Distributed Obstacle Localization in Large Wireless Sensor Networks

Frank Reichenbach, Ralf Salomon, Dirk Timmermann
Institute of Applied Microelectronics and Computer Engineering, University of Rostock
Richard-Wagner Street 31
18119 Rostock-Warnemuende, Germany
{frank.reichenbach,ralf.salomon,dirk.timmermann}@uni-rostock.de

ABSTRACT

Obstacles are but pleasing for many aspects of large real-world sensor networks. Among other things, the presence of obstacles distort the sensor node localization process and might lead to costly routing because of unnecessary detours and/or dead ends. In order to relieve these problems, this paper proposes a distributed obstacle localization algorithm, called DOL for short, in which the sensor nodes interact with each other mostly locally. The proposed algorithm is very resource and communication efficient in that all sensor nodes send only a small number of additional messages. Finally, the sensor network fine tunes the employed routing algorithm.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Algorithms, Design, Theory

Keywords

obstacle localization, local interaction rules, wireless sensor networks, routing, multi-hopping

1. INTRODUCTION

It is commonly agreed on that sensor networks consist of a (very) large number of similar (or equivalent) network nodes [1]. These network nodes, or simply called *nodes*, perform some useful tasks, such as monitoring various environmental conditions, detecting critical states, observing toxic areas, etc. The network nodes are furthermore equipped with some radio communication capabilities such that they constitute a *wireless* sensor network. By enabling multi-hopping [2] and employing an appropriate routing algorithm, the nodes are able to send their sensor data to designated data sinks at certain time intervals.

IWCMC'06, July 3–6, 2006, Vancouver, British Columbia, Canada.

Since the number of nodes in a large wireless sensor network is way too high to be set up by hand, the nodes are typically distributed in a stochastic process[1]. After their deployment, the nodes do not have any information about their own position as well as their neighborhood. Thus, the employment, configuration, and maintenance of large real-world wireless sensor networks require a high degree of autonomy and self-organization in order to function properly.

Typical application areas of wireless sensor networks include flood prevention, forest fire detection, and disaster surveillance. As opposed to laboratory setups and feasibility studies, *real-world* application terrains are normally populated with obstacles of all sorts, including hills, valleys, trees, walls, houses, cars, etc. These obstacles, as Section 2 discusses in more detail, may block the direct radio communication between certain network parts, which in turn may induce significant problems on the network's self-organization process.

Due to the importance of real-world applications, Section 3 analyzes how radio waves propagate in the neighborhood of obstacles, and how the presence of obstacles changes the visibility of surrounding network parts. This analysis results in a metric that can serve as a basis of a modified distributed localization algorithm, called DOL for short.

Section 4 presents a detailed description of the DOL algorithm and in so doing, emphasizes on the incorporated modifications. By slightly increasing the communication overhead, the nodes are able to detect nearby obstacles. Furthermore, the nodes locate (most) of the obstacle's corners, which are subsequently communicated through the network infrastructure.

Section 5 presents some simulation results in order to illustrates the algorithm's main mechanisms. Then, Section 6 discusses how the derived obstacle position(s) can be used to improve the sensor nodes' localization procedure and to allow for energy-efficient routing. Finally, Section 7 concludes with a brief discussion.

2. BACKGROUND: NETWORK SELF-ORGANIZATION AND WIRELESS COMMUNICATION

Deployment means that a (very) large number of (sensor) nodes is distributed in a potentially inaccessible area. After that, the nodes do not know anything about their local neighborhood nor do they know their (approximate)

[1]Please take into account that the application terrain might also be inaccessible for any reason.

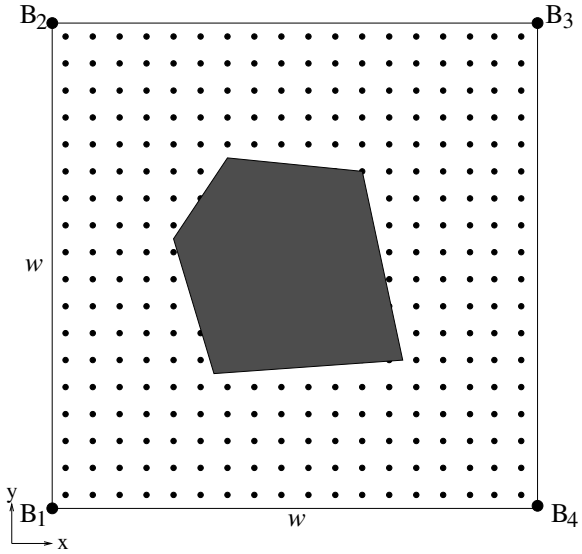

Figure 1: In a typical network topology, a small number of beacons B_i is distributed over a regular grid, with w denoting the distance between neighbors. Within each square, a relatively large number of sensor nodes is regularly distributed.

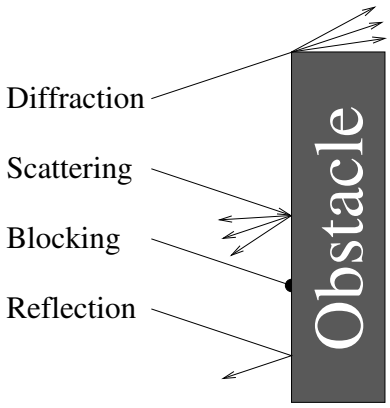

Figure 2: This figure shows how obstacles cause diffraction, scattering, blocking, and reflection, which all have rather disadvantageous side effects in wireless sensor networks.

position. It is theoretically conceivable to equip every node with a GPS module. However, this would result in huge costs, which is unacceptable for practical reasons. Therefore, practical sensor networks employ a small number of specialized network nodes, called *beacons* B, which do know their exact positions (e.g., by means of GPS), do have extended energy resources and thus enhanced communication capabilities.

Figure 1 presents a generic example in which w denotes the distance between neighboring beacons and μ denotes the density with which the nodes are distributed in an area of unit size. Thus, the model distributes $n_w = \mu w^2$ sensor nodes in the area depicted in Figure 1.

As part of the network's self-organization process, all nodes try to determine at least a position estimate. For this purposes, the pertinent literature offers a large number of algorithms [3, 4, 5] to mention but a few. The estimation accuracy ranges from quite precise to rather rough, since it depends on various factors, such as the network topology, the number of beacons, the employed algorithms, the acceptable communication overhead, etc. The remainder of this paper adopts the network topology presented in Figure 1, frequently used by others [5, 3, 6, 7, 4].

As has been outlined in the introduction, a sensor network normally employs multi-hopping [2] and routing in order to allow the (sensor) nodes to transmit their data to designated data sinks. Traditional routing algorithms, such as destination-sequenced distance-vector routing [8], dynamic source routing [9], and ad-hoc on-demand distance vector routing [8], perform routing according to the shortest-distance principle. However, under the presence of obstacles, as might be the case in real-world sensor networks, packets might get lost or stuck in dead ends, since routing cannot escape from there [5].

A key reason for routing to fail in the presence of obstacles is that the latter do not allow to determine routes by using metric information. For example, two nodes with spatial positions close to each other may not be able to (directly) communicate, since an obstacle circumvents the successful radio wave transmission.

In a recent publication [10], Yamazaki and Sezaki have proposed a solution in which the concept of reference points might help to overcome this problem. However, these reference points were not set by an automatic (self-organization) process, but rather by hand or some other, not documented mechanism. Even though this routing algorithm proposes some very interesting concepts, it is unfortunately not an alternative, since autonomous self-organization is a key element in the organization of large real-world sensor networks, particularly in hostile or inaccessible environments.

For both the localization and routing algorithms mentioned above, a free line-of-sight to the surrounding beacons (as well as neighboring nodes) is a crucial requirement. For example, Blumenthal et al. [5] have shown that even small errors in distance determination significantly affect the accuracy of the estimated sensor positions in both exact and approximate localization procedures. For example, the relative localization error of the coarse-grained localization algorithm [5, 3] increases by 5% if the visibility of the beacons decreases by 16%.

As is well known, the strength of the transmission signal $d \approx 1/r^2$ attenuates approximately with the square of the distance between sender and receiver. Obstacles that are on the direct line between sender and receiver are a further cause for a signal attenuation. Of course, this attenuation effect, also known as shadowing, depends on the object's size, the material it is made of, as well as the radio technology and the utilized frequency. For example, an isolated tree attenuates a signal at 876 MHz by about 8-20dB, bricks and tinted glass walls attenuate a 2.4 GHz signal by about 3dB and 19dB, respectively, whereas a bunch of trees and solid rocks might block the radio signals entirely (see [11] for further examples and details). Furthermore, obstacles give rise to further disadvantageous physical effects, such as diffraction, scattering, and reflection, which are briefly sketched in Figure 2.

3. SOME OBSERVATIONS

Assume a network topology as depicted in Figure 1, and assume a transmission range $r \geq \sqrt{2}w$ of the beacons. In the absence of any obstacle, all nodes within that square receive a message from all four beacons $B_{1..4}$. However, if an obstacle is present, some nodes do not receive messages from all the beacons due to shadowing. Consequently, the area of interest can be separated according to the messages $m_{1..4}$ the nodes receive. The four-bit number $c = \{c_4, c_3, c_2, c_1\}$, with c_i indicating whether or not the node has received message m_i from beacon B_i, can serve as a unique region code as is exemplified in Figure 3. It is clear that depending on the obstacle's location and the four bits c_i, an area might have up to $2^4 = 16$ different region codes.

In addition, the situation, as depicted in Figure 3, allows for the following three observations:

1. The border between two neighboring region codes appears as a straight line.
2. Depending on both the obstacle's location and shape, two or three border lines intersect. This intersection is called *virtual corner* throughout this paper. Due to shadowing, these intersections are quite close to the obstacle (see, also, Figure 3).
3. Intersections are predominantly close to the obstacle's corners and/or edges. These specific points can be found in areas with changing region codes c.

In addition to the observations with respect to region codes, the following one can be made: nodes close to an obstacle, also called edge-nodes for short, experience a lower density μ_e of neighbors in their vicinity. At the obstacle's edges, this density $\mu_e = 0.5\mu$ arrives at 50% of the original value, with increasing and decreasing values at concave and convex corners, respectively.

4. THE DOL ALGORITHM

Based on the observations discussed above, this section presents the distributed obstacle localization algorithm. It operates in three subsequent stages, called *region code determination*, *field segmentation*, and *obstacle detection*. The remainder of this section is devoted to the description of each of these three stages, as well as presenting the algorithm in pseudo code and discussing its energy consumption.

4.1 Region Code Determination

At the very beginning of the required self-organization process, each beacon broadcasts a message containing its position and a beacon-specific identifier. By employing one of the localization algorithms already mentioned above, the nodes are able to determine their approximate position. All nodes can also use these very same messages to construct their region codes c. Depending on that region code, all nodes can assign themselves to one of the possible 16 different regions. This stage does not involve any additional communication overhead for the resource-limited sensor node.

4.2 Field Segmentation

After all nodes have determined both their positions and region codes, they perform a local inquiry. To this end, all nodes broadcast their particular region code with a relatively short transmission range σ. This (local) transmission range

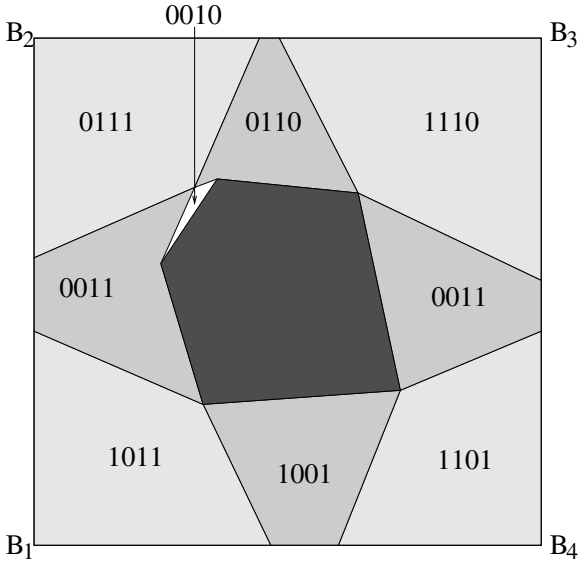

Figure 4: This figure shows the codes and thus regions the nodes have derived from the received packets. Please note the little tiny area labeled '0010' at the obstacle's uper left edge.

$\sigma \ll r$ is significantly shorter than the beacons' transmission range $r = \sqrt{2}w$; it should be a few times the average distance δ between neighboring nodes, e.g., $3\delta \leq \sigma \leq 5\delta$. Since all nodes broadcast a message, each one would normally receive $N_\sigma^{\text{exp}} \approx 2\pi\mu\sigma^2$ messages from its neighborhood, with μ denoting the node density. Then, those nodes are at (or at least close to) an intersection that receive at least three different region codes (see also Section 3). In so doing, the nodes also keep track of that number N_σ^{act} of actually received messages, since that number N_σ will also be used in the third stage.

Figure 4 presents an example in order to illustrate the result of this stage. The nodes have segmented the entire area into nine different regions, with one region labeled '0010' being very small. This second stage has a very little communication overhead, since it requests all nodes to send only one additional message. However, this tiny communication overhead helps reduce the overall costs in the subsequent stage as is discussed below.

4.3 Obstacle Detection

The main result of the previous stage was the determination of virtual corners of a virtual object slightly larger than the actual obstacle (compare also with Figure 3). This has been achieved by segmenting the entire area of interest. Consequently, this stage aims at a refining this result. To this end, the nodes at virtual corners trigger (or activate) their neighboring sensor nodes to broadcast the fraction $\Delta = N_\sigma^{\text{act}}/N_\sigma^{\text{exp}} = N_\sigma^{\text{act}}/(2\pi\mu\sigma^2)$ of received N_σ^{act} and N_σ^{exp} expected messages determined in the previous algorithm stage. At the obstacle's edges, this fraction approaches $\Delta \approx 0.5$ as has already been discussed in Section 3. In order to guarantee that this activation reaches also those nodes at the obstacles edges[2] the triggering may be

[2]It might happen, for example, that the distance $D > 2\sigma$ between virtual corner and obstacle is greater than the local

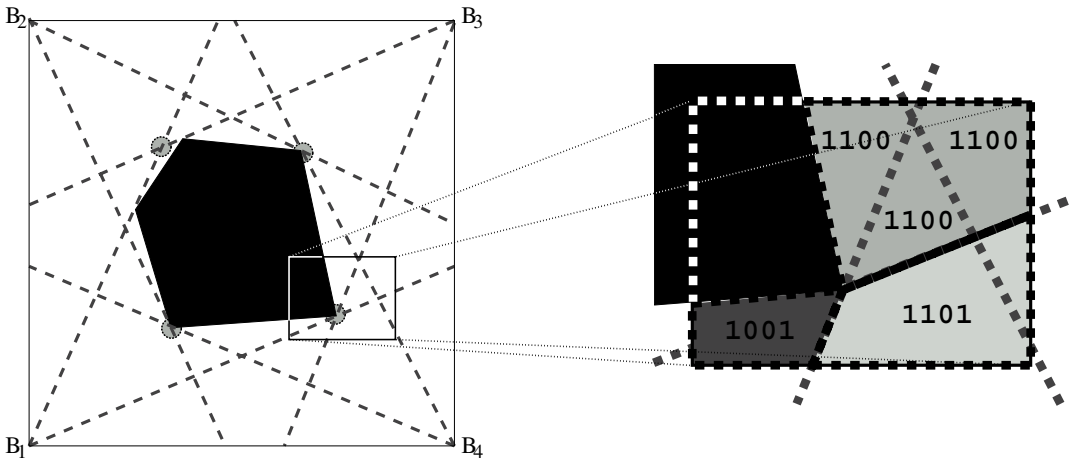

Figure 3: This figure shows how an object causes shadowing of beacons; The dashed lines indicate those regions. "Virtual corner", i.e., small areas with at least three different area codes, are highlighted. The enlargement depicts various region codes at the obstacle's edge.

forwarded by all nodes that have a fraction $\Delta < 1$ significantly smaller than 1.

At this point in time, all edge-nodes have the following knowledge: they have a fraction $\Delta \approx 0.5$, they are in the neighborhood of a virtual edge, and their neighbors also have determined a fraction $\Delta \approx 0.5$ significantly smaller than 1. If one of those conditions does not hold for a particular node, it is probably not an edge-node, and the small fraction Δ might be due to other causes, such as a non-uniform distribution of the sensor nodes or other failures.

The final step consists of the transmission of the edge-nodes' coordinates to the network infrastructure. In order to keep the communication overhead low, not all nodes but a fraction $1/\sigma$ is allowed to do so. All edge-nodes easily accomplish this goal by drawing a random number and sending a message to a nearby beacon in particular cases. From all received messages, the network infrastructure in turn reconstructs the obstacle's position. Since the beacons possess extended computing and communication facilities, they finally broadcast appropriate routing information to all sensor nodes.

4.4 Pseudo Code

For reference purposes, Figure 5 presents a concise description of the DOL algorithm. In this description, step 0 refers to a general network setup in which the beacons broadcast all relevant information, such as transmission ranges r, node density μ, area size w, etc. In this setup, the nodes also perform their localization. Steps 1 and 2 refer to the first, Steps 3 to 5 the second, and Steps 6 to 9 to the third algorithm stage.

4.5 Energy Consumption and Robustness

At first glance, it might seem that the algorithm's design is a bit awkward. However, those design steps aim at saving valuable energy resources and making the algorithm as robust as possible.

Starting in Step 3, the transmission of one additional message applies to all sensor nodes. Even though significant in total, this overhead is quite decent for every single node,

transmission range σ.

The DOL Algorithm

```
Step 0: Initialization:
        all beacons broadcast all
        relevant (network) parameters,
        e.g. transmission range r,
        the beacons' positions etc.
Step 1: all nodes wait t_1 and receive
        messages m_i from beacons B_i
Step 2: all nodes determine their
        region codes c_i
Step 3: all nodes broadcast their
        region codes c_i with small
        transmission range σ
Step 4: all nodes wait t_2 and receive
        their neighbors' region codes c_j
Step 5: all nodes determine if they are
        at a virtual corner
Step 6: all corner-nodes activate
        their neighbors to broadcast
        their message count with a
        small transmission range σ
Step 7: all edge-nodes draw a random
        number and transmit their
        coordinates to the beacons
Step 8: beacons and network infra-
        structure re-construct the
        obstacle's position
Step 9: beacons broadcast appropriate
        routing information
```

Figure 5: A concise description of the DOL algorithm in pseudo code. For further details, see text please.

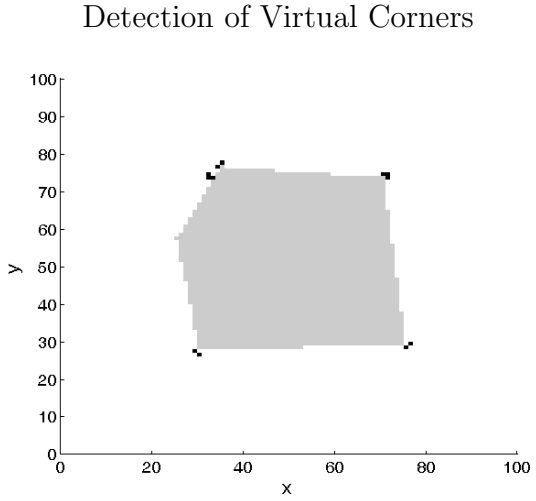

Figure 6: This figure shows how the field-segmentation stage detects virtual corners (black dots), which are close to the actual obstacle.

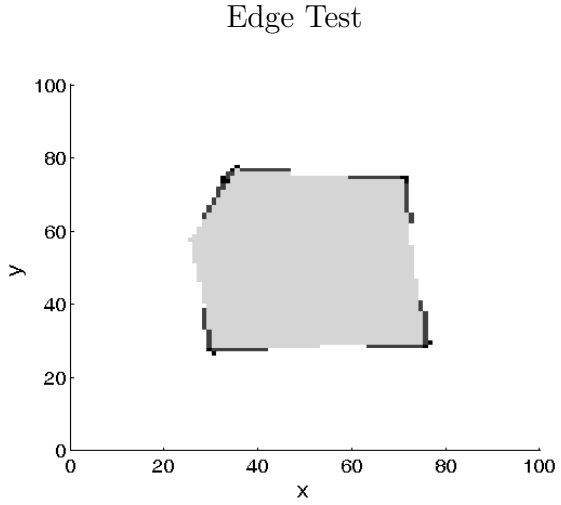

Figure 7: This figure shows the detected corners as well as edges as they appear after the obstacle-detection stage.

without having any significant impact on the network's lifetime. With the determination of virtual corners in Step 5, the algorithm limits further processing steps to those sensor nodes that are indeed close to the obstacle. Thus, only those sensor nodes might consume a small amount of energy that have to do those calculations. Furthermore, this energy consumption will not reduce the network's lifetime, since these sensor nodes are not suitable candidates for later routing activities.

The determination of the virtual corners serves a second purpose: it might seem sufficient to merely calculate the fraction Δ by comparing the expected and actually received number of messages. However, in a real-world scenario, some sensor nodes might assume to be an edge-node by mistake for any reason. Since edges require the presence of virtual corner due to geometrical reasons, the activation by means of the latter helps prevent spurious edges and obstacles. The broadcast of the fractions Δ in Step 6 is a further mechanism to improve the algorithm's robustness, since edge-nodes are surrounded by other edge-nodes in real-world scenarios.

5. SIMULATION RESULTS

The DOL algorithm described above was implemented and tested in Matlab[3]. Even though the simulation software has been used for various configurations, this section presents a case study that adopts the network and obstacle configuration as discussed in Sections 2 and 3. The simulation has used the following parameter settings: distance between beacons w=100, sensor node density μ=1 (resulting in 10,000 node in a 100×100 area), distance between sensor nodes δ=100/100=1, beacon transmission range $r = \sqrt{2}w = 142$, and a small transmission range of σ=3δ=3. The x/y coordinates of the obstacle were $(x_i, y_i) \in \{(30, 27), (75, 29), (70, 73), (35, 76), (25, 57)\}$. In order to keep the simulation time reasonable, all coordinates were integer values, the sensor nodes were systematically distributed (rather than randomly), and only radio signal blocking was implemented at obstacles.

The region codes determined in the first phase, look virtually identical to the one already shown in Figure 4, and is thus omitted. After broadcasting their own region codes into their vicinity, those nodes consider themselves an edge node that receive at least three different region codes. The result of this process is depicted in Figure 6. After detecting the virtual corners, those corner nodes activate edge-detection tests. Then, those nodes become edge-nodes, which have a fraction $\Delta < 0.7$ close to 50%. The result of this test is shown in Figure 7. Finally, the network infrastructure re-constructs the obstacle by a polygon.

This particular simulation has revealed that only four out of five corners have been detected. One corner was not found in this case, because obtuse angles of an obstacle are difficult to reveal, without additional methods. Moreover, the existence of the unformed region c=0010 avoids a successfully detection. This, however, results only in a very small error during the obstacle's re-construction. In additional simulations with other obstacles of various sizes and shapes, the algorithm precisely detected all the obstacles' corners.

6. IMPLICATIONS

The background section has outlined that during the network's self-organization process, all sensor nodes try to determine at least a position estimate for which the pertinent literature offers a large number of algorithms [5, 3, 4]. Most of these localization algorithms estimate the sensor positions according to some distance measurements, either to other sensor nodes, beacons, and/or other landmarks. In addition to the time-of-flight or the signal attenuation, *hop counts* are appropriate and widely used mechanisms to derive the required distance estimates.

A hop count counts the number of hops (relays or nodes) a message visits when traveling from source to destination. For obvious reasons, a large number of nodes, and thus hops, increases the estimate accuracy, and the shortest path along the line of sight yields the best accuracy.

[3]The simulation code can be retrieved via email from Frank Reichenbach `frank.reichenbach@uni-rostock.de`.

Unfortunately, this approach does not work well in the presence of obstacles, because the messages cannot go through the obstacle but must travel around them. This detour, however, spoils the hop count and thus both the distance and position estimates. The DOL algorithm can at least relieve this problem by enhancing the message header with an obstacle parameter. At least in theory, this parameter can compensate for the hop count errors when approaching the obstacle's edges (i.e., when visiting edge-nodes) at least to a certain degree. But the (simulation and) in-depth analysis of this approach will be devoted to future research.

The obstacles coordinates can also be used to derive additional routing information. As has been discussed above, Yamazaki's and Sezaki's routing algorithm [10] requires the location of (imaginary) reference points. Originally, Yamazaki and Sezaki have set those reference points by hand, prohibiting the self-organization of the network. Now, the network infrastructure can easily define reference points, after the DOL algorithm has determined the obstacles' positions. The DOL algorithm can thus close the gap between hand-crafted routing and self-organizing large wireless sensor networks.

7. CONCLUSIONS

This paper has briefly discussed why obstacles impose significant problems on both the localization and routing algorithms. This paper has then discussed some of the effects that can be observed at and around obstacles. It has turned out that shadowing can be beneficially exploited, since the blockage of radio signals can be used for field segmentation. The intersection of different fields (with different region codes), then, form a virtual object only slightly larger than the physical obstacle.

As a result, this paper proposes the distributed obstacle localization algorithm. By slightly increasing the communication overhead during the network self-organization process, the DOL algorithm is able to localize obstacles with sufficient precision. This paper has finally discussed how the knowledge of the obstacles' positions can be used to increase the localization accuracy as well as to allow efficient routing.

The authors consider the present algorithm a first step towards large real-world wireless sensor networks in which the presence of obstacles is the rather normal case. Future research will be devoted to several aspects relevant in realistic real-world simulations. This includes among other things diffraction of radio waves, packet losses and retransmissions, as well as more than one obstacles with also concave and rather general shapes. In addition, the algorithm presented in this paper should be thoroughly mathematically analyzed.

Furthermore, future research will be devoted to practical tests for which the institute's equipment can be utilized. The purpose of these tests are not only to validate the simulation results but also to improve the localization of both sensors and obstacles.

Acknowledgements

This work was supported by the German Research Foundation under grant number TI254/15-1 and BI467/17-1 (keyword: Geosens). The authors gratefully thank Tino Boelke for implementation and simulation support.

8. REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A Survey on Sensor Networks, *IEEE Communications Magazine,* August 2002, pp. 102-114.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y. -C. Hu, and J. Jetcheva, A Performance Comparison of Multi-Hop Wireless ad-hoc Network Routing Protocols, in *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking,* 1998, pp. 85-97.

[3] N. Bulusu, J. Heidemann, and D. Estrin, GPS-less Low Cost Outdoor Localization for Very Small Devices, in *Proceedings of the IEEE Personal Communications Magazine,* vol. 7(5), October 2000, pp. 28-34.

[4] A. Savvides, C. C. Han, and M. B. Strivastava, Dynamic Fine Grained Localization in ad-hoc Networks of Sensors, in *Proceedings of the 5th International Conference on Mobile Computing and Networking, MOBICOM 2001,* Rome, Italy, 2001, pp. 166-179.

[5] J. Blumenthal, F. Reichenbach, and D. Timmermann, Position Estimation in ad-hoc Wireless Sensor Networks with Low Complexity, *Joint 2nd Workshop on Positioning, Navigation, and Communication, (WPNC 05) and 1st Ultra-Wideband Expert Talk 2005,* 2005, pp. 41-49.

[6] L. Doherty, L. Ghaoui, and K. Pister, Convex position estimation in wireless sensor networks, in *Proceedings of the IEEE Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'01),* 2001, pp. 1655-1663.

[7] C. Savarese and J. Rabaey, Robust positioning algorithms for distributed ad-hoc wireless sensor networks, *Proceedings of the USENIX Annual Technical Conference,* 2002, pp. 317-327.

[8] C. E. Perkins and E. M. Royer, Ad-hoc On-Demand Distance-Vector Routing, in *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications,* New Orleans, LA, 1999, pp. 90-100.

[9] D. B. Johnson and D. A. Maltz, Dynamic Source Routing in ad-hoc Wireless Networks, *Mobile Computing,* Kluwer Academic Publishers, 1996, pp. 153-181.

[10] K. Yamazaki and K. Sezaki, An ad-hoc Routing Protocol with Obstacle Evasion, in *Proceedings of the First International Workshop on Networked Sensing Systems Program (INSS04),* 2004.

[11] J. Wheat, R. Hiser, J. Tucker, A. Neely, and A. Mccullough, *Designing a Wireless Network,* Syngress Publishing, 2001.