Integrating Sensor Networks in Pervasive Computing Environments Using Symbolic Coordinates

Matthias Gauger, Pedro José Marrón,
Marcus Handte, Olga Saukh, Daniel Minder
Universität Bonn, Bonn, Germany and
Fraunhofer IAIS, St. Augustin, Germany
{gauger, marron, handte, saukh, minder}@cs.uni-bonn.de

Andreas Lachenmann, Kurt Rothermel
IPVS, Universität Stuttgart, Stuttgart, Germany
{lachenmann, rothermel}@informatik.uni-stuttgart.de

***Abstract*—Wireless sensor networks can monitor different types of physical phenomena and are able to provide a diverse set of context data to interested clients. Allowing mobile pervasive computing devices to access such data requires solutions for routing messages between mobile devices and the static sensor network. This paper presents a novel approach that addresses this problem with the help of symbolic coordinates. It requires only a small amount of topology information distributed in the network and allows mobile devices to send messages to arbitrary areas. The routing task is split among the client nodes, which specify a symbolic source route, and the sensor nodes that handle node-to-node routing. The paper describes the algorithm, specific challenges associated with its design and gives an extensive evaluation of the approach and its properties, showing that the use of symbolic coordinates in these environments is a viable alternative to more traditional types of routing.**

I. Introduction

The rapid evolution of wireless sensor network technology in the last few years has allowed them to monitor a variety of physical parameters and provide this data to interested clients. For this purpose, the sensor network research community has developed different solutions for efficient data collection, data transport and data aggregation that aim to provide the required services, while at the same time conserving the scarce resources of sensor nodes.

Many pervasive computing applications rely on the use of contextual data to provide their services to the user. Pervasive computing devices can either collect this data directly from their environment or retrieve it from other devices over the wireless network. Sensor networks play an important role in pervasive computing application scenarios as providers of both raw sensor data as well as of preprocessed context data.

The combination of these two different types of systems – large scale, extremely resource-constrained sensor networks on the one hand and mobile pervasive computing devices on the other hand – creates new challenges that need to be addressed from the ground up. The most fundamental challenge deals with the efficient communication and routing of data between mobile devices and sensor nodes.

Installing a sophisticated infrastructure to collect sensor data and provide this data to pervasive computing devices can be costly both in terms of hardware and communication overhead. For this reason, we propose that mobile devices communicate directly with the sensor network in their surroundings. However, while most of the required communication will be with nearby nodes, mobile devices still need to be able to access data from nodes in any area of the network.

In the area of ad-hoc networks a variety of solutions has been developed for routing between mobile devices without relying on infrastructure support. However, their applicability to our scenario is very limited due to the special properties of wireless sensor networks [1], [2]. The main problems are the scarce availability of resources, the instability of nodes and communication links between them and the expected larger scale of sensor networks.

Several types of routing protocols have been proposed specifically for sensor networks that are optimized for efficiently collecting data or events from the network while limiting the amount of traffic and state on the nodes required for their operation [2]. Many of these algorithms use tree structures for this purpose. However, they can neither easily support multiple independent client nodes nor mobile client nodes that frequently change their position between sending queries to the network.

Location based routing protocols address most of the efficiency and scalability problems discussed above both for ad-hoc and for sensor networks. Moreover, they also provide a natural addressing scheme for sensor network scenarios where the user normally wants to retrieve data from a certain area instead of communicating with a specific node. However, location based routing requires detailed knowledge of the geographic positions of the individual nodes to allow for informed routing decisions. In many scenarios, such information is unavailable or too expensive to acquire for all sensor nodes.

The goal of the work presented in this paper is to overcome the limitations of existing routing protocols and to find an efficient routing solution for mobile pervasive computing devices in combination with wireless sensor networks. By adding a small amount of topology information in the form of symbolic coordinates we obtain the advantages of location based routing

protocols without requiring accurate position information.

We describe a simple, yet efficient routing algorithm that allows mobile nodes to access data on arbitrary sensor nodes without relying on support from a base station. The basic idea is to split the routing task between the mobile node (e.g., a PDA) that issues the request message and the static sensor nodes that forward and answer the request. The mobile client nodes calculate a symbolic source route from their current position to the destination sent as part of the request message and the sensor nodes use this information to perform node-to-node routing. The advantage of such a splitting is that the client nodes do not need to manage a detailed view of the current sensor network topology and the sensor nodes can correctly forward messages using purely local information.

We aim at application scenarios consisting of a large set of static sensor nodes distributed over the different symbolic areas of a network and several mobile client devices that move around these areas and use the data of the sensor nodes to provide pervasive services to their users. The first specific application we are working on is the integration and use of sensor networks in office scenarios without the need for complex infrastructure support. For future investigations, we are also considering larger scale sensor networks in outdoor scenarios like for agricultural applications or habitat monitoring, where the definition of the work environment and an assignment of symbolic coordinates comes naturally.

The rest of this paper is organized as follows. Section II gives information about existing approaches and their respective shortcomings for the scenarios we consider. In Section III, we introduce the concept of using symbolic coordinates in sensor networks and describe our routing algorithm. Section IV describes our prototype implementation and Section V provides an evaluation of our approach. Finally, Section VI concludes the paper and discusses future work.

II. Related Work

There has been active work on **symbolic coordinates** and location models in different areas of pervasive computing research (e.g., [3], [4], [5]). Becker and Dürr [6] give a comprehensive overview of different geometric and symbolic location models from the perspective of pervasive computing and compare their suitability for different types of queries. The focus of our work is not the location model but the use of basic symbolic location information for routing. Since our approach only requires the availability of a `neighborOf`-relationship among symbolic coordinates, it should be compatible with most existing location models.

The general idea of using symbolic coordinates in sensor networks has been formulated by Fekete et al. [7]. However, the focus of their work lies more on detecting boundaries and extracting information about the topology of the network than on actually doing message routing between nodes. Moreover, they assume very densely populated topologies that do not pose many of the challenges discussed in this paper.

There exists a large body of work on node-to-node routing in the area of ad-hoc networks which can be coarsely classified into flooding based approaches, proactive and reactive algorithms (topological approaches) and location based approaches. Unfortunately, these different approaches are not applicable to sensor networks without problems due to the differences in the system model of ad-hoc and sensor networks.

The basic way of distributing data in a network – **flooding of messages** to all nodes – generates a huge overhead in large networks as basically every node needs to participate in every communication. Different optimizations like **gossiping** [8] or **probabilistic flooding** [9] can only partially alleviate this problem since still a large fraction of the nodes needs to participate in the message distribution. **Proactive routing protocols** (e.g., [10]) only work in networks of very limited size as the continuous exchange and maintenance of global routing information creates a considerable overhead both in terms of messages and concerning the amount of state maintained by the individual nodes. **Reactive routing protocols** (e.g., [11], [12]) only calculate routes on demand which effectively limits the overhead for continuous route maintenance. However, discovering routes usually implies the flooding of route request messages. Despite optimizations like route caching, this can quickly cause an unacceptable message overhead in large sensor networks. **Source routing** (e.g., [13]) is an important subclass of reactive routing protocols. The idea is to include the discovered route from the source to the destination in the header of each message. **Hybrid routing protocols** (e.g., [14]) combine the concepts of proactive and reactive routing protocols by actively maintaining routes for a limited area (e.g., in the local neighborhood) and using reactive routing on a global scale. Other approaches combine proactive routing with location based routing [15].

Unlike routing based on symbolic coordinates, **location based** or **geographic routing** based on real-world coordinates is already well-established in sensor networks and cited in numerous sensor network publications. Its best-known representative is Greedy Perimeter Stateless Routing (GPSR) [16]. However, recent work (e.g., [17]) has shown that considerable additional effort is required to make geographic routing work in realistic environments. In sensor networks, the applicability of geographic routing depends on the availability of the required position information which is often not provided due to limitations of the hardware (e.g., no GPS receiver) or the environment (e.g., indoor scenarios).

Recently there has been some effort on making routing in large-scale wireless ad-hoc networks more efficient and scalable by avoiding the flooding of messages for message distribution or route discovery. **VRR** [18] maintains a virtual ring of nodes that allows forwarding messages to arbitrary nodes. A similar approach is followed by **Scalable Source Routing** [19] where all nodes maintain source routes to their neighbors in a virtual ring. For sensor networks, the main drawback of both approaches is the required amount of state at each node that grows with the size of the network. Together with the instability of links this can make the management of routes to the virtual neighbor nodes a complex and expensive task.

III. Symbolic Routing in Sensor Networks

This section introduces our routing algorithm based on symbolic coordinates. We start with some preliminaries and the system model before describing the basic algorithm and several extenstions required to address challenges occuring with the basic approach.

A. Preliminaries

Symbolic coordinates define a position or an area in the form of an abstract symbol, as opposed to geographic coordinates that define a position in the form of a coordinate tuple (e.g., longitude, latitude and height) relative to a global or local coordinate system. Typical examples of symbolic coordinates are room numbers, cell IDs of wireless networks and street addresses. Since symbolic coordinates do not have any direct connection to their geographic location, a location model is required to associate a geographic area with its corresponding symbolic coordinate.

A symbolic coordinate can represent areas of different shapes and sizes and, in many cases, directly represent the semantics of a location, for example when a symbolic coordinate is associated with each room of a building. This can simplify the formulation of queries since the destination area of a query can be directly expressed by an intuitive coordinate similar to the one used in everyday life.

B. System Model

For our algorithm, we assume that there are two types of devices with a large number of static, resource-constrained sensor nodes and a much smaller number of mobile, more powerful pervasive computing devices. In particular, we assume that the number of mobile devices is too small to reliably form a backbone network of mobile devices to route requests through them. Both types of devices share a common communication interface that allows the mobile devices to communicate with any sensor node in their direct neighborhood. For the communication between the nodes we assume bidirectional connections.

The mobile nodes must have access to a symbolic location model which might be preloaded to the nodes or dynamically retrieved from an external network. A continuous connection to an infrastructure is not required for the mobile nodes.

The type of information the nodes require in our approach is not the typical symbolic location model data, as addressed in the literature, but rather a symbolic topology graph expressing the neighborhood relationship between individual symbolic areas. Such a topology graph can be extracted from different types of location models using simple tools.

We assume that both sensor nodes and mobile devices know the symbolic coordinate of the area they are located in. This is a strong requirement, but it is much less demanding than requiring knowledge of the node's exact geographic location. In previous work we have developed mechanisms for a low overhead assignment of symbolic coordinates using explicit sensor triggers [20]. We are also working on mechanisms for further automating this process by grouping nodes based on sensor data similarity.

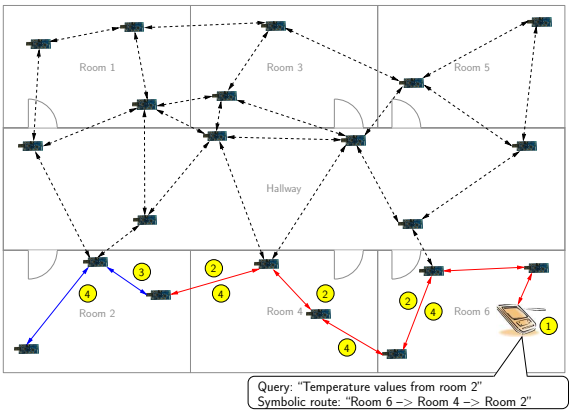


Fig. 1. Example query in an office scenario

C. Basic Approach

The basic idea of our approach is to perform symbolic routing in sensor networks by letting the mobile client nodes specify a symbolic source route and having the static sensor nodes use this source route to make node-to-node routing decisions. Fig. 1 shows an example of a mobile client device querying the sensor network in an office scenario. In addition to the query itself ("Temperature values from room 2") the client also provides the symbolic route ("Room 6 - Room 4 - Room 2") as part of the query message. The sensor nodes have learned about their local neighborhood using a beaconing mechanism. They now use this information to forward the message along the specified route until the destination area "Room 2" is reached. In the following we describe the individual parts of the algorithm in detail.

Beaconing mechanism: The sensor nodes send out beacon messages at regular intervals advertising their own symbolic coordinates and neighboring symbolic coordinates they have already heard of. For these neighboring symbolic coordinates the beacon messages also contain a distance field that advertises the number of hops on the node level to the respective neighboring symbolic coordinate. Each node only manages distance information about directly neighboring symbolic coordinates, i.e., symbolic coordinates the node's own symbolic coordinate can directly communicate with. This limits the amount of state each sensor node has to manage.

Mobile client nodes only participate passively in the beaconing process – learning from received beacon messages about their current symbolic coordinate and about sensor nodes they can forward their queries to.

Like all protocols that store distance vectors in nodes, our beaconing mechanism is vulnerable to the count-to-infinity problem when a connection breaks down. However, since we only work with local distances to neighboring symbolic coordinates, it is possible to prevent this by defining a reasonable maximum distance value.

Query preparation: When a mobile client node wants to send a query message to a specific area of the sensor network, it first needs to determine a symbolic route from its current location to the symbolic coordinate of the destination area (Step 1 in Fig. 1). Since the client node is typically more powerful than the sensor nodes, it has the capabilities to compute this route using the symbolic topology graph information contained in the location model. Note that such symbolic routes can be computed without performing a route discovery procedure at the node level.

Node-to-node routing: Once the client has passed its request message to one of the sensor nodes, it is the task of the sensor network to perform the node-to-node routing (Step 2 in Fig. 1). For doing this, the sensor nodes use the symbolic route contained in the request message as well as the local routing table calculated based on the beacon messages received from neighboring nodes. When a node receives a message to forward, it looks up the next-hop symbolic coordinate in its local routing table, retrieves the next node in direction of this coordinate and forwards the message to this node.

One simple optimization that achieves shorter communication paths is to check the source route beyond the next-hop symbolic coordinate to see whether a symbolic coordinate further down the path can be directly reached. We call this optimization **path look-ahead**.

Message delivery: Once the message reaches the first node inside of the destination area, the system provides different semantics to deliver the message to the sensor nodes (Step 3 in Fig. 1): delivering to an arbitrary node in the area (**area anycast**), delivering to all nodes in the area (**area broadcast**), or delivering to a specific node (**area unicast**) specified by the sender. In the future, we also plan to support semantic criteria for delivering messages. For example, it could be possible to send a query to all temperature sensors in an area.

We have implemented area broadcasts and area unicasts by broadcasting the message inside the destination area and delivering the message to all nodes or the unicast destination node. The costs for such a broadcast are limited to the nodes in the destination area since nodes of neighboring areas do not forward the broadcast.

Reply routing: Sending a reply message from a sensor node back to the original sender of the request is done by reversing the symbolic route received with the query message (Step 4 in Fig. 1) and sending the reply as an area unicast to the original sender. Note that this does not require any model knowledge stored on the sensor nodes.

The reply message can only be delivered to the mobile node if it stays within the transmission range of the symbolic area it sent the request from for the short time period between request and reply. Note that this should be the common case as we expect the speed of nodes to be rather small compared to the speed of message transmission. If, however, the mobile node moves to another symbolic area it is able to detect this situation based on the beacon messages it receives from its new neighbor nodes and can resend the request.

We imagine that a more sophisticated solution could be useful in scenarios where higher node mobility is a critical factor: The mobile node could explicitly store forwarding information at one of the sensor nodes before leaving the symbolic area it sent a request from.

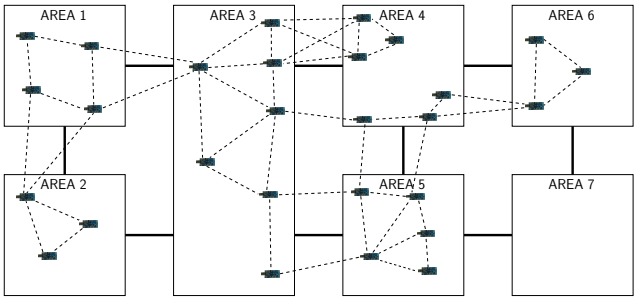


Fig. 2. Example of a symbolic area graph and a node connection graph

D. Challenges

As long as the symbolic routing path specified in a message is covered by the sensor network, the basic routing approach described above will always work. In this case, message loss is only caused by transmission errors, for example due to collisions, and can be dealt with at the MAC layer (hop-by-hop) or at the application level (end-to-end). However, in realistic scenarios, a certain part of the routing attempts will fail even though the specified symbolic route is perfectly legal. This is due to a discrepancy between the topology graph formed by symbolic coordinates and their neighborhood relationships and the communication topology graph of the sensor nodes. Our basic routing approach assumes that a neighborhood relationship between two symbolic coordinates implies that nodes in these two areas are able to communicate with each other. While this is a helpful heuristic in most cases, it does not have to hold in general, as shown in the example in Fig. 2. We have identified three different types of problems caused by the limitations of our heuristic: **Communication holes**, **coverage holes** and **area partitionings**.

The sensor network contains a **communication hole** when sensor nodes deployed in two neighboring symbolic areas cannot communicate with each other without going through nodes in other symbolic areas. In Fig. 2 this problem shows between area 2 and area 3. A message coming from a node in area 2 must go through area 1 to reach area 3 even though the symbolic path might specify area 3 directly following area 2.

We speak of a **coverage hole** when the area of a symbolic coordinate does not contain any sensor nodes. In Fig. 2 area 7 shows an example of this. Coverage holes cause the same problems as communication holes, i.e., routing over symbolic paths including the uncovered coordinate fails. Additionally, messages addressed to the uncovered coordinate cannot be delivered at all. However, this second problem cannot be solved by the routing algorithm – coverage of all relevant areas in a sensor network must be ensured at deployment time.

More subtle than the communication and coverage hole problems is the problem of having **area partitionings**. We call

a symbolic area partitioned if two nodes of the same symbolic coordinate are not connected through a communication path only consisting of nodes lying in the same symbolic area. Fig. 2 illustrates this problem: While the sensor network as a whole is connected, the nodes in the upper half of area 4 can only communicate with the nodes in the lower half of the same area by going through area 3.

Area partitions cause two types of problems. Firstly, the success of routing through a partitioned area can depend on which partition is used. In the example in Fig. 2 a message coming from area 3 can only be routed to area 6 if the partition in the lower half of area 4 is used. Secondly, partitioned areas also interfere with the delivery of messages: Area broadcasts can only reach nodes in the partition the broadcast is started in. An area unicast only completes successfully if the destination node lies in the partition reached by the symbolic routing.

E. Extensions to the Basic Approach

In the following paragraphs we describe several extensions to the basic symbolic routing approach that address the challenges described above. They can be classified along whether they aim to prevent problems (*preventing holes*), whether they are used to react to problems (*recovering from holes*, *recovering from partitionings*) or a mixture of both (*collecting connectivity information*).

1) Preventing holes: With the help of some model knowledge or control over the network topology, it is possible to avoid many of the communication and coverage holes on symbolic paths used for routing. This helps reducing the number of routing failures experienced by the clients.

The canonical way of preventing communication holes, coverage holes and area partitionings is to provide for a sufficiently **dense deployment** of sensor nodes over the network area. However, a network topology with these properties usually cannot be guaranteed over the complete network area due to both cost and deployment reasons.

Another way of avoiding the appearance of communication holes on symbolic paths is to **store knowledge about holes in the location model** used by the client nodes for calculating symbolic routes. If such information is available, clients can remove the affected neighborhood relations from their symbolic location graph. One slightly less demanding alternative is to store a **weight factor based on the (expected) node density** for each symbolic area. The reasoning behind this is that two neighboring areas with high node density are connected with a higher probability than two neighboring areas with low node density. By avoiding areas with low node density, a node can select a safe path from source to destination.

2) Recovering from holes: We have developed two approaches to recover from holes that cannot be avoided when using the mechanisms described above: **local symbolic broadcasts** and **feedback messages**.

When using **local symbolic broadcasts**, a node that is not able to forward to the specified next-hop symbolic coordinate (i.e., it detects a hole) forwards the message to all neighboring symbolic coordinates and asks them to check whether they can find a way back onto the original symbolic route specified in the message. Note that broadcasting the message to neighboring symbolic coordinates of the node does not require any flooding since each neighboring coordinate can be reached using unicast messages following the entries in the node's routing tables. The maximum depth of the symbolic broadcast is configurable. However, the forwarding costs grow considerably when going beyond a depth of one or two.

The alternative to broadcasting the message locally is to send a **feedback message to the original message sender** in order to inform the client node about the routing failure and to provide information about where the routing failed. The client node is then able to calculate a new symbolic route not using this connection and is also able to buffer this information for later use. After a sufficient number of such message send and feedback cycles it should always be possible to send messages to the desired destination area, provided the sensor network is not partitioned. Although not part of our current implementation, we also imagine that mobile nodes could share this feedback information by exchanging messages when meeting each other or pushing it back to infrastructure.

Which of the two solutions – broadcasting the message or sending feedback to the original sender – is preferable might depend on the amount of symbolic hops the message has already travelled. A message that has almost reached the destination coordinate should probably be broadcasted whereas it might be more efficient to send back a feedback message when the message has only covered a few symbolic steps.

3) Recovering from partitionings: While routing over partitioned areas is possible in most cases using hole prevention and hole recovery, it is more difficult to find a solution for the second type of problem caused by area partitionings – the delivery of broadcast and unicast messages. While the two partitions in our example in area 4 of Fig. 2 are connected through the neighboring area 3, it is also possible that the shortest connecting path between two partitions traverses multiple symbolic areas. Finding a solution for the general case would thus require searching and storing complex routing structures that are not necessarily limited to the local neighborhood.

In the following solution we only consider the (common) case where just one symbolic area must be traversed to connect two partitions of a symbolic area. The idea is that a node receiving a beacon message from a neighboring symbolic coordinate not only stores the information that it can directly reach this coordinate but also calculates a hash value over the set of neighboring coordinates this node advertises. Every once in a while it advertises this hash value to the other nodes in its own symbolic area. For example, in Fig. 3 the upper most node in area 4 would send the hash value $hash(1, 4, 5)$ which corresponds to the symbolic neighbors of area 3. If the neighboring coordinate is partitioned and both partitions are connected to nodes of this symbolic coordinate two different hash values will be advertised. Note that the hash values of the two partitions should differ since it is extremely unlikely that two partitioned sets of nodes of a symbolic area are connected

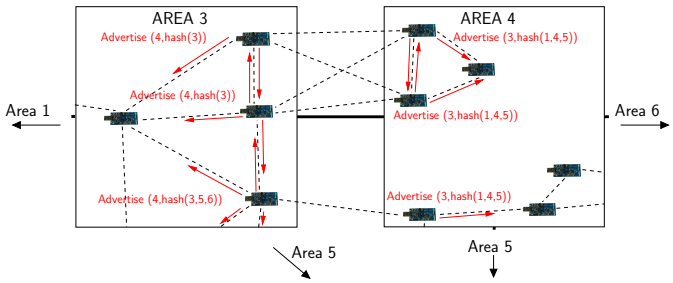


Fig. 3. Example of advertisement messages for a partitioned symbolic area (Extract from Fig. 2)

to exactly the same set of neighboring coordinates. In the figure, two different hash values are distributed in area 3 for the partitioned area 4 whereas in area 4 only one hash value exists for representing the unpartitioned area 3.

When the nodes in a symbolic area continuously receive more than one hash value advertised for a neighboring coordinate they detect a partitioning and can treat the partitions like separate neighbor coordinates. Treating partitions separately can help solving both problems caused by area partitionings by *a)* forwarding messages addressed to the partitioned area to all known partitions or even forwarding all messages going through the partitioned area to all known partitions and by *b)* forwarding broadcast or unicast deliveries overheard from one partition to all other known partitions of the area.

4) Collecting connectivity information: Our final solution to deal with holes, the **collection of connectivity information** over time, takes advantage of the mobility of the client devices by allowing them to collect information about available and missing connections in the network.

The mobile client nodes can take advantage of the beacon messages the sensor nodes periodically exchange. Their information can be used to validate or invalidate connections between symbolic coordinates in the map stored by the client node. Together with the information from feedback messages received after failed routing attempts, this data can provide an up-to-date view of the sensor network and is able to support the client in communicating successfully with the network.

In principle, by visiting large parts of the area of the sensor network, a mobile client node could also learn the complete map information it needs for sending queries to the sensor network without ever loading a symbolic topology graph from the external location model.

IV. Prototype Implementation

We have implemented a prototype of our symbolic routing approach for Tmote Sky sensor nodes running the widely-used TinyOS 2.0 operating system. The complete system including all communication components and a simple prototype application logic consumes about 14800 bytes of the 48 kilobytes program memory and less than 1 kilobyte of the 10 kilobytes of main memory. As mobile client nodes we use Linux PDAs from Sharp (Sharp Zaurus SL-3200) that allow us to run both native and Java-based applications interacting with the sensor network. Our prototype client nodes communicate with the sensor network using a Tmote Sky sensor node connected to the PDA over USB as a bridge node. For assigning symbolic coordinates to the sensor nodes we use the application described in [20].

V. Evaluation

A. Simulation Setup

To learn more about the properties of our algorithm and to evaluate its behavior in a larger scenario we implemented it as an extension to the ns-2 network simulator. The ns-2 simulator provides us with more flexibility concerning the heterogeneity of the network than TOSSIM, the TinyOS sensor network simulator. We used TOSSIM to test our TinyOS implementation and to verify the results obtained with ns-2.

Neither a purely random distribution of nodes in the simulation area nor a uniform distribtuion of nodes are a good representation of reality and can lead to extreme results in individual scenarios. For our simulations, we opted for a mixture of both models using uniform distribution but adding an additional random factor to move the nodes between 0 and 20 meters away from their originally assigned position.

For the following simulations, we use an area of 36x36 meters which we divide into between 15 and 100 rectangular symbolic areas with a random layout. For our sensor nodes we decided for a maximum communication distance of 7 meters, a value obtained by our experiences with sensor nodes in office scenarios, using the Two Ray Ground radio model. In our experiments we also vary the number of nodes. Results are shown from a low density with 50 static sensor nodes (corresponding to 0.039 nodes per square meter) up to very high densities with 400 sensor nodes (corresponding to 0.309 nodes per square meter). Each situation of all experiments was repeated 100 times with randomly generated scenarios. We use 40 mobile client nodes that send query messages to randomly selected destination areas every 2 seconds. The client nodes are placed randomly and are moved every 5 seconds in order to cover a large set of possible communication pairs.

B. Evaluation Results

This section analyzes the performance of symbolic routing in sensor networks both for our basic approach and the various extensions that deal with holes and partitionings.

One important evaluation metric is the success rate, defined as the percentage of messages sent by one of the client nodes and successfully delivered at the destination area. Besides the success rate of communication, another important evaluation metric that allows to assess the costs for using symbolic routing is the so-called stretch, the average ratio between the real path length used and the shortest path length possible. A stretch value of 1.0 is, therefore, optimal.

1) Success rate of the basic approach: Fig. 4 a) and b) show the percentage of expected connectivities between symbolic areas that are missing in the node communication graph and thereby illustrate the importance of being able to deal with communication and coverage holes. Even in our scenarios

with 400 nodes, as an example of a very dense topology, the percentage of missing connectivities caused by communication and coverage holes grows to over 10 percent when increasing the number of symbolic areas up to 100. Fig. 4 c) illustrates that empty areas are responsible for a large portion of these missing connectivities.

Fig. 5 shows the resulting success rate for both sparse and dense node populations when we vary the number of symbolic areas and neither a message recovery nor any of the described prevention mechanisms is used. Only the densely populated scenarios are able to maintain acceptable success rates when increasing the number of symbolic areas. In the other scenarios the success rate quickly declines with the sparse topologies already starting at quite low levels.

2) Preventing holes: Adding knowledge about communication and coverage holes to the mobile nodes – our simplest method for preventing the occurrence of holes on the communication path – directly impacts the success rate of communication: the more knowledge a node possesses about holes the higher is the success rate it is able to achieve when sending messages to random destination nodes. An analysis of the more advanced problems related to this (e.g., costs for storing and processing this information on the mobile nodes) is out of the scope of this paper.

Another option for preventing holes is to use node density information to estimate the success probabilities of route alternatives. We calculate a density measure d_a for a symbolic area a by dividing the node density of the area a and the node density of the complete network area using the following equation: $d_a = \frac{numNodes_a \cdot area_{total}}{numNodes_{total} \cdot area_a}$, where $numNodes_a$ is the number of sensor nodes located in the symbolic area a and $numNodes_{total}$ is the total number of nodes in the complete network area. $area_a$ is the size of the geographic area represented by a and $area_{total}$ is the total size of the network area. This gives us a density value larger than 1 if the node density lies above average and smaller than 1 if it lies below average. We then use this density to calculate a path metric m_p for a path $p = a_1, a_2, \ldots, a_n$ (where $a_1, a_2, \ldots, a_n$ are the symbolic areas on path p) using the following function:

$$m_p = \frac{1}{(d_{a_1})^g} + \frac{1}{(d_{a_2})^g} + \ldots + \frac{1}{(d_{a_n})^g} \qquad (1)$$

The metric considers both the path length (by adding up the cost values of the individual areas) and the density of the areas. The influence of these two factors can be controlled using the density weighting factor g with $g \geq 0$. In our experiments we use $g = 1.0$, $g = 2.0$ and $g = 100.0$. Using $g = 0.0$ corresponds to the routing without using density information.

Fig. 6 a) and b) compare the resulting success rate for different density weighting factors for both 150 and 250 nodes. Using density information has the desired effect of increasing the success rate of communication. Particularly interesting is that it prevents the decrease of the success rate when increasing the number of areas. This can be explained with the higher quality of the density information available when using symbolic areas of small sizes.

As expected, increasing the density weighting factor can only improve the success rate within certain limits so that the results for a density weighting factor of 100.0 are only slightly better than the results for the factor of 2.0.

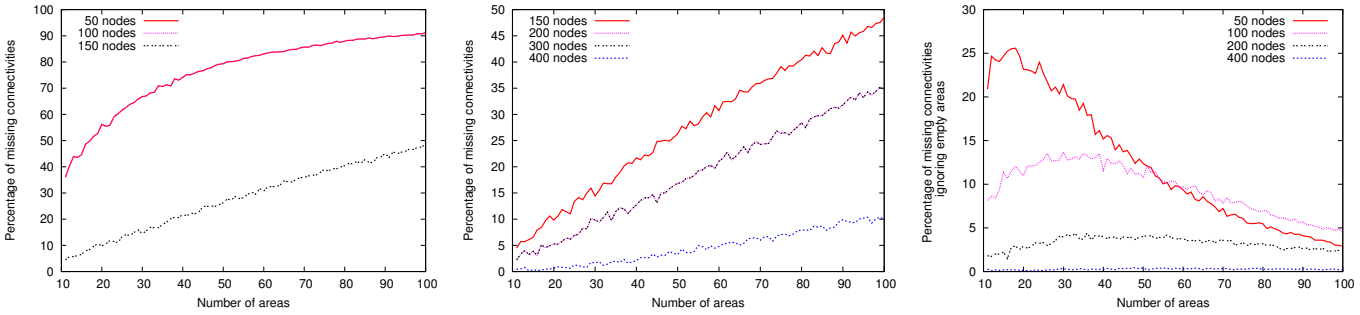
3) Recovering from holes: With the help of feedback messages it is possible to almost reach a 100% success rate over time (when ignoring message loss due to collisions and the like) assuming that the network is not partitioned. How long it takes for a mobile node to arrive at this reliability level mainly depends on the rate of messages it sends to different receivers. Due to space limitations we do not analyze the different combinations here but concentrate on recovery using local symbolic broadcasts.

We analyzed the number of broadcast steps required to recover from a communication or coverage hole by randomly selecting 1000 communication paths for each scenario and searching for the minimum number of broadcast steps for each hole appearing on these paths. Fig. 7 shows the percentage of holes recoverable with a maximum of 3 broadcast steps for scenarios with 50, 100 and 150 nodes. For scenarios with more than 150 nodes, the proportion of holes recoverable by one broadcast step quickly approaches 100%.

The results might look discouraging for sparse topologies like for the scenarios with only 50 nodes. However, it is only possible to cover a very small number of areas with 50 nodes and the results must be interpreted accordingly. With 25 symbolic areas – resulting in 2 nodes per area on average – around 70% (90%) of holes are recoverable in one (two) broadcast step(s). This is a promising result considering that some symbolic areas still are without coverage due to our random deployment of nodes.

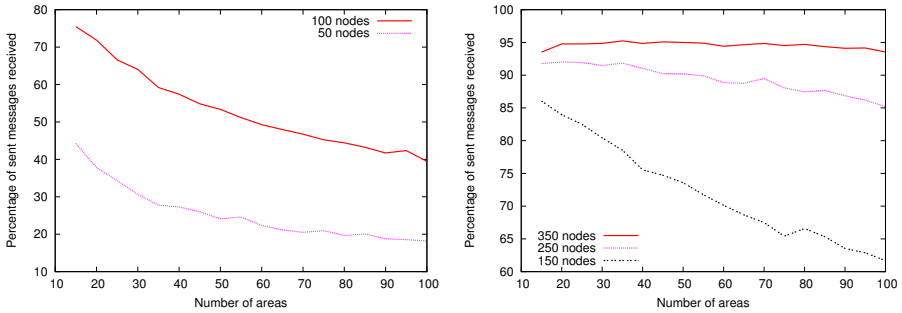
Fig. 6 c) then shows the average success rate when using a maximum of 1 broadcast step for recovering from communication or coverage holes. The positive effect compared to the results without recovery is considerable for all three node densities (compare Fig. 5). However, as expected, in the sparse scenarios with 50 nodes it still declines sharply with an increasing number of areas. Not reaching a 100% success rate with 150 nodes despite the results shown in Fig. 7 can be attributed in large parts to moving client nodes that try to send queries to neighbors they have lost the connection to before receiving beacon messages from their new neighbor nodes.

4) Recovering from area partitionings: To assess the impact of the area partitioning problem we first investigated how often this occurs in different scenarios. Fig. 8 (a) shows the average number of area partitionings that can be found depending on the overall number of areas the simulation area is split into. The ability to cope with partitioned areas is obviously less important for dense topologies and for scenarios with symbolic areas that are small compared to the transmission range of the sensor nodes. Particularly interesting is the observation that the curves are not monotonically decreasing with an increasing number of areas but possess a maximum (e.g., clearly visible for the case of 150 nodes at an area number of around 40). This is due to the fact that for very small numbers of areas only few partition candidates exist whereas for a large number of areas



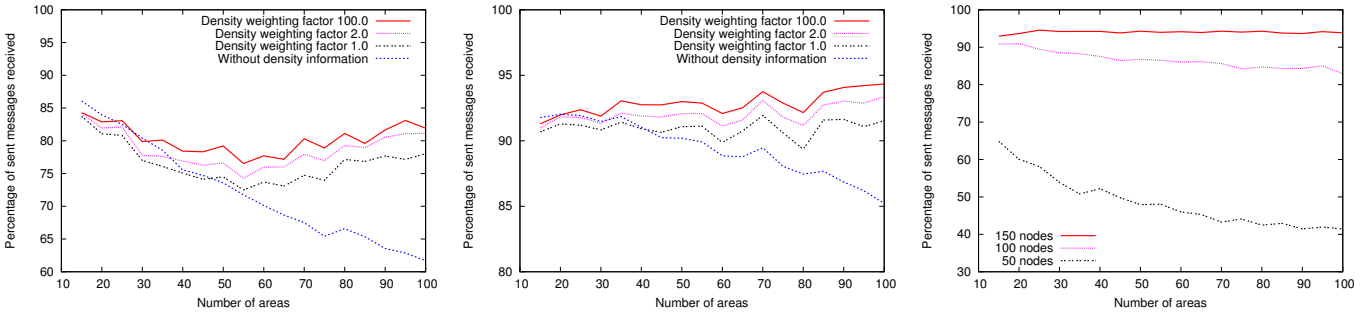
(a) Sparse topologies (b) Dense topologies (c) Ignoring empty areas

Fig. 4. Percentage of missing connectivities



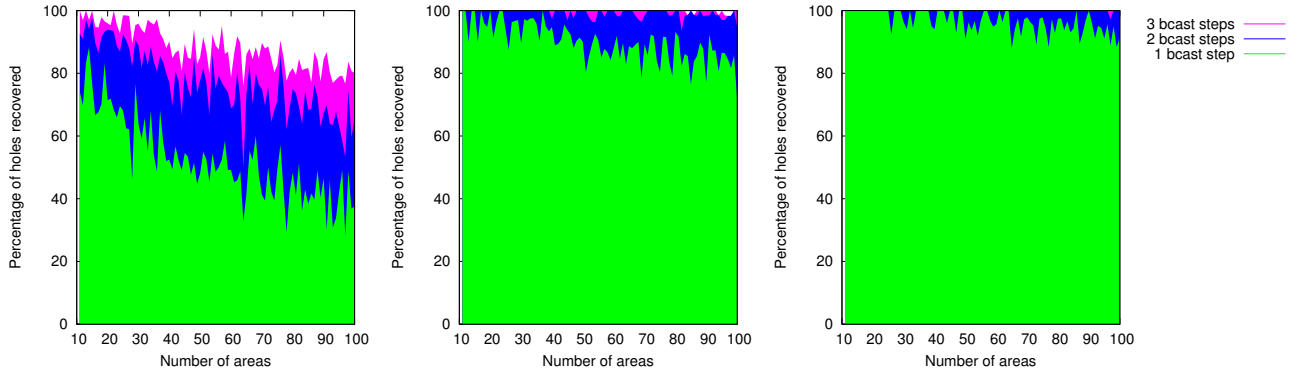
(a) Sparse topologies (b) Dense topologies

Fig. 5. Average success rate without recovery mechanisms



(a) Using density information, 150 nodes (b) Using density information, 250 nodes (c) With a maximum of one broadcast step

Fig. 6. Average success rate



(a) With 50 nodes (b) With 100 nodes (c) With 150 nodes

Fig. 7. Percentage of required broadcast steps for hole recovery

partitioned areas tend to get split up into different symbolic areas.

The solution for the area partitioning problem presented in Section III works for cases where the partitions are connected through only one intermediate symbolic area. Fig. 8 (b) shows for which percentage of the partitionings this is the case. Sparse topologies not only tend to have many area partitionings but are also only able to recover a very limited proportion of them using our solution. However, already in the scenarios with 100 nodes it is possible to recover a very large percentage of partitionings. Even more dense topologies can recover partitionings in the vast majority of cases. Here the question is whether the overhead of a recovery solution is justified considering the small number of partitionings expected to occur in practice.

5) Connectivity information collection: We also investigated to what extent and how fast mobile client nodes can collect connectivity information by listening to beacon messages while moving in the network area. Our analyses showed that a considerable part of the connectivity information is learned in a relatively short time. However, the learning curve also levels off quickly so that a complete knowledge of the connectivity information can only be expected – if at all – after a very long time. Unfortunately, providing general results is difficult as such an analysis is influenced by many external factors specific to the particular situation (e.g., node mobility or floor plan characteristics). Due to space limitations, we refrain from showing further results on this issue here.

6) Stretch: Fig. 9 shows the average stretch of our symbolic routing depending on the number of areas for scenarios with 150, 250 and 350 nodes (a) without and (b) with path look-ahead. The stretch for the cases with path look-ahead is caused by the purely local path length optimization of the algorithm, i.e., each node knows the shortest paths to its neighboring coordinates but not the shortest paths to all symbolic areas in the network. The larger stretch for cases without path look-ahead is due to possible short cuts the algorithm does not take.

We have also measured the effect of using density information on the average stretch. The results are shown in Fig. 9 c) for scenarios with 250 nodes. Clearly, selecting "safer" paths using density information increases the average path length. However, even for the very high density weighting factor of 100.0, the path length overhead hardly grows over 50%.

To set the shown results into perspective, Table I compares the average stretch achieved by the geographic routing algorithm GPSR with the average stretch of symbolic routing when communicating in the same scenarios. For GPSR the stretch does not depend on the number of areas as it does not use symbolic areas but geographic coordinates for its routing decisions. While the path length overhead for our symbolic routing is larger than for GPSR, we still consider an overhead of around 30% (or below 50% when using density information) compared to the shortest path as reasonably small considering the limited amount of position and topology information we are using.

TABLE I
AVERAGE STRETCH OF GPSR AND SYMBOLIC ROUTING

Number of nodes	Average stretch		
	GPSR	Symbolic routing without look-ahead	Symbolic routing with look-ahead
150	1.10	1.31 - 1.58	1.23 - 1.28
250	1.14	1.31 - 1.65	1.19 - 1.28
350	1.16	1.32 - 1.66	1.19 - 1.28

C. Discussion

Our simulations have shown that symbolic routing in sensor networks is possible with relatively low overhead compared to the optimal path and a promising success rate. Dealing with inevitable holes and partitionings is possible with the methods presented in this paper.

The simulation results support our decision to support both local symbolic broadcasts and the sending of feedback messages in our implementation. While local broadcasts are very effective in most cases, they can become expensive for very sparse topologies (see Fig. 7 and Fig. 6 c)) without being able to circumvent all holes. Feedback messages give the message sender all information and control about the message transfer process. This is advantageous both in sparse scenarios and when the client node expects to send multiple messages to the same destination area.

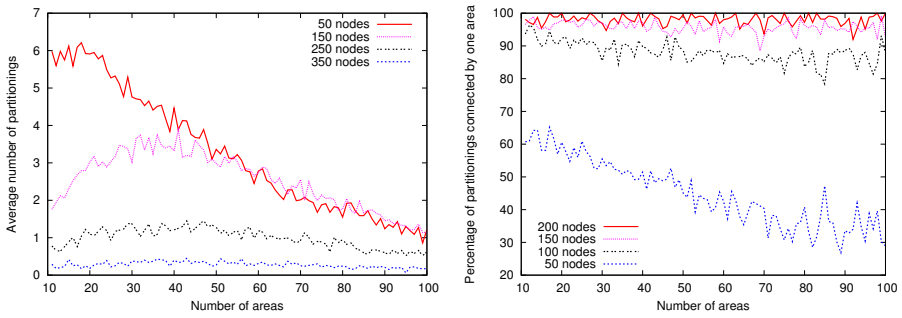
Using density information has also been shown to be worthwhile for scenarios where such data is available. Using density information is particularly attractive as it does not require any implementation on the sensor nodes and therefore does not incur any runtime overhead for processing or storing data on these nodes.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a novel routing approach for heterogeneous networks consisting of sensor nodes and mobile pervasive computing devices. Routing is based on symbolic coordinates and the routing task is split among the mobile devices that specify a symbolic source route and the static sensor nodes that realize the node-to-node routing based on this source route.

We have shown that our approach works in a variety of scenarios with different node densities. It generates a small overhead below 30% on the path length and is able to achieve high success rates with the help of different failure prevention and recovery mechanisms. One important advantage of the protocol is its flexibility concerning the amount of topology information known to the clients which allows using it in various types of application scenarios.

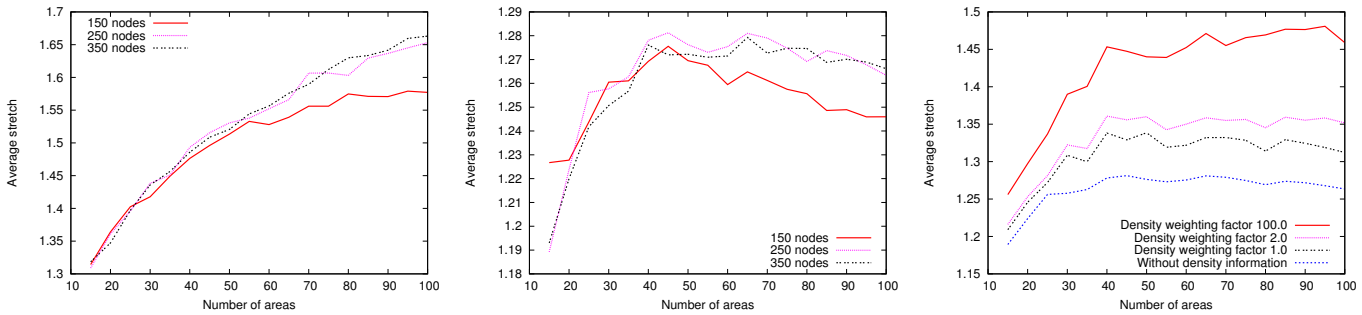
For future work we plan to continue our work on developing solutions for a **(semi-)automatic assignment of symbolic coordinates** to the sensor nodes in a network. Another important question is how to use symbolic coordinate information to support **topology control** in sensor networks, e.g., for managing the transmission power level of the sensor nodes. In addition to potential energy savings, a topology control mechanism is also very important for our approach to limit the amount of neighborhood state that must be managed on the individual sensor nodes. We also plan on implementing



(a) Number of partitionings

(b) Percentage of partitionings connected by one area

Fig. 8. Analysis of partition recovery



(a) Without look-ahead

(b) With look-ahead

(c) Using density information, 250 nodes

Fig. 9. Average stretch

more advanced **solutions to the area partitioning problem** that help delivering messages despite partitionings in a large fraction of cases. We are going to to **integrate symbolic routing with routing metrics** to deal with different link qualities. After successfully applying symbolic location models to sensor networks we also plan to investigate how **hybrid location models** that combine symbolic coordinate data with geographic information can be used.

REFERENCES

[1] A. Woo, T. Tong, and D. Culler, "Taming the underlying challenges of reliable multihop routing in sensor networks," in *Proc. of the 1st int. conf. on Embedded networked sensor systems*, 2003.

[2] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Elsevier Ad Hoc Networks*, vol. 3, pp. 325–349, 2005.

[3] B. Brumitt and S. Shafer, "Topological world modeling using semantic spaces," in *Workshop Proc. of Ubicomp 2001: Location Modeling for Ubiquitous Computing*, 2001.

[4] C. Jiang and P. Steenkiste, "A hybrid location model with a computable location identifier for ubiquitous computing," in *Proc. of the 4th Int. Conf. on Ubiquitous Computing*, 2002.

[5] T. Drosdol, D. Dudkowski, C. Becker, and K. Rothermel, "Efficient indexing of symbolic location information," in *Proc. of the 2nd Workshop on Context Awareness for Proactive Systems*, 2006.

[6] C. Becker and F. Dürr, "On location models for ubiquitous computing," *Personal Ubiquitous Computing*, vol. 9, 2005.

[7] S. P. Fekete, A. Kröller, C. Buschmann, S. Fischer, and D. Pfisterer, "Koordinatenfreies Lokationsbewusstsein," *it - Information Technology*, vol. 47, pp. 70–78, 2005.

[8] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-based ad hoc routing," *IEEE/ACM Trans. Netw.*, vol. 14, no. 3, pp. 479–491, 2006.

[9] M. B. Yassein, M. Ould-Khaoua, and S. Papanastasiou, "On the performance of probabilistic flooding in mobile ad hoc networks," in *Proc. of the Int. Conf. on Parallel and Distributed Systems - W'shops*, 2005.

[10] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (dsdv) for mobile computers," in *Proc. of the conf. on Comm. architectures, protocols and applications*, 1994.

[11] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," in *WMCSA '99: Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*, 1999.

[12] V. D. Park and M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proc. of the 16th Conf. of the IEEE Computer and Communications Societies*, 1997.

[13] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*. Kluwer Academic Publishers, 1996.

[14] Z. J. Haas, "A new routing protocol for the reconfigurable wireless networks," in *Proc. of the IEEE Int. Conf. on Universal Personal Communications*, 1997.

[15] L. Blazevic, S. Giordano, and J.-Y. Le Boudec, "Self Organized Terminode Routing," *J. of Cluster Computing,*, vol. 5, 2002.

[16] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proc. of the 6th Int. Conf. on Mobile computing and networking (MobiCom '00)*, 2000.

[17] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Lazy cross-link removal for geographic routing," in *Proc. of the 4th int. conf. on Embedded networked sensor systems*, 2006.

[18] M. Caesar, M. Castro, E. Nightingale, G. O'Shea, and A. Rowstron, "Virtual ring routing: Network routing inspired by DHTs," in *Proc. of the ACM SIGCOMM 2006*, 2006.

[19] T. Fuhrmann, P. Di, K. Kutzner, and C. Cramer, "Pushing chord into the underlay: Scalable routing for hybrid manets," Fakultät für Informatik, Universität Karlsruhe, Tech. Rep., 2006.

[20] M. Gauger, P. J. Marrón, D. Kauker, and K. Rothermel, "Low overhead assignment of symbolic coordinates in sensor networks," in *Proc. of the Int. Conf. on Wireless Sensor and Actor Networks (WSAN 2007)*, 2007.