Efficient Algorithms for Distributed Detection of Holes and Boundaries in Wireless Networks

Dennis Schieferdecker, Markus Völker, and Dorothea Wagner

Karlsruhe Institute of Technology (KIT), Germany
{schieferdecker,m.voelker,dorothea.wagner}@kit.edu

Abstract. We propose two novel algorithms for distributed and location-free boundary recognition in wireless sensor networks. Both approaches enable a node to decide autonomously whether it is a boundary node, based solely on connectivity information of a small neighborhood. This makes our algorithms highly applicable for dynamic networks where nodes can move or become inoperative.

We compare our algorithms qualitatively and quantitatively with several previous approaches. In extensive simulations, we consider various models and scenarios. Although our algorithms use less information than most other approaches, they produce significantly better results. They are very robust against variations in node degree and do not rely on simplified assumptions of the communication model. Moreover, they are much easier to implement on real sensor nodes than most existing approaches.

1 Introduction

Wireless sensor networks have become a prominent research topic in recent years. Their unique structure and limitations provide new and fascinating challenges. A sensor network consists of a union of small nodes that are equipped with sensing, communication, and processing capabilities. The nodes usually only have a limited view of the network. Therefore, distributed algorithms that work on local information are best suited for the emerging tasks in these environments.

Many applications in sensor networks require a certain knowledge of the underlying network topology, especially of holes and boundaries. Examples are intrusion detection, data gathering [16], mundane services like efficient routing within the network [5], or event detection [4]. In many situations, holes can also be considered as indicators for insufficient coverage or connectivity. Especially in dynamic settings, where nodes can run out of power, fail, or move, an automatic detection of holes and boundaries is inevitable.

For this reason, many boundary recognition algorithms have been developed previously. However, most of them have certain disadvantages. Some rely on oversimplified assumptions concerning the communication model or on knowledge about absolute or relative node positions, which is usually not available in large-scale sensor networks. Other algorithms are not distributed or require information exchange over long distances, so they do not scale well with network

P.M. Pardalos and S. Rebennack (Eds.): SEA 2011, LNCS 6630, pp. 388–399, 2011.

size. And those algorithms that solely work locally usually produce many misclassifications. Furthermore, many of the existing algorithms are too complex for an actual implementation on real sensor nodes. So there is still demand for simple and efficient algorithms for boundary recognition.

Related Work. Since there is a wide range of applications that require boundary detection, there is an equally large number of approaches to detect holes. Based on the underlying ideas, they can be classified roughly into three categories.

Geometrical approaches use information about node positions, distances between nodes, or angular relationships. Accordingly, these approaches are limited to situations where GPS devices or similar equipment are available. Unfortunately, in many realistic scenarios this is not the case. Examples for geometrical approaches are Fang *et al.* [5], Martincic *et al.* [12], and Deogun *et al.* [3].

Statistical approaches try to recognize boundary nodes by low node degree or similar statistical properties. As long as nodes are smoothly distributed, this works quite well. However, as soon as node degrees fluctuate noticeably, most statistical approaches produce many misclassifications. Besides, these algorithms often require unrealistic high average node degrees. Prominent statistical approaches are Fekete *et al.* [6,7], and Bi *et al.* [1].

Topological approaches concentrate on information given by the connectivity graph and try to infer boundaries from its topological structure. For example, the algorithm of Kröller *et al.* [11] works by identifying complex combinatorial structures called flowers and Funke [8] and Funke *et al.* [9] describe algorithms that construct iso-contours and check whether those contours are broken. Further examples of topological approaches are given by Wang *et al.* [16], Ghrist *et al.* [10], De Silva *et al.* [2], and Saukh *et al.* [13]. A recent distributed algorithm by Dong *et al.* [4] is especially aimed at locating small holes.

An extended version of this article has been published as technical report [14]. It includes a more comprehensive list on related work and an overview on existing classification schemes for network holes and boundaries. There, we also present a more detailed description of our algorithms and additional simulation results, as we had to condense this article significantly due to page restrictions.

2 Model Description

2.1 Network Model

A sensor network consists of nodes located in the two-dimensional plane according to some distribution. Communication links between nodes induce a *connectivity graph* $\mathcal{C}(V, E)$, with graph nodes $v \in V$ corresponding to sensor nodes and graph edges $(u, v) \in E$; $u, v \in V$ to communication links between sensor nodes. An *embedding* $p : V \to \mathbb{R}^2$ of the connectivity graph $\mathcal{C}$ assigns two-dimensional coordinates $p(v)$ to each node $v \in V$. For easier reading, distances are normalized to the maximum communication distance of the sensor nodes.

Communication model. Two communication models are considered. Both assume bidirectional communication links. In the *unit disk graph* (UDG) model, two sensor nodes $u, v \in \mathcal{C}$ can communicate with each other, i.e., there exists a communication link between them, if their distance $|p(u)p(v)|$ is at most 1. In the *quasi unit disk graph* (d-QUDG) model, sensor nodes $u, v \in \mathcal{C}$ can communicate reliably if $|p(u)p(v)| \leq d$ for a given $d \in [0, 1]$. For $|p(u)p(v)| > 1$ communication is impossible. In between, communication may or may not be possible.

Node distribution. Two node distribution strategies are considered. Using *perturbed grid placement*, nodes are placed on a grid with grid spacing 0.5 and translated by a uniform random offset taken from $[0, 0.5]$ in both dimensions. Using *random placement*, nodes are placed uniformly at random on the plane.

2.2 Hole and Boundary Model

For evaluation, well-defined hole and boundary definitions are required. The definitions introduced by previous contributions are often too complicated or not extensive enough. Several approaches define holes but do not specify which nodes are considered boundary nodes, while others classify boundary nodes without taking into account their positions. In our work, we take a very practical look at what to label as holes. In short, we call large areas with no communication links crossing them holes and nodes on the borders of these areas boundary nodes.

Hole Definition. Some previous definitions are based on abstract topological definitions. In contrast, we think that the hole definition should be based on the embedding of the actual sensor network. Thus, for evaluation only, we take advantage of the true node positions.

All faces induced by the edges of the *embedded connectivity graph* $p(\mathcal{C})$ are hole candidates. Similarly to [11], we define holes to be faces of $p(\mathcal{C})$ with a circumference of at least h_{min}. Fig. 1(left) depicts a hole according to our definition. Take note that the exterior of the network is an infinite face. Thus, it is regarded as a hole for the purpose of computation and evaluation.

Boundary Node Definition. As seen in Fig. 1(left), hole borders and node locations do not have to align. Thus, there exists the problem which nodes to classify as boundary nodes. For example, it can be argued whether nodes A and B should be boundary nodes or not. To alleviate this problem, we classify nodes into three categories:

- *Mandatory Boundary Nodes.* Nodes that lie exactly on the hole border are boundary nodes.
- *Optional Boundary Nodes.* Nodes within maximum communication distance of a mandatory node can be called boundary nodes but do not have to be.
- *Interior Nodes.* All other nodes must not be classified as boundary nodes.

The resulting node classification is shown in Fig. 1(right). Mandatory boundary nodes form thin bands around holes, interrupted by structures like for nodes

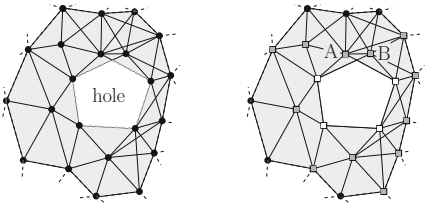


Fig. 1. (left) Hole Definition: Border as dashed line. (right) Boundary Node Classification: Mandatory nodes (white boxes), optional nodes (gray boxes), interior nodes (black circles).

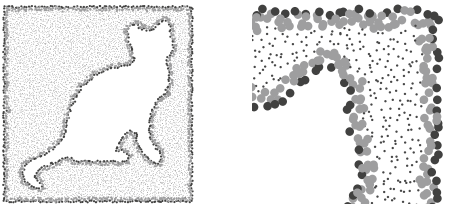


Fig. 2. Node Classification. Border outline of mandatory nodes (large, black), halo of optional nodes (large, gray), interior nodes (small, black). Full network and magnified upper right corner.

A and B before. Together with the optional boundary nodes, they form a halo around each hole. Any point within the halo is at most one maximum communication distance away from the border of the enclosed hole. A sample classification is depicted in Fig. 2.

3 Multidimensional Scaling Boundary Recognition (MDS-BR)

Both of our algorithms work in a distributed fashion and only require local connectivity information. Each node independently decides whether it is a boundary node or an interior node, solely using information from a small neighborhood.

Our first algorithm is a geometrical approach at its core. But instead of using real node coordinates, which are usually not known in sensor networks, it uses multidimensional scaling (MDS) [15] to compute virtual coordinates. Two angular conditions are then tested to classify a node, followed by a refinement step after all nodes have classified themselves. Subsequently, the base algorithm and a refinement step of MDS-BR are described. We refer to [14] for an analysis of runtime, message complexity, and possible variants of MDS-BR.

Base Algorithm. Each node performs the following base algorithm to decide independently whether to classify itself as a boundary node. At first, each node u gathers its 2-hop neighborhood N_u^2 and computes a two-dimensional embedding of $N_u^2 \cup \{u\}$, using hop distances to approximate true distances between nodes. Then, using these virtual locations, node u declares itself to be a boundary node if two conditions are fulfilled. First, the maximum opening angle α between two subsequent neighbors v, w of u in circular order and u must be larger than a threshold α_{min} as depicted in Fig. 3(a). This primary condition models the observation that boundary nodes exhibit a large gap in their neighborhood compared to interior nodes, which are usually completely surrounded by other nodes. Secondly, neighbors v, w of u must not have common neighbors

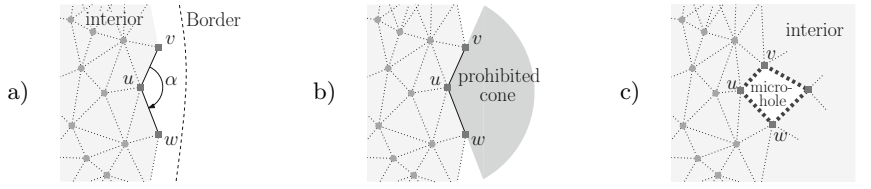


Fig. 3. MDS-BR Conditions. (a) Opening angle. (b) Prohibited cone. (c) Micro-holes.

other than u in the cone opened by (uv) and (uw). This condition is exemplified in Fig. 3(b). It filters micro-holes framed by 4 nodes with a circumference of at most 4 maximum communication distances as seen in Fig. 3(c). If such holes are to be detected, the condition can be omitted.

Both conditions only require angular information. Thus, any embedding algorithm yielding realistic angles between nodes is sufficient – we are not limited to MDS. Furthermore, we do not need complex embedding techniques to compensate for problems occurring in large graphs such as drifting or foldings since we only embed very small graphs. In particular, we only compute embeddings of 2-hop neighborhoods around each node, i.e., graphs of diameter 4 or less.

Refinement. The base algorithm already yields good results as shown in Fig. 4(a). But it retains some "noise" due to detecting boundary nodes around small holes one might not be interested in and due to some misclassifications. If desired, a refinement step can be used to remove most of these artifacts as seen in Fig. 4(b).

The refinement is performed distributed on the current set of boundary nodes. First, each boundary node u gathers its r_{min}-hop neighborhood $\tilde{N}_u^{r_{min}}$ of nodes marked as boundary nodes by the base algorithm, where r_{min} is a free parameter. Then, u verifies if there exists a shortest path of at least r_{min} hops in $\tilde{N}_u^{r_{min}} \cup \{u\}$ that contains u. If no such path exists, u classifies itself as interior node. This approach removes boundary nodes that are not part of a larger boundary structure, with r_{min} specifying the desired size of the structure. Note that only connectivity information is required for the refinement.

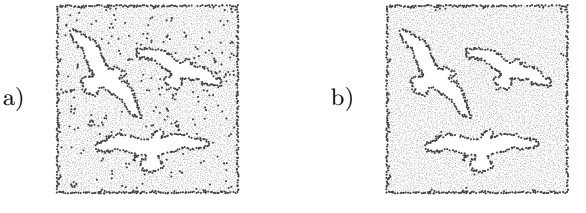


Fig. 4. Classification results of MDS-BR on a sample network. Boundary nodes marked as larger black nodes. (a) Results of the base algorithm. (b) Results after refinement.

4 Enclosing Circle Boundary Recognition (EC-BR)

Our second algorithm, EC-BR, allows to detect if a node is surrounded by other nodes without having to reconstruct node positions. Every node considers only nodes that are exactly two hops away. For a node u, we denote the corresponding subgraph as $G_u^{2\setminus 1} = (N_u^{2\setminus 1}, E_u^{2\setminus 1})$. Based on connectivity information in $G_u^{2\setminus 1}$, the node tries to decide if it is surrounded by a closed path C. If such a path exists, one can be sure that the node is not a boundary node (cf. Fig. 5(a)-(c)).

Given node positions, it would be easy to decide if an enclosing circle exists. However, we do not have this information and we do not want to reconstruct node positions in order to save computation time. So, how can we distinguish enclosing circles as the one in Fig. 5(b) and non-enclosing circles such as the one in Fig. 5(c)? Circle length is no sufficient criterion as both circles have the same length and only the first one is enclosing. Fortunately, there is a structural difference between both types of circles: in the first case, for each pair v, w of circle nodes, the shortest path between them using only circle edges is also a shortest path between them in $G_u^{2\setminus 1}$. Now we try to find a preferably long circle with this property. This can be achieved using a modified breadth-first search. The corresponding search tree for $G_u^{2\setminus 1}$ of Fig. 5(b) is depicted in Fig. 5(d). We start from a random node z in $G_u^{2\setminus 1}$ with maximum degree. In every step, we maintain shortest path lengths for all pairs of visited nodes. When a new edge is traversed, either a new node is visited or a previously encountered node is revisited. In the first case we set the shortest path distances between the already visited nodes and the new node. This can be done efficiently, as all distances can be directly inferred from the distances to the parent node. In the second case we found a new circle in $G_u^{2\setminus 1}$. The length of the circle is the current shortest path between the endpoints v and w of the traversed edge $e = (v, w)$ plus one. Subsequently, we update the shortest path information of all visited nodes. During search, we keep track of the maximum length of a circle encountered so far. Depending on this maximum length, the considered node is either classified as a boundary node or as an inner node. This enclosing circle detection can be achieved with time complexity $O(|E_u^{2\setminus 1}|)$. See [14] for further details.

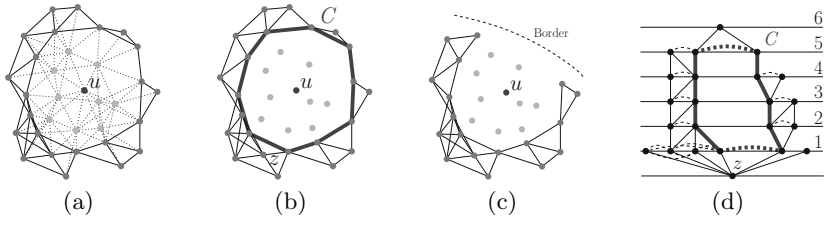


Fig. 5. Basic Idea of EC-BR. (a) 2-hop neighborhood of u. (b) Enclosing circle C. (c) Boundary node without enclosing circle. (d) Modified breadth-first search.

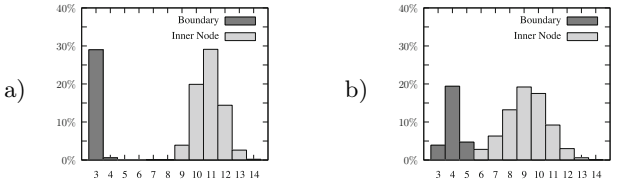


Fig. 6. Distribution of Maximum Circle Lengths. (a) UDG. (b) QUDG.

Fig. 6 depicts histograms of maximum circle lengths in our simulations with networks based on unit disk graphs and quasi unit disk graphs. There are apparently two very well defined peaks, corresponding to nodes with and without enclosing circles. Based on this distribution, we classify all nodes that have a maximum circle with length of at least 6 as inner nodes and all other nodes as boundary nodes. Our simulations indicate that this statistical classification works extremely well for both UDGs and QUDGs. Later on, we will see how good this correlates with being in the interior or on the boundary of the network.

It is also noteworthy that this kind of classification is extremely robust to variations in node degree: it does not matter whether $N_u^{2\setminus 1}$ consists of a small number of nodes or hundreds of nodes. The classification stays the same, as long as we assume that the node density is sufficiently high so that inner nodes are actually surrounded by other nodes.

Fig. 7 shows an example of a classification with EC-BR. In comparison with MDS-BR, the recognized boundaries are broader and EC-BR also detects small-scale holes which occur in areas of low node density.

Refinement. If one is only interested in large scale boundaries, EC-BR can be extended with a simple refinement. The key insight is that nodes which lie immediately next to the hole are surrounded by nodes that are marked as boundary nodes and by the hole itself. So a boundary candidate simply has to check whether a certain percentage γ of its immediate neighbors are currently classified as boundary nodes. If this is not the case, the node changes its classification to being an interior node. The effect of this simple strategy with $\gamma = 100\%$ is depicted in Fig. 7(c). Apparently, all nodes but the ones near large-scale holes are now classified as inner nodes and the boundary is very precise.

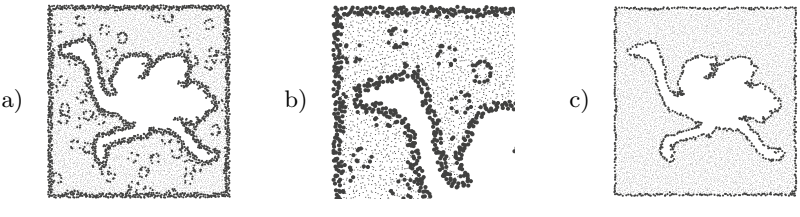


Fig. 7. Classification of EC-BR. (a,b) Before refinement. (c) After refinement.

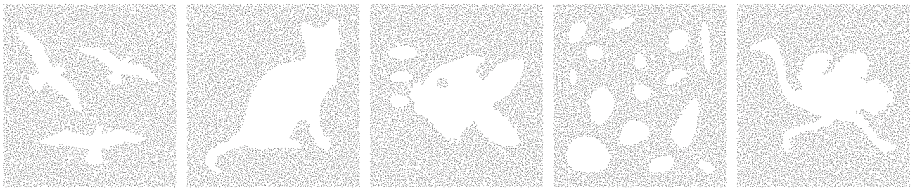


Fig. 8. Hole Patterns. Node distributions with perturbed grid placement and $d_{avg} = 12$.

5 Simulations

5.1 Simulation Setup

Network Layout. We generate network layouts by iteratively placing nodes on an area of 50×50 maximum communication distances according to one of the distribution strategies described in Section 2: perturbed grid placement (pg) or random placement (rp). After each node placement, communication links are added according to the UDG or QUDG model. Nodes are added until an average node degree d_{avg} is reached. To generate holes, we apply hole patterns such as the ones in Fig. 8. Our default layout uses perturbed grid placement, the UDG model and average node degree $d_{avg} = 12$.

Considered Algorithms. We compare the performance of our approaches EC-BR and MDS-BR to three well-known boundary recognition algorithms: The algorithm by Fekete *et al.* [7] (labelled Fekete04) and the centralized and distributed algorithms by Funke [8] and Funke *et al.* [9] (labeled Funke05 and Funke06, respectively). In addition, we show qualitative comparisons of these algorithms and the algorithm by Wang *et al.* [16] in Section 5.5. We apply our own implementation of these algorithms according to their description in the respective publications. For MDS-BR, $\alpha_{min} = 90°$ and $r_{min} = 3$ are used throughout the simulations. For the refinement of EC-BR, $\gamma = 100\%$ is used if not stated otherwise. Additional details on parameter selection are given in [14].

Measurement Procedure. Each setup is evaluated 100 times for each hole pattern in Fig. 8. Faces with circumference $h_{min} \geq 4$ are considered to be holes, e.g. a square of edge length 1 with no communication link crossing it. The analysis lists mean misclassification ratios (false negatives) in percent. For optional boundary nodes, we give the percentage of nodes classified as interior nodes.

5.2 Network Density

First, we consider the performance of all algorithms on networks with different average node degrees d_{avg}. Table 1 shows the percentage of misclassifications for mandatory boundary nodes and interior nodes with increasing d_{avg}. Results for optional boundary nodes state the percentage classified as interior nodes.

Table 1. Misclassification ratios (false negatives) in percent for average node degrees between 9 and 21 (resp. classification as interior nodes for optional boundary nodes)

	Mandatory					**Optional**					**Interior**				
	9	12	15	18	21	9	12	15	18	21	9	12	15	18	21
EC-BR	2.1	**0.0**	**0.0**	**0.0**	**0.0**	**0.3**	**0.1**	**0.2**	**0.2**	**0.3**	54.8	7.5	3.8	2.2	1.6
EC-BR Ref	4.4	0.4	0.6	1.0	1.3	51.2	80.4	81.3	82.8	84.3	**7.1**	**0.0**	**0.0**	**0.0**	**0.0**
MDS-BR	**1.9**	2.9	3.5	3.8	3.9	68.0	79.8	79.8	79.8	80.0	19.0	0.7	0.3	0.1	**0.0**
Fekete04	34.7	14.2	6.7	3.4	1.9	83.2	80.3	69.0	63.5	64.6	9.8	3.5	7.2	6.9	2.5
Funke05	16.6	6.3	5.7	5.1	5.0	61.5	59.5	55.4	52.5	50.6	21.7	3.5	2.0	1.3	0.9
Funke06	39.7	13.8	16.6	18.9	20.9	80.6	70.7	71.9	72.5	73.2	13.0	3.4	1.4	0.6	0.3

EC-BR with refinement and MDS-BR both classify almost all interior nodes correctly, except for the smallest node degree. But in this extreme scenario, all algorithms start having problems as small holes arise due to overall sparse connectivity. The classification of mandatory boundary nodes is also excellent. Here, EC-BR dominates all other algorithms. The performance of MDS-BR fluctuates only slightly over the various node degrees.

The numbers for optional boundary nodes state how many nodes within maximum communication distance of a hole are not classified as boundary nodes. Here, the results for EC-BR are particularly interesting. Before refinement, the algorithm classifies almost all of the optional nodes as boundary nodes while still providing a strict separation to the interior nodes.

A more detailed analysis shows that a lot of small-scale holes emerge in networks with average node degree of 10 or less. Accordingly, a different boundary definition and adjusted algorithms might be more appropriate for such networks. For further details we refer to our extended technical report [14].

5.3 Random Placement vs. Perturbed Grid

In this section we examine the performance when random placement is used instead of perturbed grid placement. Table 2 compares the performance of all algorithms for both placement strategies and $d_{avg} = 15$. The performance of the existing algorithms decreases dramatically compared to perturbed grid placement. In contrast, the results of the new algorithms only decrease noteworthy for interior nodes. For mandatory boundary nodes they remain roughly constant.

We also compare the influence of the network density when using random node placement. Table 3 shows the classification results for mandatory and interior nodes. Overall, our approaches dominate the other algorithms for both, mandatory boundary nodes and interior nodes. For sparse networks, we see an increased misclassification of interior nodes. This is partly caused by recognizing very small holes.

5.4 Beyond Unit Disk Graphs

Unit disk graphs are frequently used for theoretical analyses and in simulations. They are motivated by the assumption that each node has a fixed transmission range. However, under realistic conditions the transmission range depends on

Table 2. Misclassifications for random (rp) and perturbed grid placement (pg)

	Mandatory		Interior	
	rp	pg	rp	pg
EC-BR	**2.0**	**0.0**	48.7	3.8
EC-BR Ref	4.0	0.6	**4.1**	**0.0**
MDS-BR	5.2	3.5	12.2	0.3
Fekete04	26.2	6.7	13.0	7.2
Funke05	15.5	5.7	16.1	2.0
Funke06	45.8	16.6	7.9	1.4

Table 3. Misclassifications dependent on average node degree for random placement

	Mandatory			Interior		
	15	20	25	15	20	25
EC-BR	**2.0**	**1.6**	**0.5**	48.7	25.9	11.3
EC-BR Ref	4.0	3.1	1.9	**4.1**	**0.5**	**0.1**
MDS-BR	3.8	5.2	5.8	13.4	5.2	1.7
Fekete04	26.2	13.7	7.7	13.0	13.9	12.3
Funke05	15.5	9.4	6.7	16.1	8.1	3.6
Funke06	45.8	29.1	26.4	7.9	6.1	2.8

unpredictable effects such as interference or signal reflections. We now evaluate the algorithms in a more realistic context by representing uncertainties with the d-QUDG model.

Table 4 shows the performance for average node degrees 12 and 15 on 0.75-QUDG networks. The increased error rate of MDS-BR occurs because the base algorithm produces a candidate set which is not necessarily connected. Thus, the refinement classifies many correct boundary candidates as interior nodes since the connected substructures are not large enough. Fekete04, Funke05 and Funke06 yield even higher error rates and perform significantly worse than on UDGs. For QUDGs, we use $\gamma = 70\%$ for the refinement of EC-BR because in QUDGs mandatory boundary nodes are not necessarily completely surrounded by boundary candidates and holes. This value of γ works equally well for UDGs, only producing slightly broader boundaries than $\gamma = 100\%$. EC-BR with this refinement outperforms all other approaches significantly. In Table 5, we go a step further and compare 0.25-QUDGs with 0.75-QUDGs. This implies a very high level of uncertainty. As expected, all algorithms produce more misclassifications. Again, EC-BR with refinement outperforms the other algorithms easily.

5.5 Visual Comparison

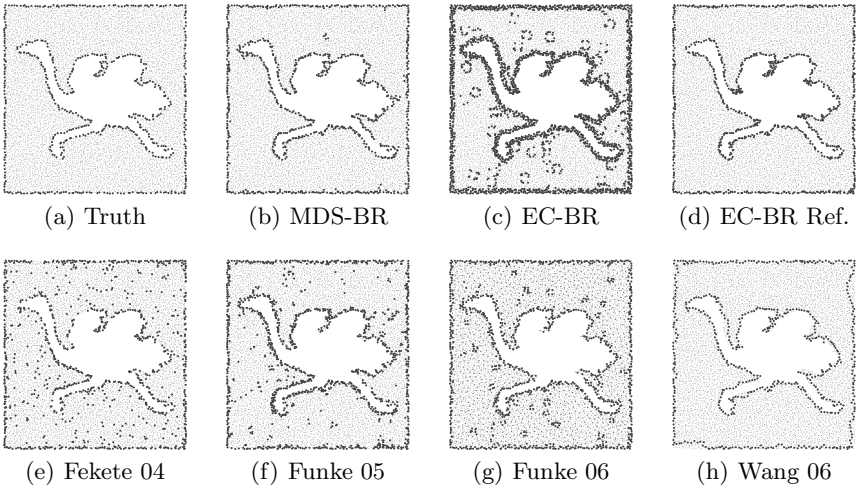
We now present a visual comparison of the results of the considered algorithms in Fig. 9. Both, EC-BR with refinement and MDS-BR return thin outlines of the inner and outer border with almost no artifacts. In contrast, Funke05 returns a broader outline with more noise. The results of Fekete04 show many artifacts and a lot of boundary nodes are not detected. Funke06 correctly identifies the

Table 4. Misclassifications for the 0.75-QUDG model and different node degrees

	Mandatory		Interior	
	12	15	12	15
EC-BR	**0.0**	**0.0**	28.5	7.7
EC-BR Ref	**0.0**	**0.0**	**4.9**	**0.3**
MDS-BR	8.3	11.2	8.3	1.6
Fekete04	16.9	6.9	8.8	8.9
Funke05	9.0	7.4	12.9	5.2
Funke06	15.6	15.4	12.4	3.7

Table 5. Misclassifications for 0.25- and 0.75-QUDG models with node degree 12

	Mandatory		Interior	
	0.25	0.75	0.25	0.75
EC-BR	**3.0**	**0.0**	41.5	28.5
EC-BR Ref.	12.7	**0.0**	**1.7**	**4.9**
MDS-BR	27.7	8.3	11.8	8.3
Fekete04	14.6	16.9	13.6	8.8
Funke05	11.5	9.0	17.8	12.9
Funke06	24.2	15.6	2.8	12.4



(a) Truth (b) MDS-BR (c) EC-BR (d) EC-BR Ref.

(e) Fekete 04 (f) Funke 05 (g) Funke 06 (h) Wang 06

Fig. 9. Visual comparison of several algorithms for boundary detection.

boundaries with some artifacts. Similar to EC-BR, the apparent noise is caused by small holes which are surrounded by these marked nodes. We also present classification results of the global algorithm by Wang *et al.* [16]. It yields closed boundary cycles without artifacts, but due to its nature, marked boundaries are not always at the true border but shifted inwards. Hence, we did not include the algorithm into our analysis, as it would result in unfairly poor ratings.

6 Conclusion

We proposed two novel *distributed* algorithms for *location-free* boundary recognition. Both of them only depend on *connectivity information* of small *local neighborhoods*, at most 3 hops for MDS-BR and only 2 hops for EC-BR. Their *low communication overhead* makes both algorithms excellent choices for boundary recognition in large-scale sensor networks and in dynamic scenarios.

We showed in extensive simulations that both algorithms are very robust to different network densities, communication models, and node distributions. Despite their simplicity and low communication overhead, they outperformed the other considered approaches significantly. Additionally, they have much lower computational complexity than most existing approaches.

Acknowledgments

This work was supported by the German Research Foundation (DFG) within the Research Training Group GRK 1194 "Self-organizing Sensor-Actuator-Networks". We would like to thank Bastian Katz for useful discussions, as well as Yue Wang and Olga Saukh for providing us with the topologies of their respective networks.

References

1. Bi, K., Tu, K., Gu, N., Dong, W.L., Liu, X.: Topological hole detection in sensor networks with cooperative neighbors. In: International Conference on Systems and Networks Communication (ICSNC 2006), pp. 31–35 (2006)
2. De Silva, V., Ghrist, R.: Coordinate-free coverage in sensor networks with controlled boundaries via homology. International Journal of Robotics Research 25(12), 1205–1222 (2006)
3. Deogun, J.S., Das, S., Hamza, H.S., Goddard, S.: An algorithm for boundary discovery in wireless sensor networks. In: Bader, D.A., Parashar, M., Sridhar, V., Prasanna, V.K. (eds.) HiPC 2005. LNCS, vol. 3769, pp. 343–352. Springer, Heidelberg (2005)
4. Dong, D., Liu, Y., Liao, X.: Fine-grained boundary recognition in wireless ad hoc and sensor networks by topological methods. In: MobiHoc 2009: Proceedings of the Tenth ACM International Symposium on Mobile ad Hoc Networking and Computing, pp. 135–144. ACM, New York (2009)
5. Fang, Q., Gao, J., Guibas, L.J.: Locating and bypassing routing holes in sensor networks. In: INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies (2004)
6. Fekete, S.P., Kaufmann, M., Kröller, A., Lehmann, N.: A new approach for boundary recognition in geometric sensor networks. In: Proceedings 17th Canadian Conference on Computational Geometry, pp. 82–85 (2005)
7. Fekete, S.P., Kröller, A., Pfisterer, D., Fischer, S., Buschmann, C.: Neighborhood-based topology recognition in sensor networks. In: Nikoletseas, S.E., Rolim, J.D.P. (eds.) ALGOSENSORS 2004. LNCS, vol. 3121, pp. 123–136. Springer, Heidelberg (2004)
8. Funke, S.: Topological hole detection in wireless sensor networks and its applications. In: DIALM-POMC 2005: Proceedings of the 2005 Joint Workshop on Foundations of Mobile Computing, pp. 44–53. ACM, USA (2005)
9. Funke, S., Klein, C.: Hole detection or: how much geometry hides in connectivity? In: SCG 2006: Proceedings of the Twenty-Second Annual Symposium on Computational Geometry, pp. 377–385. ACM, USA (2006)
10. Ghrist, R., Muhammad, A.: Coverage and hole-detection in sensor networks via homology. In: IPSN 2005: Proceedings of the 4th International Symposium on Information Processing in Sensor Networks, pp. 254–260. IEEE Press, USA (2005)
11. Kröller, A., Fekete, S.P., Pfisterer, D., Fischer, S.: Deterministic boundary recognition and topology extraction for large sensor networks. In: 17th Annual ACM–SIAM Symposium on Discrete Algorithms (SODA 2006), pp. 1000–1009 (2006)
12. Martincic, F., Schwiebert, L.: Distributed perimeter detection in wireless sensor networks. Tech. Rep. WSU-CSC-NEWS/03-TR03, Wayne State University (2004)
13. Saukh, O., Sauter, R., Gauger, M., Marrón, P.J.: On boundary recognition without location information in wireless sensor networks. ACM Transactions on Sensor Networks (TOSN) 6(3), 1–35 (2010)
14. Schieferdecker, D., Völker, M., Wagner, D.: Efficient algorithms for distributed detection of holes and boundaries in wireless networks. Tech. Rep. 2011-08, Karlsruhe Institute of Technology (2011)
15. Torgerson, W.S.: Multidimensional Scaling: I. Theory and Method. Psychometrika 17(4), 401–419 (1952)
16. Wang, Y., Gao, J., Mitchell, J.S.: Boundary recognition in sensor networks by topological methods. In: MobiCom 2006: 12th Annual International Conference on Mobile Computing and Networking, pp. 122–133. ACM, USA (2006)