodes of x as the nodes in the neighborhood of x that have NBS neighbors other than x. Extreme nodes can be identified using purely local information provided that the BSs neighboring x are connected. If a NBS x has more than two extreme nodes, the boundary is disconnected in which case x is not promoted to a VBS to avoid holes in the ROI.

The extreme nodes can be readily computed in $O(d(x) \log d(x))$ time where $d(x)$ is the number of boundary nodes in the neighborhood of x. We construct a convex hull [11] of the neighboring boundary sensors and the two nodes that are connected by an edge on the convex hull without a corresponding Delaunay edge will be the two extreme nodes (see nodes e_1 and e_2 in Figure 2). The Delaunay edges do not need to be calculated during the identification of extreme nodes because the Voronoi Diagram is assumed to be precomputed from which Delaunay edges can be obtained.

It should be noted that extreme nodes cannot be found if a NBS only has two neighboring BSs. This is not a limitation of our method: if a NBS only has two neighboring BSs,

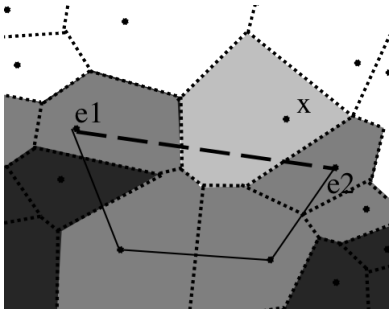

Fig. 2. The dark gray cells are the Voronoi regions of the nodes inside the boundary. The gray cells are the nodes on the boundary. The white cells are the nodes not in the boundary. The light gray cell is the NBS we are currently considering. The thin solid lines are the Delaunay edges and the thick dashed line is the edge of the hull that has no corresponding Delaunay edge. Nodes e_1 and e_2 are the extreme nodes identified. It is obvious that e_1 and e_2 determines if x causes an inlet or not.

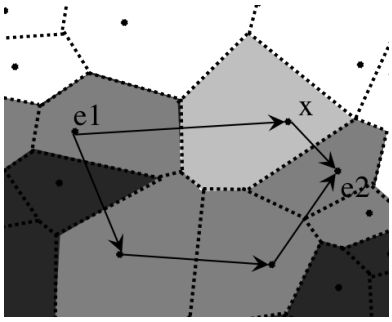

Fig. 3. e_1 and e_2 are the extreme nodes of x. Clearly, the distance from e_1 to e_2 through x is less than the path length along the original boundary. Hence the Length Criterion is met and x should be promoted to a VBS.

promoting the NBS to a VBS will result in a longer boundary and hence should not be considered in the first place.

B. Geometric Criteria

Now that we have established the notion of extreme nodes, we present our geometric criteria for inlet identification.

1) Length Criterion: The Length Criterion for identifying inlets states that if the sum of the distances from extreme node e_1 to x and from x to another extreme node e_2 is less than the distance from e_1 to e_2 along the original boundary, then x must be an inlet (see Figure 3). In an ideal scenario, we could connect e_1 to e_2 with a straight line hence closing the inlet without bulging. If there is no direct path between these two nodes, the path most similar to a straight line is taken because it will introduce the least bulging or indentation to the boundary. By geometry, this occurs when the alternative path is lowest. Furthermore, given a uniform distribution of nodes, the number of nodes on the boundary should be proportional to the length of the boundary hence it is desirable to reduce the boundary length.

2) Extended Length Criterion: The extended length criterion is a extension of the length criterion. The node x is temporarily promoted to a VBS. If at least one of x's neighbors is promoted to a VBS by either the Length Criterion or the Extended Length criterion as a result of x's temporary promotion, then x's promotion should be made permanent. If no neighbor is promoted as a result of x's promotion, x should be demoted back to a NBS. This criterion is useful for inlets

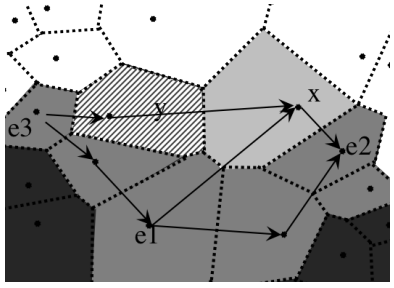

Fig. 4. e_1 and e_2 are the extreme nodes of x. e_3 and e_1 are the extreme nodes of y. x and y are both NBSs. Both x and y are in an inlet: if they are both promoted, the boundary will be shorter. x and y will not be promoted by the Length Criterion alone. However, if x were temporarily promoted to a VBS, x would become an extreme node of y and the distance from e_3 to x through y would be shorter than the distance through the original boundary. Therefore y would be promoted to a VBS by the Length Criterion. Since y's promotion was a result of x's temporary promotion, x will remain a VBS according to the Extended Length Criterion.

that require more than one NBS to be promoted at the same time to fully zip up (see Figure 4).

Note that the Extended Length Criterion can propagate to nodes that are not one-hop neighbors of x. When x is temporarily promoted, its neighbors may also be temporarily promoted if they meet the Extended-Length Criterion. The Extended Length Criterion is stateless because it only maintains the state of one-hop neighbors.

3) Other Criteria: Other possible geometric criteria have been examined. We investigated using the number of neighbors as well as the convex hull of neighbors to determine if a node is causing a inlet. Preliminary experimentation conducted on these geometric criteria showed little promise since they were at times either too conservative in identifying inlets, which resulted in inlets not being zipped, or were too optimistic in identifying inlets, causing unnecessary bulging. Hence, further investigation of these criteria was not pursued.

C. SpatialZip Algorithm

Now that we have defined how we will identify inlets, we present our boundary simplification algorithm in Algorithm 1.

Algorithm 1 SpatialZip(x,VN_x)

/VN_x is the set of Voronoi Neighbors for node x */*

```
1: [e1,e2]=compute_extreme_nodes(x,VN_x)
2: if Length_Criterion(x,VN_x,e1,e2) OR
3: Extended_Length_Criterion(x,VN_x,e1,e2) then
4:     VBS_x ← TRUE //Promote x to VBS
5:     for all i in VN_x lying outside the region do
6:         SpatialZip(i,VN_i)
7:     end for
8: end if
```

The whole zipping process is initiated when a node discovers that it causes a inlet on the boundary. It will turn into a VBS and the zipping propagates to neighboring nodes until no more inlets can be identified. Consequently, the perimeter of the boundary in that region will have reached a local minima.

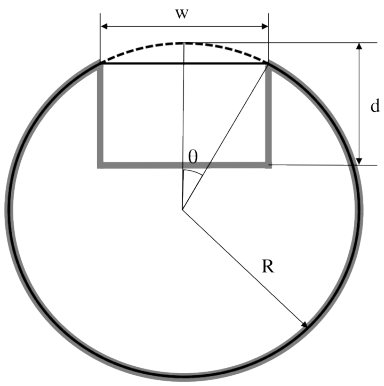

Fig. 5. The above figure illustrates our notion of a theoretical average result. The thick gray lines and the thin black lines represent the boundary before and after simplification respectively. The dotted line shows the circle from which at inlet was cut to create the phenomenon. This boundary will only be formed with ideal placements of nodes.

IV. Performance Evaluation

A. Evaluation Parameters

We first formalize the evaluation parameters that we use to analyze the performance of SpatialZip.

1) Reduction in Boundary: A measure of our algorithm's effectiveness in reducing energy expenditure is the reduction in the number of nodes on the boundary. If there were originally N nodes on the boundary and M nodes on the boundary after SpatialZip, we define the *reduction in boundary* (BR) to be:

$$BR = (N - M)/N$$

2) Increase in Area inside Boundary: The *increase in contained area* (IA) signifies the cost of the simplification of the boundary. The original ROI is always a subset of the ROI resulting from SpatialZip since we are simplifying by overestimation. Let the area enclosed by the original boundary be A and the area enclosed by the simplified boundary be B, the IA is defined as:

$$IA = (B \setminus A)/A$$

B. Theoretical Average Solution

In order to provide a benchmark for the performance of SpatialZip, we introduce the theoretical average solution.

We define the theoretical average solution to be the BR and IA resulting from removing the inlet from a boundary that exactly bounds the phenomenon. This is illustrated in Figure 5. The theoretical average solution is an indicator of how the BR and IA are affected by the experiment settings if they were only constrained by our definition of our boundary. For a boundary to be in-line with our definition of a boundary, each BSs must be a neighbor of the next BS on the boundary. Discrepancies between the trend of the theoretical average solution's performance and SpatialZip's performance can be attributed to properties of our algorithm and how it is affected by the distribution of sensor nodes.

Tonguz and Ferrari [12] showed that the average distance r between Voronoi Neighbors for an *average uniform* node spatial distribution is given by $1/\sqrt{\rho}$ where ρ is the node spatial density. A node distribution is uniform if there exists a Voronoi tessellation in which every Voronoi cell is contained in a disk of radius r. The node spatial density can be shown to be equal to the probability P of a cell having a node placed in it. Therefore, the average distance between Voronoi Neighbors can also be expressed as $1/\sqrt{P}$.

The theoretical average BR can be found by dividing the change in perimeter of a phenomenon when removing its inlet by the average distance between Voronoi Neighbors in the deployment. The following relationship expresses the theoretical average BR in terms of P, R (radius of the phenomenon), d (depth of the inlet), w (width of the inlet) and θ (which can be expressed as $\arcsin \frac{w}{2R}$):

$$1 - \frac{2R(\pi - \theta) + w}{2R(\pi - 1 + \cos\theta - \theta) + 2d + w}$$

Similarly, the theoretical average IA can be deduced to be:

$$\frac{R^2(\pi - \theta + \frac{1}{2}\sin 2\theta)}{R^2(\pi - \theta + \frac{1}{2}\sin 2\theta) - w(d - R + R\cos\theta)} - 1$$

C. Experiment Variables

The following phenomenon settings will affect the performance of SpatialZip.

1) Depth of rectangular inlet (d): As the inlet gets deeper, if SpatialZip completely zips it up, we can expect the boundary reduction to increase. However, if the inlet is too deep, SpatialZip may get stuck at a local minimum where the inlet is not completely zipped up yet. Therefore, we determine at what depth SpatialZip's performance deteriorates.

2) Density of sensors (ρ): The density of sensors will play a role in determining the performance of our algorithm even though our average theoretical solution says otherwise. With a higher sensor density, more NBSs need to be promoted before the inlet can be zipped up and may therefore cause SpatialZip to terminate before the inlet is completely closed. The density of the sensors can be changed by changing the (common) probability of a cell in the grid having a sensor inside it.

V. Results

A. Simulation Setup and Test Methodology

In this section we will show that our algorithm can successfully simplify the boundary of a phenomenon being monitored by a sensor network. We will run our experiments using the evaluation parameters mentioned in the previous section; the BR and IA will be used to gauge how these phenomenon settings affect the performance of SpatialZip.

We run SpatialZip on a variety of phenomenon settings in our custom-developed simulator. We generate the location of the sensors by first placing a 100 cells by 100 cells grid over the XY plane. Each cell is both 10m in height and width and has a common probability of having a sensor placed in a random location inside the cell. From preliminary investigations, a probability of 0.3 has been selected as the default probability in order to simulate a realistic deployment and to provide meaningful results. The phenomenon being monitored is represented by a circle with a rectangular inlet cut into it. The radius of the phenomenon will be 40% of the height and width of the deployment to allow for some adjacent

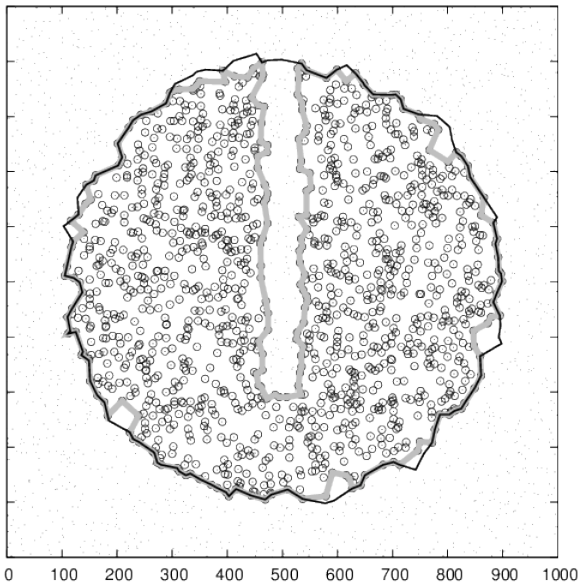

Fig. 6. The above image shows the simulation results for a 600m-deep inlet. The circles represent the sensors that are detecting the phenomenon. The thick gray line shows the boundary initially detected and the thin black line represents the boundary resulting from SpatialZip.

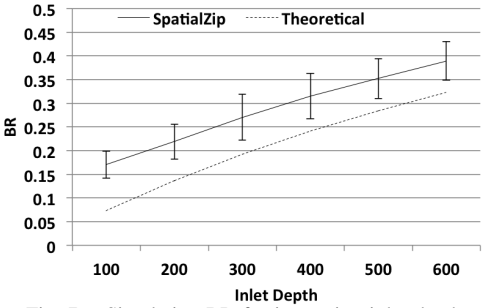

Fig. 7. Simulation BR for increasing inlet depth.

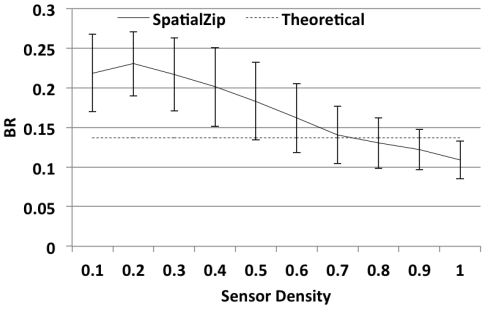

Fig. 8. Simulation BR for increasing sensor density.

cells for the ROI to grow. The inlet will be 200m deep (half of the radius) and 50m wide unless otherwise specified.

As mentioned above, we will measure the BR and IA for a variety of experimental settings. We will vary the inlet depth between 100m to 600m and the sensor density from 0.1 to 1; only one variable will be changed at a time to allow us to see their impact on SpatialZip's performance. Furthermore we will compare the boundary resulting from SpatialZip with the *theoretical average* solution discussed in the previous section. Each test is run 100 times on randomly-generated sensor deployments to obtain a representative average result, and a sense of the variation in the quality of the Zip program.

Besides comparing the performance of SpatialZip on a variety of phenomenon settings, we will also test the convergence of SpatialZip to see if the resulting boundary will change if different nodes initiate the zipping process. Since SpatialZip is a distributed algorithm, it might not always converge to the same simplified boundary meaning that its performance is dependent of the initiating node.

Finally, to convince ourselves that SpatialZip works on different types of phenomena besides circular ones, we will run SpatialZip on a square phenomenon containing a rectangular inlet. If SpatialZip works not just on circular phenomena, SpatialZip should be able to zip up the inlet in a square phenomenon just as it would in a circular one.

B. Discussion

Figure 6 shows a graphical result of our simulation. Figures 7-10 show the numerical results of our simulation.

As demonstrated by Figure 7, as the inlet gets deeper, the BR increases. SpatialZip performed as expected and closely matches the trend of the performance of the theoretical average solution. Even when the depth of the inlet was 75% of the diameter of the phenomenon, SpatialZip was able to zip up the inlet consistently. Furthermore, SpatialZip achieved a BR of 0.38 at this inlet depth, which means that the length of the boundary was reduced by almost 40%.

It is worth noting that although the number of nodes lying on the inlet increases linearly, the BR with an increasing inlet depth does not exhibit a similar behavior. This is because the resulting boundary is the same for all inlet depths and hence the average of the resulting boundary lengths should remain constant. Due to the definition of BR, if the resulting boundary length remains constant, the increase in BR should be inversely proportional to the original boundary length.

The experimental results also show that when the sensor density is above 0.2, SpatialZip's performance deteriorates (see Figure 8). This contrasts the theoretically predicted trend, which suggests that sensor density will not affect the level of zipping. This deterioration in performance is likely a result of an increasing number of nodes that need to be identified as inlets before the entire inlet can be completely zipped up. This property of SpatialZip is desirable: as the sensor density increases, the boundary becomes more refined. This means that the boundary is an accurate representation of the extent of the phenomenon and simplifying the boundary will be guaranteed to reduce accuracy substantially. If the sensor density is low, the original boundary is less accurate and hence it is not possible to tell whether the simplified boundary is more or less accurate than the original. Therefore when the sensor density is high, the boundary should not be simplified significantly.

Figure 9 to 10 shows the IA with different experiment settings. As expected, as the BR increases, the IA increases as well. As explained previously, IA is a measure of the reduction in accuracy and since a high BR results from a reduced accuracy, as BR increases, IA should increases too.

We note that there is a significant variance in every result. For example, the standard deviation of the BR in the experiment with a sensor density of 0.1 is up to 26% of the mean BR. These large variances suggests that SpatialZip's performance

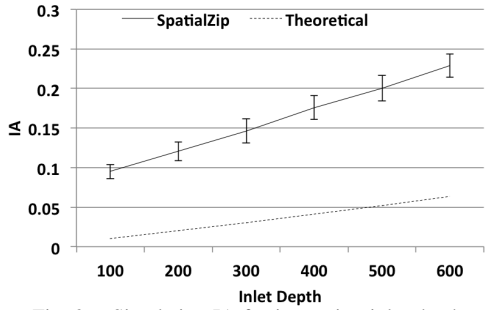

Fig. 9. Simulation IA for increasing inlet depth.

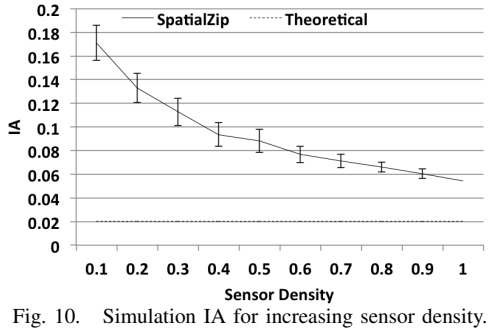

Fig. 10. Simulation IA for increasing sensor density.

heavily depends on the distribution of sensor nodes. This is expected since we are using geometric properties to identify inlets that are dependent on the node distribution.

Although we plotted the theoretical average results on the same graphs as the SpatialZip results for comparison, the values for BR and IA for the theoretical average solution cannot be compared directly with the values of BR and IA for SpatialZip. The higher BR for SpatialZip than for the theoretical case is due to the fact that SpatialZip does not produce a boundary that is as accurate as the boundary used to calculate the theoretical solution. This is substantiated by the IA for SpatialZip being significantly higher than the IA for the theoretical solution. Whether the SpatialZip boundary or the average theoretical boundary is more desirable is hard to decide. It is not possible to definitively state if it is worth sacrificing accuracy for the resulting energy savings, since this depends on the application. This evaluation parameter should be determined per application by the user to see if the accuracy trade off is worth the energy savings.

To test the convergence of SpatialZip, we initiated the zipping process from every NBS once to see if the boundaries are identical. After doing this on multiple experimental setups, we conclude that SpatialZip does converge to the same boundary regardless of which node initiates SpatialZip. Consequently, we do not need to identify which node to initiate the zipping process with to get the optimal performance.

When we ran SpatialZip on a square shaped phenomenon containing a rectangular inlet, SpatialZip was able to zip up the inlet. Therefore, we are confident that SpatialZip works not just on circular phenomena.

VI. Conclusion

Methods for conserving energy in wireless sensor networks are very important. Boundary simplification is one such method used to achieve the conservation of energy. Existing boundary simplification algorithms are centralized and therefore aren't energy efficient. In this paper, we presented an algorithm called SpatialZip for efficiently simplifying the boundary of a phenomenon in a wireless sensor network. The algorithm achieved this by repeatedly identifying local inlets and zipping them up.

Our experiments justify the design of SpatialZip. We found that SpatialZip significantly reduces the length of the boundary, thus reducing the battery consumed in transmitting the boundary to a sink. Furthermore, it is shown that if the initial boundary is highly refined, SpatialZip will not change the boundary significantly thereby maintaining a true representation of the boundary.

Empirically, SpatialZip demonstrated that it converges to the same boundary no matter which sensor initiates the zipping process. As a result, it is not required to identify inlets to initiate zipping with to provide the optimal amount of energy savings since the no matter which sensor starts this process, the resulting boundary will be the same.

There are several research directions for future work. As mentioned above, we aim to investigate whether SpatialZip works with phenomena with holes, whose boundaries are disconnected. Furthermore, additional geometric criteria for identifying inlets can be investigated. For example, an additional criteria may leverage on the M2 Advantage described in [13] to use non-local information attained without additional energy expenditure to determine if a NBS is a inlet or not.

References

[1] S. Duttagupta, K. Ramamritham, and P. Ramanathan, "Distributed boundary estimation using sensor networks," *IEEE Int. Conf. on Mobile Adhoc and Sensor Systems*, vol. 0, pp. 316–325, 2006.

[2] C. Zhang, Y. Zhang, and Y. Fang, "Localized coverage boundary detection for wireless sensor networks," in *Proc. of the 3rd Int. Conf. on Quality of Service in Heterogeneous Wired/Wireless Networks*, 2006.

[3] J.-H. Kim, K.-B. Kim, S. H. Chauhdary, W. Yang, and M.-S. Park, "Democo: Energy-efficient detection and monitoring for continuous objects in wireless sensor networks," *IEICE Trans.*, vol. 91-B, no. 11, pp. 3648–3656, 2008.

[4] R. B. McMaster, "Automated line generalization," *Cartographica: The Int. Journal for Geographic Inf. and Geovisualization*, vol. 24, pp. 74–111, 1987.

[5] Y. Song, J. Shen, and D. Yoon, "A bipolar model for region simplification," in *Proc. of the Int. Conf. on Comput. Graph. and Virtual Reality*, June 2006, pp. 147–160.

[6] P. van der Poorten, S. Zhou, and C. Jones, "Topologically-consistent map generalisation procedures and multi-scale spatial databases," in *Geographic Inf. Science*, 2002, vol. 2478, pp. 209–227.

[7] J. M. Hellerstein, W. Hong, S. Madden, and K. Stanek, "Beyond average: Toward sophisticated sensing with queries," in *Proc. of the Workshop on Inf. Process. in Sensor Networks*, 2003, pp. 63–79.

[8] S. Gandhi, J. Hershberger, and S. Suri, "Approximate isocontours and spatial summaries for sensor networks," in *Proc. of the Int. Conf. on Inf. Process. in Sensor Networks*, 2007, pp. 400–409.

[9] F. Aurenhammer, "Voronoi diagrams–a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, pp. 345–405, September 1991.

[10] W. Alsalih, K. Islam, Y. Núñez Rodríguez, and H. Xiao, "Distributed voronoi diagram computation in wireless sensor networks," in *Proc. of the Annual Symp. on Parallelism in Algorithms and Archit.*, 2008, pp. 364–364.

[11] C. B. Barber, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Trans. Math. Softw.*, vol. 22, pp. 469–483, December 1996.

[12] O. Tonguz and G. Ferrari, "Is the number of neighbors in ad hoc wireless networks a good indicator of connectivity?" in *Communications, Int. Zurich Seminar on*, 2004, pp. 40 – 43.

[13] M. Umer, E. Tanin, and L. Kulik, "Opportunistic sampling in wireless sensor networks," in *Proc. of the ACM SIGSPATIAL Int. Conf. on Adv. in Geographic Inf. Systems*, 2009, pp. 492–495.