Scenario-based Routing for Sensor Networks Applied to Ambient Navigation Assistance

Violeta Felea
Computer Science Department
University of Franche-Comté, Besançon - France
Email: vfelea *at* lifc *dot* univ-fcomte *dot* fr

Kamal Beydoun
Faculty of Science I
Lebanese University, Hadath - Lebanon
Email: kamal *dot* beydoun *at* ul *dot* edu *dot* lb

***Abstract*—Wireless sensor networks can be used in ambient navigation assistance systems since they form a bridge between the physical world and the need of environment awareness. Navigation needs to cover both static and dynamic obstacle negotiation in route planning. Our previous work on the ZHRP protocol over sensor networks deals with routing information infrastructure which can be exploited in this application context.**

In this paper, we evaluate extensively the performance of this protocol concerning route discovery scenarios. Both routing information construction and route discovery scenario were implemented on top of the J-Sim simulator. Two main metrics, critical in sensor networks, were estimated: the network lifetime - and the memory size - both theoretical and experimental. We compare performance of ZHRP with RIP. Analytical expressions for the memory size needed by the routing data structures give indications to calibrate the network infrastructure parameters.

I. Challenges in embedded applications oriented navigation assistance

One emerging technology used to analyze and retrieve information on real-world phenomena is based on embedded devices, among which sensors stand out. Particular scenarios in ambient assisted living concern navigation systems. Similar to GPS tools, the challenge of route discovery here is more complex because information on the environment necessary to integrate is both static and dynamic.

Route discovery in sensor networks is generally based on dissemination techniques which are too expensive in terms of communication. Moreover, existing solutions make particular hypothesis: nodes can be geo-localized [1], the transmission power control can be dynamically adapted [2] and is heterogeneous [3], or centralized algorithms are applied [4]. These assumptions are too restrictive and costly to be applied in real wireless sensor networks, characterized by a high level of decentralization, low resources and small storage capacity and requiring low cost operations (of both implementation and maintenance).

The construction of adaptive routes, context-dependent in terms of both dynamic and static information, is less developed in literature. For example, in research on designing electronic devices for the aid of the visually impaired, a high percentage of the proposed systems are focused either on obstacle avoidance or on orientation. Obstacle negotiation is the ability to move depending on the knowledge of immediate objects and obstacles, of the ground shape and of dangers both moving and stationary. NavBelt [6] proposes obstacle negotiation. Orientation is the process of locating targets and of constructing routes to them. Projects proposing orientation are UbiBus [7], Noppa [8]. Navigation, which includes both obstacle negotiation and orientation, takes into account the obstacle information in route discovery and planning. Drishti [9] is an example of wireless pedestrian navigation system for the visually impaired and disabled. Compared to Drishti, based on a set of different technologies (GIS database-based GPS system, ultrasound positioning, beacons) which integrates only static obstacles in the route planning, our approach of navigation system is based on top of sensors only, detecting and managing both static and dynamic obstacles. Moreover, our solution can be applied to different scenarios - not only for the visually impaired - which require adaptation in dynamic contexts.

Another example concerns the technology for vehicle orientation, which is currently exclusively oriented towards route discovery based on static information issued from plans.

Whereas geo-localization solutions give good results at a very large scale and for static system behavior, they are less efficient when dynamic, unforeseeable phenomena appear. This is the challenge we address: efficiently provide route discovery mechanisms and dynamically adapt them based on contextual information, both static and dynamic information being retrieved using embedded devices, more particularly, sensors, disseminated in large areas, covering town areas, at the most. We present the assumptions of the assisted navigation applications we address in section 2, our proposal of route discovery protocol in section 3 and its evaluation in section 4. We finally conclude.

II. Application architecture

Assisted navigation may concern different configurations (indoor, outdoor), all based on the following features (see figure 1):

- several target areas are identified (for example, railway station, post office, campus, etc.),
- several targets can precisely indicate destinations in a specific area (department of computer science on the campus, railway station platform, ATM in a post office, etc.),

- some other intermediary stops are relays, but not necessarily targets.

Requests from users in navigational systems have the following form "Which is the route to take in order to reach a specific target in a particular area?".

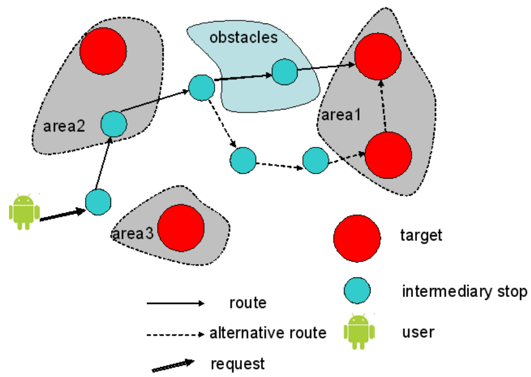

Fig. 1. Application features

This corresponds to the route discovery issue addressed in networks. We may implement standard solutions of shortest paths in graphs (modeling network structure) whose nodes represent the targets and the intermediary stops. Meanwhile, these techniques can not be applied because they are centralized and require global network view. Moreover, adapting route construction to environmental context is difficult to achieve. Our solution is focused on the use of wireless sensor networks. Sensors, assigned to disjoint *zones*, correspond to both targets and intermediary stops (see figure 2). Sensors are needed in order to retrieve and to relay dynamic information. RFID may be used for targets, but this choice would reduce their functionality in the network. We use here sensor networks in a particular situation, requiring node addressing, rather than data addressing more commonly used in this kind of networks.

The answer to the previous question may be given by querying the closest node (target or intermediary) which will operate route discovery: find a set of nodes, forming a chain (each one being able to communicate with its predecessor and its successor in the chain). The user will be informed of the next node to reach along with his progress (see successive steps in figure 2). Consequently, we may consider the user as the sink in a standard sensor network configuration, having the particularity of being mobile.

When dynamic information (e.g. obstacles) prevents the user to follow the next intermediary node, some other route should be proposed (figure 1 shows alternative route - dotted arrows).

III. Route discovery

Embedded sensor systems in the environment, oriented to assist navigation, need more often large deployment. In this context, route discovery can not exploit traditional techniques like flooding, used for example in sensor networks. We propose table-driven pro-active approaches, inspired from ad-hoc wireless networks. Several reasons justify this choice:

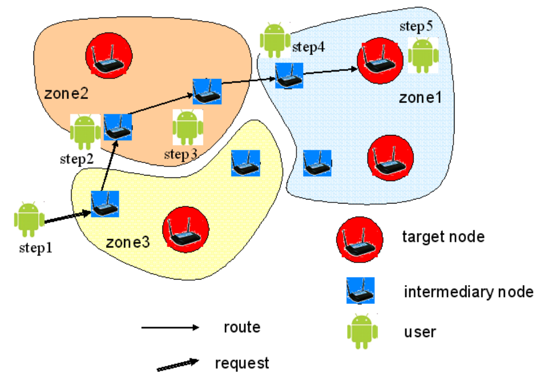

Fig. 2. Application architecture

- route availability - routes should be rapidly determined,
- static network architecture - no sensor node mobility is involved (only the sink node is mobile),
- nodes are GPS-free,
- nodes may be IP-like addressed - even though nodes do not have an IP-address assigned, some of them can be considered as targets, thus being clearly identified in the network.

Table-driven solutions for routing in largely-deployed networks need network structuring to diminish node's view concerning potential destinations. We are using the ZHRP (Zone-based Hierarchical Routing Protocol) routing approach [10], based on network partitions into zones. We briefly present, in III-A and III-B, the principles which are thoroughly explained in [10]. We focus in this paper on the route construction algorithm (that was not addressed previously), which exploits routing information. The concerned algorithm is described in the last part of this section.

A. Routing information

The idea of creating virtual network zones is not only motivated by the large number of network nodes. It is also justified by the application aim: route discovery concerning a particular target may not at all concern areas of some other far geo-localized target. Zones help giving a macroscopic view when trying to reach the target zone and a microscopic view when trying to reach the target inside the zone.

Zones are defined based on two parameters:

- randomly chosen *invitation nodes* (they are by default assigned to different zones),
- the *zone radius* in terms of number of radio-hops.

Invitation messages are initiated by the invitation nodes and broadcast to their neighbors, which operate the same. This propagation stops either because the number of consecutive hops issued from a same invitation node exceeds the zone radius, either because the node receiving the invitation message is already assigned to a zone. During this construction, two types of nodes are defined:

- *normal nodes* (labeled NORMAL) - used to relay messages inside a zone,

- *border nodes* (labeled BORDER) - used to relay messages between adjacent zones.

These nodes will dispose of different routing information, retrieved during the construction of the routing tables:

- the normal nodes have an intra-zone routing table, denoted *IntraZoneRT*. An entry in this table contains the following information:
  - the destination node (*destNdId*),
  - the next hop (*nextHopId*),
  - the metric (*M*) - computed in number of hops,
  - the type of the destination node, normal or border (*nodeType*),
  - moreover, if the destination node is a border node, the entry contains the list of neighboring zones (*neighZoneIds*).
- the border nodes have an intra-zone routing table, just like the normal nodes, and moreover, an inter-zone table, denoted *InterZoneRT*. One entry in this table contains
  - the destination zone (*destZoneId*),
  - the next zone (*nextZoneId*),
  - the zone metric (*zoneM*).

The zone metric is defined as the average length (in hops) of the longest path between any two nodes of the zone, and quantifies the maximum cost of crossing a zone.

The DV (Distance-Vector) algorithm is applied in both intra-zone and inter-zone routing table construction based on the hop metric inside a zone, and on the zone metric between zones.

B. Architecture implementation

The following steps are applied in the implementation of the navigation system using sensors:

1) set up the sensor network
   a) deploy the nodes (targets are already assigned to zones, called *target zones*)
   b) define the number of zones (including the target zones)
   c) choose the inviting nodes (one for each target zone randomly chosen among the targets of the same zone, and one randomly chosen among the other nodes)
   d) assign nodes to zones
2) construct routing information (see [10])
   a) intra-zone routing table
   b) inter-zone routing table
3) construct the route, depending on the route discovery query

The last step is detailed next.

C. Route construction

Node addressing for route construction purpose uses both $nodeId$ and $zoneId$ identifiers. This is particularly the case in embedded sensor navigation systems, where potentially addressed destinations are clearly identified.

Consider the scenario of identifying the route from a source node when sending a data packet to a particular destination node, *finalDestId*, assigned to the *destZoneId* zone. This leans on routing the packet from a source to a destination.

Packets exchanged during the data routing phase contain the following information: the source node, the next hop, the local destination node in the current zone, the final destination node, the zone of the final destination node, and the transmitted data.

The final destination defines the destination node to which data is sent. The local destination node is necessary when routing a packet into a zone that is not the zone of the final destination. When the packet gets to the zone of the final destination node, these two attributes have the same value.

If the source node is normal and the destination zone is directly adjacent to the current zone, it forwards the packet towards one border node, at the frontier of the destination zone; otherwise, it means that the packet needs to cross multiple zones, so a random border node is chosen as local destination and the packet will be sent to it. This border node will choose, depending on its *InterZoneRT* table, the appropriate zone across which the packet will be transmitted.

When the source is a border node, similar principle is applied (the algorithm is omitted here).

Generally, if the node receiving the packet belongs to the zone of the final destination, the *IntraZoneRT* is exploited; otherwise, if it is not a border node, it forwards the packet to the local destination node. If it is a border node to a different zone from the destination zone, the *InterZoneRT* is exploited to find the border node able to forward the packet to the destination zone.

Dynamic environment information can be taken into account by temporarily invalidating the next hop on the route. The current node forces the discovery of an alternative route, by sending the packet to one randomly chosen neighbor (excepting that for which the link is invalidated). This node will continue the route construction based on its own routing table.

Using this solution, dynamic information does not generate propagation of routing table modifications; therefore no additional overhead is involved.

IV. Route scenarios setup and evaluation

The J-Sim simulator[1] [11] has been our implementation choice for the simulation of the application scenario, because it allows large-scale simulations and offers the possibility of integrating customized routing protocol. Built upon the concept of autonomous component programming model, customized sensor environments can be defined (sensor's layers and the communication infrastructure). J-Sim is developed entirely in Java and uses Tcl as script language to facilitate scenario setups.

a) Scenario description: The route discovery scenario is the following: a number of requests (varying from 5 to 12) are generated every 60 milliseconds during 6 minutes

[1] http://www.j-sim.org/

of simulation. Requests concern route discovery between two nodes, randomly generated over the entire deployment area. Nodes receiving the requests apply the route construction algorithm presented previously towards randomly generated destinations.

b) Metrics: We intend to estimate the lifetime of the network in this particular scenario, as well as the scalability property of the routing algorithm. We do not differentiate here between different nodes (normal or border); we consider an average communication cost. Another critical aspect of sensor networks is resource management. Sensor nodes are memory constrained and the size of data structures needing storage should be minimized. Therefore, we evaluate the required memory size. Moreover, we compare these metrics to a well-known flat, table-driven approach, RIP, also built upon the DV algorithm.

A. Battery consumption

The first simulations were performed for 100 nodes deployed on an area of 300 square meters. The parameters for the zone partitioning (the zone radius and the number of zones) are both set to 5.

Energy consumption in data routing scenario is directly dependent on the number of received and sent messages (in average, the number of received messages is more important than the number of sent messages, because of the wireless communication pattern, based on broadcast rather than on unicast communications). Mica2 sensors drain more battery essentially because of low data rate in packet transmission.

Figure 3 shows the average percentage of the battery consumption in each node of the network, using the energy model of the TmoteSky and of the Mica2 sensors.

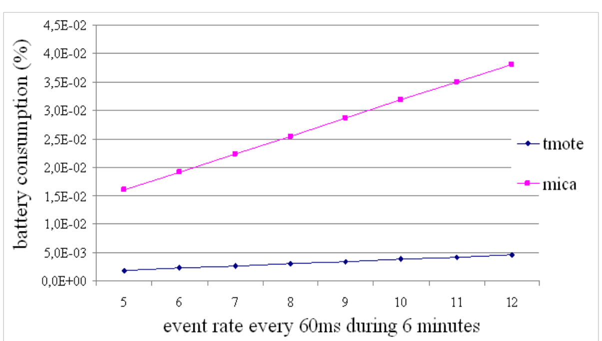

Fig. 3. Battery consumption during route construction scenario

These computations depend on the total number of received ($nbRM$) and sent ($nbSM$) packets (figure 4), according to the following formula:

$RC = \frac{(nbSM * RC_{Tx} + nbRM * RC_{Rx}) * packetSize}{dataRate} mAh.$

where $nbSM$ denotes the number of sent messages and $nbRM$ the number of received messages ; RC_{Tx} is the battery consumption in radio transmission, RC_{Rx} is the battery consumption in radio reception.

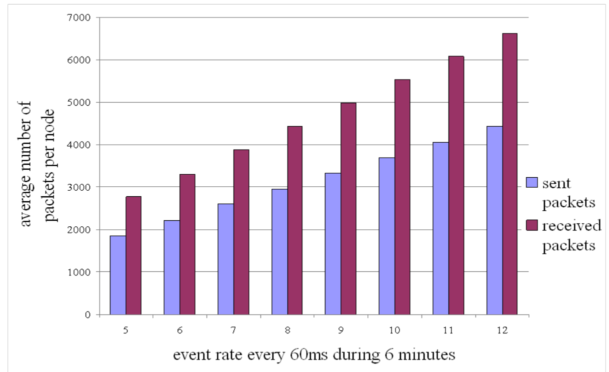

Fig. 4. Number of sent/received packets in the route construction scenario

B. Network lifetime and scalability

The network lifetime may be computed considering the moment when either the first sensor node or the last sensor node dies. More generally, when the sensing coverage is below a particular threshold, network is disconnected, moment which defines its lifetime. We consider here the worst case, identified by one node failure. Our route discovery scenarios are using randomly chosen source-destination node pairs, therefore our network lifetime is computed based on s average current draw. Border nodes are more intensely used than normal nodes, so they are generally depleted first. However, the routing principle is based on several border nodes per zone which may help keeping inter-zone connectivity active.

We do not consider here current draw for the network initialization. The network lifetime in the route construction scenarios described previously is based on the energy consumption shown in figure 3.

The network lifetime computed based on these consumptions varies from 26 to 11 days for the Mica2 sensor network, while that of the TmoteSky network varies from 218 to 91 days (see figure 5). The important gap between the two types of network is essentially caused by a small data rate of the Mica2 sensors compared to the data rate of the TmoteSky.

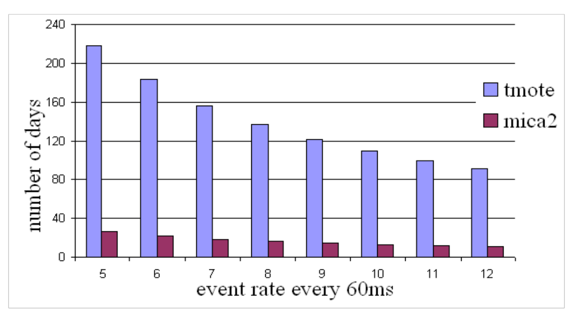

Fig. 5. Lifetime of Mica2 sensor and TmoteSky sensor networks

Actually, in a real world context, nodes heterogeneity in terms of technical features is present: indoor nodes (TmoteSky) and outdoor nodes (Mica2) are both deployed. We may infer the optimal number of indoor nodes and outdoor nodes based on both the lifetime metric (to be maximized) and the nature of the deployment environment.

While both RIP and ZHRP route discovery are based on the same concepts of routing tables and information dissemination on the DV basis, ZHRP hierarchical network view is meant to better adapt to network scale. Therefore, we measured the network lifetime (see figure 6) when proceeding to 3 random source-destination route construction queries every 60 ms for different node deployments on an area of 1500 square meters. The network size was varied from 100 to 292 nodes (the maximum size supported by the J-Sim simulator for the RIP protocol), R is the zone radius and nZ is the number of zones.

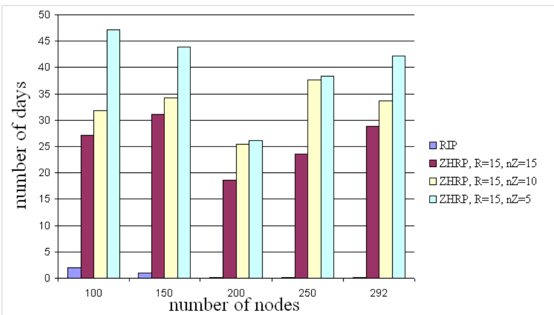

Fig. 6. Network lifetime using RIP and ZHRP

Network lifetime when using RIP varies from 0.2 to 2 days, because of the heavy control traffic, while ZHRP gives good performance, making nodes last from 46 to 118 days. In ZHRP, the choice of a small number of zones (the nZ parameter) generally generates longer network lifetime than greater values; this remark is no longer true for the memory size parameter, as shown next.

We cannot infer any pertinent information out of the comparative analysis between network lifetime when different node deployments are considered. In these cases, both node positions and network zone structure are different.

C. Storage space

One other important evaluation metric for the proposed system is the storage space. Using table-driven algorithms may need important memory space in the context of largely-deployed networks. This complexity has already been reduced by the two-level routing tables which are constructed. We estimate the space complexity, in terms of number of bytes occupied by the involved data structures.

c) Analytical expressions: The formula computing the size (in bytes) for the routing data structures in ZHRP is given next, for *N* deployed nodes, when *nZ* zones are constructed, each zone having in average *nB/nZ* border nodes (where nB is the total number of border nodes). In contrast, RIP uses 8*N bytes for its routing table in every node.

ZHRP	Zone Construction	Routing Table Construction
Border	2(4+6(nB/nZ))	2(3nZ+1+6N/nZ)
Normal	4*2	2(1+6N/nZ)

These analytical expressions give indication on the memory size needed by the routing data structures in ZHRP, depending on the total number of deployed nodes and the number of zones (see figures 7 and 8 - in which nB was arbitrarily chosen to 20% of the total number of deployed nodes). Graphs show here that the choice of great values for the number of zones is less memory consuming than small values. Correlated with previous results, this shows the necessity of a trade-off in the choice of this parameter, to both maximize network lifetime and minimize required memory capacity.

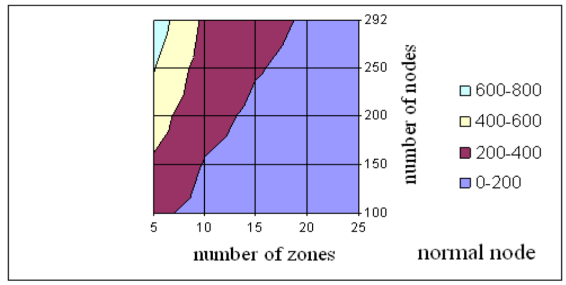

Fig. 7. Analytical results - required memory size (in bytes) for ZHRP normal nodes

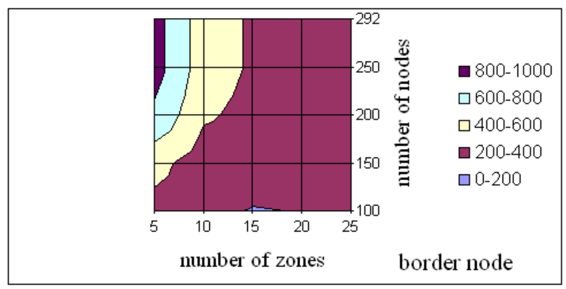

Fig. 8. Analytical results - required memory size (in bytes) for ZHRP border nodes

Optimal number of zones when expecting minimum memory size for the routing data in border nodes is obtained when the number of zones equals $\sqrt{2(N+nB)}$, where N is the number of deployed nodes and nB the total number of border nodes. The expression for the memory size of data structures in a normal node is asymptotically decreasing to 10 when the number of zones is important (hyperbola having the abscissa $x = 10$ and the ordinate $y = 0$). Meanwhile, an important number of zones generates too many entries in the border tables, and therefore too much overhead due to control packets exchanged between zones in the construction of the complete border tables.

We emphasize that completing intra-zone routing tables for n nodes is less costly than completing the inter-zone routing table for n zones, because in inter-zone routing table construction, packets need to be sent throughout zones. Consequently, we can infer the upper bound for the number of zones which is $\sqrt{2(N+nB)}$. A lower bound for the number of zones can be established, depending on the maximum memory capacity which can be used to store routing data.

d) Experimental results: We experimentally compute the size needed for the data structures used during the routing table construction, in both ZHRP and RIP (see figures 9 and 10). The network parameters are the same as in the previous scenario setup, meant to estimate the lifetime of exclusively Mica2 sensor networks.

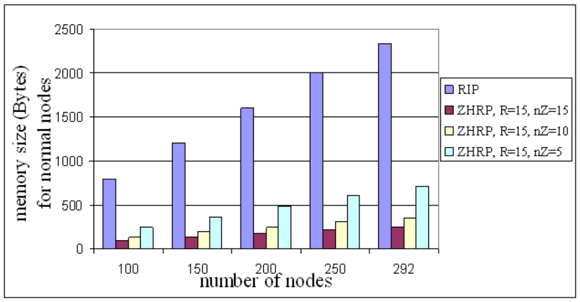

Fig. 9. Required memory size (in bytes) for ZHRP normal nodes, compared to memory size required for the RIP nodes

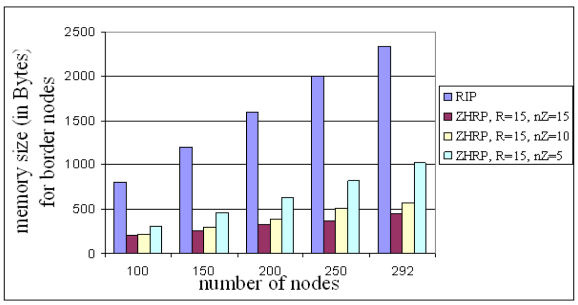

Fig. 10. Required memory size (in bytes) for ZHRP border nodes, compared to memory size required for the RIP nodes

Memory size requested by the normal nodes in ZHRP linearly depends on the number of nodes per zone, varying from 90 to 711 bytes (see figure 9). Compared to normal nodes, border nodes use up to a 2 factor more memory than normal nodes (see figure 10). One particular reason explaining this behavior is that experiments were done while increasing the number of nodes deployed on the same area. Consequently, the increased density of the network has an impact on the number of border nodes per zone, and thus on the size of data stored by the border nodes. Meanwhile, compared to RIP, when considering the border nodes - which is the worst case in terms of memory size - we remark a 5 down to 2.5 decreasing factor. Another encouraging aspect concerns low occupation of the total memory of a Mica2 sensor (4KB), generally up to 15%, for reasonable values for the number of zones.

V. Conclusions

Wireless sensor networks are recently quite widely employed as a bridge between the physical world and the need of environment awareness. Generally used to monitor and analyze large-scale, real-world phenomena, sensor networks are proposed here as technological solution for ambient navigation assistance. Navigation includes both obstacle negotiation and orientation (route planning). The deployment area is partitioned in zones, every sensor being assigned to one zone, on the basis of hop counts within a predefined zone radius. All nodes can be addressed (including targets in the navigational system). Route construction exploits two-level routing tables, built in order to minimize response time for route discovery requests. Mobile obstacle negotiation is managed by temporarily invalidating links to neighboring nodes which forces discovery of alternative routes. Sensor systems are known resource-constrained, in both memory and energy. Therefore, two metrics, space complexity and network lifetime are estimated for simulations performed using J-Sim implementation. Up to 15% of the Mica2 sensor memory size is necessary to store routing information. In a route discovery scenario simulating 5 requests every 60 ms over a 100 node network, a TmoteSky sensor network lasts for 218 days, while the exclusive use of Mica2 sensors makes network go on for only 26 days. When route discovery request rate is lower - 3 requests every 60 ms - in larger deployment area and networks, network lasts for up to 47 days using ZHRP, while RIP drains node energy after 2 days.

Future work will focus on the evaluation of the route construction algorithm in the context of dynamic environments (simulating obstacles).

References

[1] H. Tian, J.A. Stankovic, L. Chenyang, and T. Abdelzaher. SPEED: a stateless protocol for real-time communication in sensor networks. In *Proc. of Int. Conf. on Distributed Computing Systems*, pages 46–55, 2003.

[2] O. Chipara, Z. He, G. Xing, Q. Chen, X. Wang, C. Lu, J. Stankovic, and T. Abdelzaher. Real-Time power aware routing in sensor networks. In *Proc. IEEE International Workshop on Quality of Service (IWQoS)*, pages 83–92, New Haven, USA, 2006.

[3] W.R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient Communication Protocol for Wireless Microsensor Networks. In *Proc. of the IEEE Hawaii International Conference on System Sciences*, Vol. 2, p. 10, 2000.

[4] W.B. Heinzelman, A.P. Chandrakasan. An Application-Specific Protocol Architecture for Wireless Microsensor Networks. In *IEEE Transaction on Wireless Communication*, 1(4):660–670, 2002.

[5] K. Beydoun, V. Felea, and H. Guyennet. Wireless Sensor Network Infrastructure: Construction and Evaluation. In *Proc. of Int. Conf. on Wireless and Mobile Communications*, pages 279–284, France, 2009.

[6] S. Shoval, J. Borenstein and Y. Koren. The NavBelt-a computerized travel aide for the blind based on mobile robotics technology. In *IEEE Trans Biomed Eng.*, 45(11):1376-86, 1998.

[7] M. Banâtre, C. Couderc, J. Pauty, and M. Becus. Ubibus: Ubiquitous computing to Help Blind People in Public Transport. In *Proc. of the International Symposium on Mobile Human-Computer Interactions*, pages 310-314, Glasgow, Scotland, 2004.

[8] V. Ari and K. Sami. Towards Seamless Navigation. In *Proc. of the Mobile Venue*, Athens, Greece, 2004.

[9] L. Ran, S. Helal, and S. Moore. Drishti: An Integrated Indoor/Outdoor Blind Navigation System and Service. In *Proc. of the Second IEEE International Conference on Pervasive Computing and Communications*, pages 23–30, Orlando, Florida, 2004.

[10] K. Beydoun and V. Felea. Wireless Sensor Networks Routing over Zones. In *Proc. of 18th Int. Conf. on Software, Telecommunications and Computer Networks (SoftCOM)*, Split-Bol, Croatia, 2010.

[11] A. Sobeih, J.C. Hou, K. Lu-Chuan, Li Ning, Z. Honghai, C. Wei-Peng, T. Hung-Ying, and L. Hyuk. A simulation and emulation environment for wireless sensor networks. *IEEE Wireless Communications*, 13:104–119, 2006.