Supporting Navigation in Multi-Robot Systems through Delay Tolerant Network Communication [⋆]

Frederick Ducatelle, Alexander Förster, Gianni A. Di Caro and Luca M. Gambardella

Istituto Dalle Molle di Studi sull'Intelligenza Artificiale (IDSIA)
Galleria 2, 6928 Manno-Lugano, Switzerland
(e-mail: {frederick,alexander,gianni,luca}@idsia.ch)

Abstract: In this paper we study a problem of navigation in networked multi-robot systems. The robots are deployed in a confined area, where they move around and solve tasks. They communicate with each other through an infrared communication device, so that an ad hoc network is formed among them. Due to the limited range and line of sight nature of the infrared communication, this network has intermittent connectivity. The question we address is how a particular robot can use this network to find a target location that is indicated by another robot (e.g., the other robot has identified a task to be serviced by the searching robot). All other robots are involved in tasks of their own, and do not change their movements to help the searching robot find its destination. However, they do offer support by forwarding messages over the network. We propose a new algorithm based on routing in ad hoc and delay tolerant networks that can run on the network formed between the robots and provide navigation information to the searching robot. We evaluate the validity of our approach both in simulation and through an implementation on a group of 16 e-puck robots.

Keywords: Networked robotics, delay tolerant networks, robot navigation, network routing, infrared communication

1. INTRODUCTION

We address a problem of navigation in a networked multi-robot system deployed in a confined area. We study a situation where a robot A needs to find a location indicated by another robot B. This might for example occur when robot B has found a task that needs to be serviced at a given location, and robot A is idle and possesses the capacities needed to deal with the task. We assume that all other robots are busy with tasks of their own and do not alter their movements in order to help robot A with its navigation problem. They do however offer support through communication, by forwarding messages and gathering information over the network formed between the robots.

Communication between the robots takes place using an infrared range and bearing system (IrRB) (Pugh and Martinoli (2006); Gutiérrez et al. (2008)). This system consists of a number of infrared emitters and receivers placed all around the robot. It is able to transmit small amounts of data over short distances, as well as to estimate the relative range and bearing of the robots it is receiving data from. Precise data about the performance and precision of this system depends on its implementation and is discussed in sections 4 and 5. Using the IrRB system, the robots form a mobile ad hoc network among them. Due to the limited range and the line-of-sight nature of the communication device, connectivity is expected to be very intermittent. Especially in cases of sparse robot density or deployment in cluttered environments, the impossibility to establish end-to-end paths will be the norm rather than the exception. Therefore, algorithms and applications running on the network need to be delay tolerant (Fall (2003)).

In this paper we propose an algorithm that uses the communication network to gather information to support the navigation of individual robots to their target. The algorithm is inspired by ad hoc network routing protocols, but is adapted to work as a delay tolerant application in the intermittently connected network between the mobile robots: navigation information can travel both through telecommunication and through physical transportation on board of moving robots. The ability of the IrRB system to couple the reception of each data packet to relative location information about the sender allows us to link routing data to geographical navigation information. Moreover, the fact that infrared communication is only possible within direct line-of-sight ensures that communication links correspond to obstacle free paths for robots. The main novelty in this paper is that we offer a practical approach that allows mobile robots to assist each other in navigation while pursuing different goals and without

[⋆] This work was partially supported by the SWARMANOID project, funded by the Future and Emerging Technologies programme (IST-FET) of the European Commission under grant IST-022888. The information provided is the sole responsibility of the authors and does not reflect the European Commissions opinion. The European Commission is not responsible for any use that might be made of data appearing in this publication.

having to adapt their own movements. The fact that we explicitly account for intermittent connectivity of the network allows to deploy the system with a relatively low robot density or in complex environments (where line-of-sight communication is frequently blocked by obstacles).

The rest of this paper is organized as follows. First, we describe the related literature that is most relevant to our work. Then, we give a detailed description of our navigation algorithm. Next, we present some results from tests we carried out in simulation, and after that we describe some tests that we ran with real robots.

2. RELATED WORK

A number of publications have addressed the topic of communication aided robot navigation. The simplest form of such systems is one where a single moving robot is guided by a network of static nodes. These static nodes can be deployed by the robot itself, as in Batalin and Sukhatme (2003), or by an independent process, as in O'Hara and Balch (2004) (and later papers by the same authors). Localization can be done via GPS (Corke et al. (2003)), or through hop count and/or signal strength measures (O'Hara and Balch (2004); Witkowski et al. (2008)). Other approaches use a combination of mobile and static nodes. E.g., Sit et al. (2007) let mobile robots move to areas of low connectivity in the network of static nodes. Finally, other systems rely entirely on moving robots, which is the case most similar to our work. In Park et al. (2008) each robot makes a map of its local environment, and communicates it with other robots using multi-hop communication over a mobile ad hoc network. In Witkowski et al. (2008) mobile robots spread over an area and then choose fixed positions where they serve as beacons that form a communication network and support tasks such as search, localization and navigation. In Nouyan and Dorigo (2006), a swarm intelligence approach is proposed, whereby mobile robots physically form a chain to guide another robot towards a goal using visual cues. Payton et al. (2001) presents pheromone robotics, whereby robots spread out over an area and indicate the direction to a goal robot using infrared communication. Compared to the previously described approaches, this work is more closely related to ours. However, it requires robots to adapt their movements to cooperate in the search and does not have support to deal with intermittent connectivity. O'Hara (2003) presents an approach similar to the previous. In Sgorbissa and Arkin (2003) communication is used for local navigation: a robot is required to go to a sequence of goal locations and one or more explorer robots move around to help it using line-of-sight communications. Different from our system, this work requires robots that are dedicated to support navigation ("explorer robots"). Moreover, the authors only present results for relatively small groups of robots (one searcher and up to three supporting robots), and only in simulation. Finally, in Ducatelle et al. (2008) we have presented a robot navigation system based on an adaptive ad hoc routing algorithm. That approach has similar properties as the one presented here, but uses end-to-end paths between the searcher and the target robot (the higher the frequency paths can be established, the more effective the algorithm is), so it is difficult to use in cluttered environments or with a low density of robots. Moreover, the system presented here is simpler and requires less bandwidth, so that it can easily be implemented in robotic systems with limited capacities and communication bandwidth (such as the e-puck robots and their low bandwidth IrRB system that we used in our experiments; see section 5).

3. THE ROBOT NAVIGATION ALGORITHM

We use an ad hoc network routing algorithm to support robot navigation. Since communication goes over the IrRB system, distance information is available for each link in the network, and the routing algorithm can therefore search the path with the shortest physical travel distance to the destination. Our routing algorithm is based on Bellman-Ford routing (Bellman (1958)): each robot periodically broadcasts to its neighbors the best distance estimate it has available to each target robot, and a robot receiving an estimate from a neighbor adds to it the distance between itself and the neighbor to obtain its own distance estimate for the target. All distance estimates are labeled with a sequence number given out by the target robot, as is done in some ad hoc routing protocols (e.g., DSDV by Perkins and Bhagwat (1994) and AODV by Perkins and Royer (1999)). This allows to distinguish old from new information. Also, we provide mechanisms to deal with intermittent network connectivity: robots adapt their distance estimates as they move according to their odometry, so that estimates remain valid also when lack of network connectivity prevents the reception of new updates. In what follows, we describe the working of our navigation system in detail.

Tables and messages

Each robot maintains two tables: a routing table and a neighbor table. The routing table has one entry per target robot. Each entry contains three values: the target's id, a distance estimate, and a sequence number indicating the recency of the information. The information in this table is broadcast in periodic update messages (sent every 2 seconds in our experiments). Each update message contains also the id of the broadcasting robot. The neighbor table has one entry for each neighbor, i.e. for each robot a message was recently received from. Here, an entry contains the neighbor's id, a distance, a direction and a timestamp indicating when the last message was received from this neighbor. The distance and direction are obtained from the IrRB system at the reception of each message. The neighbor table is frequently checked to eliminate neighbors that have not been heard in a while (more than 2 update message periods).

Route requests

When a robot needs to find a target, it adds an entry to its routing table with the id of the target and sequence number 0 and distance 0. From then on, each periodic update message it sends out contains this entry. The sequence number 0 indicates that the entry is in fact a request towards the other robots to start gathering information. Robots receiving the request add it to their own routing table (without changing the sequence number or the distance) and forward it in subsequent periodic update messages. When the request finally reaches the target robot, also this one will add it in its routing table. The only difference in behavior for the target robot is at the moment of broadcasting its periodic update messages: when a robot finds its own id in its routing table, it increases the sequence number each time it includes it in an update message. This way, the sequence number is no longer 0, so it ceases to be a request, and information starts traveling back to the searching robot. Moreover, the increasing sequence number serves as a pulse from the target robot and allows to compare the relative recency of different routes. The distance value broadcast by the target remains 0 as in the request, as this is the correct distance to itself.

Processing update messages

When a robot A receives a periodic update message from a robot B, it first updates its neighbor table. If B is not yet in the table, it adds a new entry with the distance and direction obtained from the IrRB system; otherwise, it updates the distance and direction to neighbor B using a moving average (with discount value 0.7 in our experiments). Then, it goes through the list of entries in B's message. For each non-request entry about a target C, it calculates the estimate of the distance from A to C by adding together the reported estimated distance from B to C and the measured distance from A to B (from the neighbor table). Notice that this way we do not obtain the straight line distance from A to C, but rather the distance over the path traveled by the routing information. While the straight line path might contain obstacles, the length of the path traveled by the routing information is an upper bound for the shortest obstacle free path. Once the new distance estimate is calculated, robot A compares the new information with that in its routing table. It replaces the information for target C in the routing table with the new information if the new information either has a higher sequence number (has traveled from the destination more recently), or has the same sequence number but a shorter distance. The rationale behind preferring more recent information, is to direct robot navigation to areas that are better connected and therefore get navigation information more frequently. In empirical tests, we found that using sequence numbers always allowed a faster completion of the navigation task.

Updates for odometry

As a robot is moving, it continuously updates all distance estimates in its routing table according to odometry measures. It adds the distance it travels to the estimate in the table. The goal is to maintain a good distance estimate between the reception of different update messages, so that even long periods without new messages can be overcome. This way, routing information can travel partly via telecommunication in update messages and partly while being carried on board a moving robot, and the system is able to work in intermittently connected networks. Notice that we do not take the direction of the robot's movements into account, only the distance (as integrated over the robot's travel path). This may lead to overestimation of the distance (e.g., if the robot moves in circles), but this way we are sure that it is an upper bound of the length of the shortest obstacle free path. Finally, the robot also updates the estimated location (distance and direction) of each neighbor according to its odometry. Here, the movement direction is accounted for.

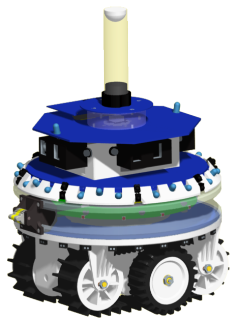
Fig. 1. Model of a foot-bot (taken from Swarmanoid (2007)).

Robot navigation

Each time the searching robot gets a valid update (with a better sequence number or with the same sequence number and shorter distance than previous information) about its target, it records the distance and direction to the robot it got the update from, and goes to that location. If it reaches that location before getting a new valid update, it waits there (different strategies could be applied here, such as continuing on the previous direction or executing a local search; instead, we opted for a conservative approach in order to focus our investigations on the network guided navigation system). If it finds the target robot in its neighbor table, it goes straight there.

4. TESTS IN SIMULATION

In this section we present results from experiments in simulation (experiments with real robots are reported in section 5). The robotic platform we use in simulation is the foot-bot (see Figure 1), which is developed as part of the Swarmanoid project[1]. It is about 15 cm wide and long and 20 cm high, and moves on the ground using Treels, which are a combination of tracks and wheels. The foot-bot's IrRB system has a maximum range of about 3 meters and a precision of 20% for range estimates and 30 degrees for bearing estimates. Its nominal bandwidth is 40kbps. These performance data are the ones we used in simulation and are based on an initial design of the system; the final system is expected to have better performance. For complete details of the foot-bot, see Swarmanoid (2007).

All tests are done using the Swarmanoid simulator (Swarmanoid (2008)). For each test scenario we execute 30 independent runs. We report the average with 95% confidence interval (using a t-test) of the time needed for the searching robot to reach its target. We compare our delay tolerant robot navigation algorithm with two other navigation systems based on ad hoc routing algorithms proposed in Ducatelle et al. (2008). The first of these sets up a route between searcher and target in the ad hoc network and lets

[1] `http://www.swarmanoid.org`

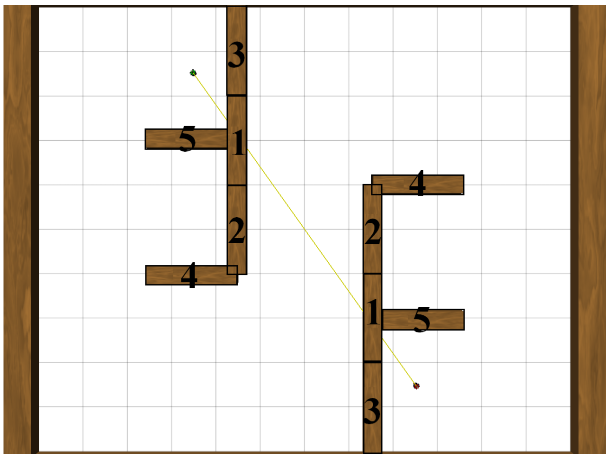

Fig. 2. Scenarios of increasing difficulty: scenario 5 includes all obstacles labeled 1 up to 5, scenario 4 includes the obstacles labeled 1 up to 4, and so on until scenario 0. The robot in the right bottom corner is the searcher, and the one in the top left corner the target. The line between them symbolizes this relationship.

the searching robot physically follow the route, stopping each time it falls without a valid route. In the following, we refer to this algorithm as "Follow route". The second system also creates a route between searcher and target, but uses it only to obtain an estimate of the straight line direction and distance to the target, and lets the searcher travel straight there. We refer to this strategy as "Follow estimate". While "Follow route" requires continuous end-to-end connectivity, "Follow estimate" can overcome moments of missing connectivity but has more difficulties when non-convex obstacles block the straight line path.

We use a room of 10 $\times$ 12 m^2, with one searching robot and one target robot, located in opposite corners of the room. All other robots execute random movements, which simulates the fact that they are involved in tasks of their own and that their behavior is independent from the task of guiding the searching robot. The random movements consist in choosing a new speed and turning radius from a gaussian distribution at regular intervals of 5 seconds (the average speed is 0.15 m/s and the average turning radius 2 m; we used a maximum speed to 0.34 m/s). All robots are equipped with a minimal obstacle avoidance mechanism based on proximity sensors.

In a first set of tests, we evaluate the different navigation systems in scenarios of increasing difficulty. We start from an obstacle free room (scenario 0) and gradually add blocks until we get a complex environment with multiple dead ends (scenario 5). The scenarios are illustrated in Figure 2, where the obstacles are labeled 1 up to 5 so that each scenario i $(0 \leq i \leq 5)$ contains all obstacles j for which $j \leq i$. We use 40 robots in total. The results are shown in Figure 3. The delay tolerant system consistently outperforms the "Follow route" system. Both approaches are based on identifying obstacle free paths through line-of-sight communication, but the "Follow route" system requires end-to-end network connectivity. The "Follow estimate" approach works slightly differently: it requires only occasional end-to-end connectivity, in order to get an estimate of the direction to go in. It can therefore

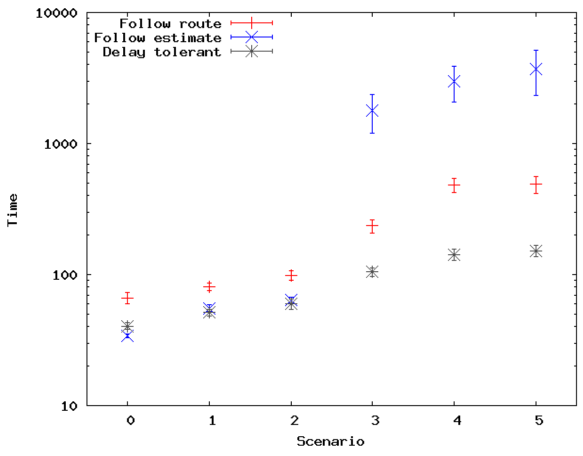

Fig. 3. Results for tests in scenarios with increasing complexity. We compare the delay tolerant navigation algorithm ("Delay tolerant") with one based on following a route in an ad hoc network ("Follow route") and one using an estimate of the goal location ("Follow estimate"). We show the time needed for the searching robot to find the target (in seconds, on a log scale), with a 95% confidence interval.

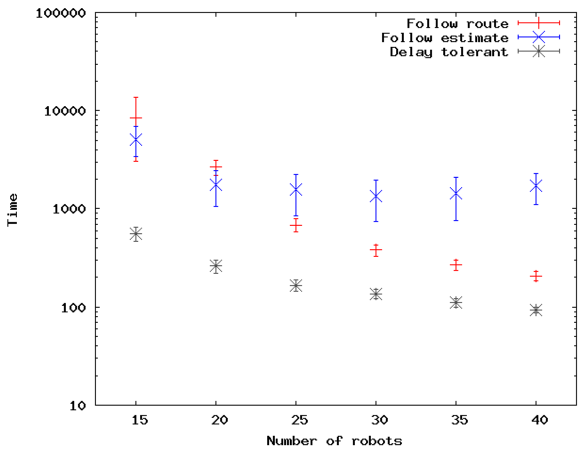

Fig. 4. Results for tests with a varying number of robots (using scenario 3 from Figure 2).

outperform the delay tolerant approach in the open space scenario, but has difficulties when the environment gets more complex, both due to the reduced network connectivity and the fact that it looses time in dead ends where it has to rely on wall following to escape.

In a second set of tests, we evaluate the behavior when varying the number of robots. We use scenario 3 from Figure 2 and increase the number of robots from 15 up to 40 (with steps of 5). The results are given in Figure 4. They show that the delay tolerant approach always performs much better than the other two. The "Follow route" approach gives bad results for low numbers of robots, but improves as the robot density increases and the network connectivity gets better. It can however never get close to the performance of the delay tolerant algorithm. The "Follow estimate" system does not profit from the increased robot density in the same way because it also struggles with the dead ends in the environment.

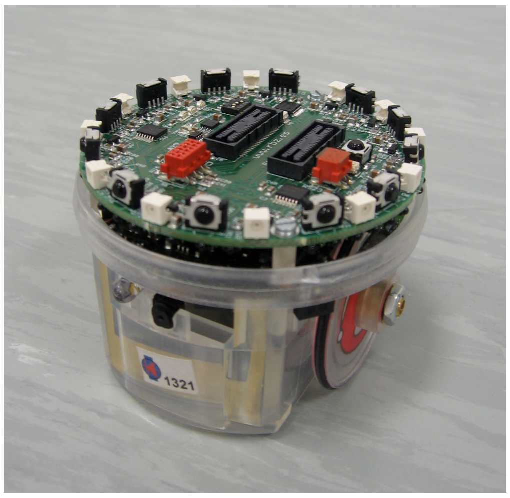

Fig. 5. An e-puck robot with IrRB module.

5. TESTS IN HARDWARE

To further evaluate our navigation system, we tested it on real robots. We used a set of 16 e-puck robots (since at the time of writing, the foot-bots still had to be completed). The e-puck (see Mondada et al. (2009)) is a small, low-cost mobile robot developed primarily for educational purposes. It has a diameter of 0.07 m and a maximum speed of 0.13 m/s. We equipped the e-pucks with the IrRB module described in Gutiérrez et al. (2008). The module contains 12 sets of infrared emitters and receivers, placed around its perimeter. It is reported to give an average error of 0.0768 m on the range estimate and 6.69 degrees on the bearing estimate (averaged over distances going from 10 cm to 6 m), but due to differences between individual boards we were faced with larger errors. Its maximum range can be tuned between 0.40 m and 6 m. It can send up to around 100 messages of 2 bytes per second, although the unavailability of a MAC protocol severely limits the practical bandwidth in situations with multiple robots; we limit ourselves to 2 messages of 2 bytes per robot every 2 seconds. Due to the limited bandwidth, it was not possible to implement the "Follow route" and "Follow estimate" algorithms on the e-pucks. A photograph of an e-puck with the IrRB module on top is shown in Figure 5.

We use an arena of 2.4×3 m^2, which means that distances are about a factor 4 smaller than in simulation. This roughly corresponds to the difference in speed between the e-pucks and the simulated foot-bots (we use a maximum speed of 0.087 m/s on the e-pucks vs. 0.34 m/s in simulation). We also tune the maximum range of the e-puck IrRB communication to be 0.75 m, which is one quarter of the range of the foot-bot's IrRB system in simulation.

Like in simulation, we carry out tests with scenarios of increasing complexity. We use four different scenarios. They are represented in Figure 6, where obstacles are labeled 1 to 3 so that each scenario i ($0 \leq i \leq 3$) contains all obstacles j for which $j \leq i$. In terms of the structure of the environment, scenarios 0 to 3 correspond to scenarios 0, 2, 3 and 4 of the simulation experiments of section 4 respectively. The searching robot is placed in the bottom right corner of the arena, and the target in the top left corner (indicated with circles). The 14 other e-pucks are

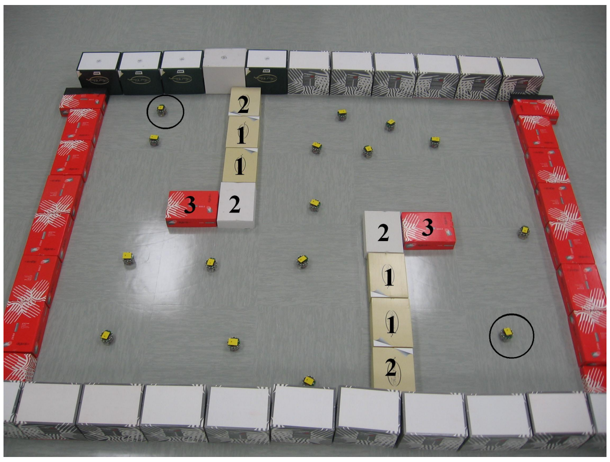

Fig. 6. Scenarios of increasing difficulty for the real robots: scenario 3 includes all boxes labeled 1 up to 3, scenario 2 includes the boxes labeled 1 and 2, and so on. The robot indicated with a circle in the bottom right is the searcher, and the one in the top left is the target.

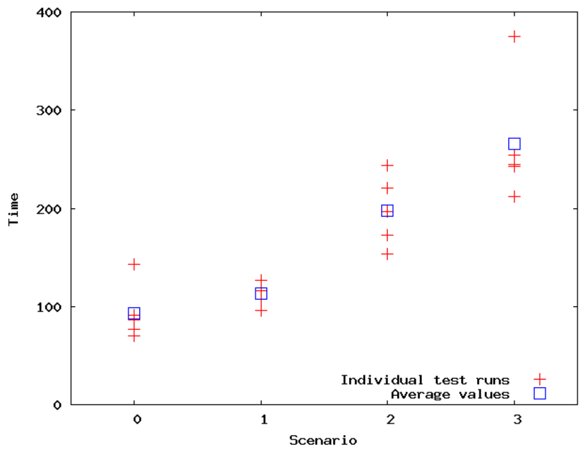

Fig. 7. Results for tests with e-puck robots in scenarios of increasing complexity. We show the time it took the searcher to find the target (in seconds) for 5 different runs per scenario, together with the averages.

placed randomly and execute random movements. These random movements consist in an in place turn of between -90 and 90 degrees, and a move straight forward at 0.087 m/s for 5 s. All robots are equipped with an obstacle avoidance mechanism. We execute 5 test runs per scenario.

In Figure 7 we report the time required for the searcher to reach its target in each scenario. These results are of a similar order of magnitude as the ones in simulation (see Figures 3 and 4). Moreover, we observe a similar trend as in Figure 3: while there is a jump in performance between scenarios 1 and 2 (scenarios 2 and 3 in simulation), the algorithm is able to cope with the increased complexity relatively well. It is also interesting to compare the travel times with the shortest possible. This can easily be done for the open space scenario (scenario 0): the straight line distance between the start position of searcher and target is about 3 m, and the speed of the searcher is 0.087 m/s, so that the minimum travel time is about 35 s. In the best run the searcher needed the double of this, while on average it needed 2.5 times this time. The lost time includes time

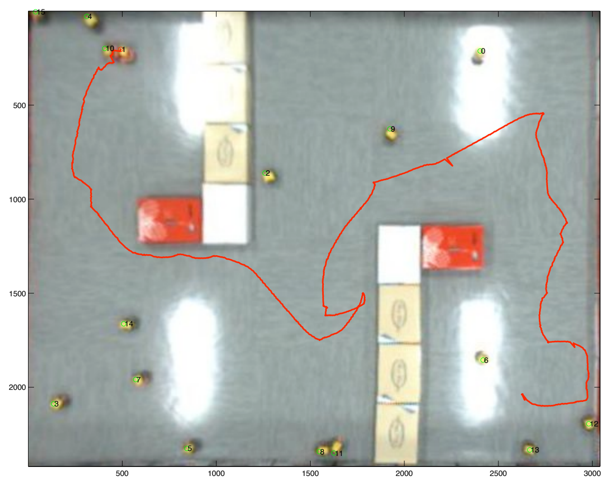
Fig. 8. Example of a navigation path in scenario 3 (the tics on x and y-axis indicate distances of 0.5 m).

for the message to travel from searcher to target and back, time spent waiting when no new updates are received, and time lost due to navigation errors. The latter are due to errors estimating direction and distance using the IrRB system, as well as to the fact that the network of robots samples the arena space in discrete points. An example navigation path (in scenario 3) is given in Figure 8.

6. CONCLUSIONS

We have presented a robot navigation system for networked multi-robot systems. It relies on the use of a local communication device that is able to provide range and bearing estimates for communicating robots. The navigation system is based on routing algorithms for ad hoc networks, but is adapted to be delay tolerant, in order to deal with situations of intermittent network connectivity. We implemented it both in simulation and on real robots. We show that it works even in cases of low robot density and network connectivity, and in complex environments.

In future work, we plan to first improve the working of the IrRB system, in order to get better distance and direction estimates, and higher bandwidth. Then, we will extend our algorithm, e.g. implement a strategy for the searcher to deal with periods in which no new updates are received.

ACKNOWLEDGEMENTS

The authors would like to thank Álvaro Gutiérrez and Alexandre Campo for their help in obtaining, programming and using the e-puck IrRB boards.

REFERENCES

Batalin, M. and Sukhatme, G. (2003). Coverage, exploration and delpoyment by a mobile robot and communication network. In *Proc. of the International Workshop on Information Processing in Sensor Networks*.

Bellman, R. (1958). On a routing problem. *Quarterly of Applied Mathematics*, 16(1), 87–90.

Corke, P., Peterson, R., and Rus, D. (2003). Networked robots: Flying robot navigation using a sensor net. In *Proc. of the 11th Int. Symp. of Robotics Research*.

Ducatelle, F., Di Caro, G., and Gambardella, L.M. (2008). Robot navigation in a networked swarm. In *Proc. of the Int. Conf. on Intelligent Robotics and Applications (ICIRA)*, volume 5314 of *LNAI*. Springer.

Fall, K. (2003). A delay-tolerant network architecture for challenged internets. In *Proc. of ACM SIGCOMM*.

Gutiérrez, A., Campo, A., Dorigo, M., Amor, D., Magdalena, L., and Monasterio-Huelin, F. (2008). An open localization and local communication embodied sensor. *Sensors*, 8.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proc. of the 9th Conf. on Autonomous Robot Systems and Competitions*.

Nouyan, S. and Dorigo, M. (2006). Chain based path formation in swarms of robots. In *Proc. of the 5th Int. Workshop on Ant Algorithms and Swarm Intelligence*.

O'Hara, K. (2003). Navigation networks: Biological inspiration for large-scale multi-robot navigation. In *Proceedings of the second International Workshop on the Mathematics and Algorithms of Social Insects*.

O'Hara, K. and Balch, T. (2004). Pervasive sensor-less networks for cooperative multi-robot tasks. In *Proceedings of the Seventh International Symposium on Distributed Autonomous Robot Systems (DARS-04)*.

Park, S., Jung, J., and Kim, S.L. (2008). Multi-robot path-finding in a random maze with multihop communications. In *Proc. of the 1st Workshop on Wireless Multihop Communications in Networked Robotics (WMCNR)*.

Payton, D., Daily, M., Estowski, R., Howard, M., and Lee, C. (2001). Pheromone robotics. *Autonomous Robots*, 11(3).

Perkins, C. and Bhagwat, P. (1994). Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proc. of ACM SIGCOMM*.

Perkins, C.E. and Royer, E.M. (1999). Ad-hoc on-demand distance vector routing. In *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*.

Pugh, J. and Martinoli, A. (2006). Relative localization and communication module for small-scale multi-robot systems. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 188–193.

Sgorbissa, A. and Arkin, R.C. (2003). Local navigation strategies for a team of robots. *Robotica*, 21.

Sit, T., Liu, Z., Ang Jr., M., and Seah, W. (2007). Multi-robot mobility enhanced hop-count based localization in ad hoc networks. *Robotics and Autonomous Systems*, 55(3), 244–252.

Swarmanoid (2007). Hardware design. Internal Deliverable D4 of IST-FET Project *Swarmanoid* funded by the European Commission under the FP6 Framework.

Swarmanoid (2008). Simulator prototype. Internal Deliverable D7 of IST-FET Project *Swarmanoid* funded by the European Commission under the FP6 Framework.

Witkowski, U., El-Habbal, M., Herbrechtsmeier, S., Tanoto, A., Penders, J., Alboul, L., and Gazi, V. (2008). Ad-hoc network communication infrastructure for multi-robot systems in disaster scenarios. In *Proceedings of the International Workshop on Robotics for Risky Interventions and Surveillance of the Environment*.