Assembly of Configurations in a Networked Robotic System: A Case Study on a Reconfigurable Interactive Table Lamp

Christopher M. Cianci, Julien Nembrini, Amanda Prorok, and Alcherio Martinoli

***Abstract*—In the present study, we are interested in verifying how the progressive addition of constraints on communication and localization impact the performance of a swarm of small robots in shape formation tasks. Identified to be of importance in a swarm-user interaction context, the time required to construct a given spatial configuration is considered as a performance metric. The experimental work reported in this paper starts from global and synchronized localization information, shown to be successful both on a real hardware system and in simulation. In a second step, communication is constrained to a local scale, thus obliging a single designated robot to disseminate the global localization information to the other agents. The reliability of the radio communication channel and its impact upon the performance of the system are considered.**

I. Introduction

With much current research going into wireless networking, including highlighting aspects of power consumption [13] and agent mobility [7], it seems only logical to consider the possibility of leveraging these new radio technologies and protocols in robotic research. A "Networked Robotic System" is a group of artificial mobile autonomous agents that utilize wireless communication among themselves or with their environment, allowing them to better fulfill their objectives.

One class of applications for such a networked robotic system is assistive and interactive environments [1]. Here, we will present the implementation of a reconfigurable interactive table lamp using a group of ten e-puck[1] robots along with a realistic simulation of the same.

Each of the robots in the system is equipped with a light, and the group is given the task of assuming various configurations as a function of user activity or instructions (see example in Figure 1; a full description of the physical setup will be given in Section II). There are several ways in which one could imagine approaching this type of coordination: through local rules for self-assembly [11], potential fields [17], or environmental templates [5]. In order to be effective, all of these methods will require some degree of communication between the agents and with the user. The study of human interfaces for multi-robot systems [19] still remains an open research question; for simplicity and efficiency, our implementation supplies the robots with user-defined configuration templates, which they must then build in a cooperative fashion.

Christopher Cianci, Amanda Prorok, and Alcherio Martinoli are members of the Distributed Intelligent Systems and Algorithms Laboratory at the École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland. {chris.cianci, julien.nembrini, amanda.prorok, alcherio.martinoli}@epfl.ch

[1]http://www.e-puck.org/

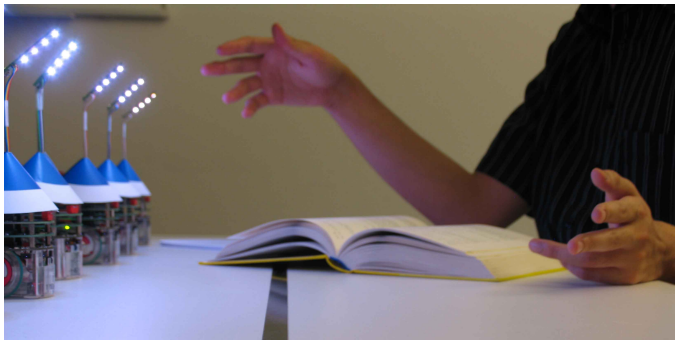

Fig. 1. The multi-robot system forms a reconfigurable interactive table lamp, assuming various configurations based on user activity or input.

The robots form an ad-hoc network, and our focus here is on the impact of different network topologies for testing the convergence to spatial configurations using the same algorithm under different conditions. In particular, the necessary communication with or among the agents is considered, and possibilities for further decentralization are explored. In the interest of systematic testing, it is useful to have a simulated model as well, which allows us to explore the parameter space more efficiently. The entry point shown here is a realistic module-based simulation.

To give a thorough illustration of the concepts introduced above, we will first outline the reconfigurable interactive lamp case study and experimental setup in Sections II and III, followed by the definition and calibration of the simulated model in Section IV, and a description of the implemented control algorithms in Section V. The results of the corresponding experiments, real and simulated, are subsequently presented and discussed in Sections VI and VII.

II. Case Study: A Reconfigurable Interactive Table Lamp

We take our inspiration from a case study we recently developed in collaboration with researchers in interactive design from the École Cantonale d'Art de Lausanne (ECAL). As a demonstrator for swarm robotic systems to be used in human environments, a fleet of mobile "lighting" robots was developed, and made to move about on a large table, such that the swarm of robots forms a kind of "distributed table lamp." In the presence of human users, the swarm of robots should quickly aggregate to form a lamp whose shape and function depends on the positions of the users, and their behavior.

As implemented, the set-up consists of a collection of e-puck robots (described in Section III-B below) fitted with lamp turrets, a table with marked boundaries for them to interact on,

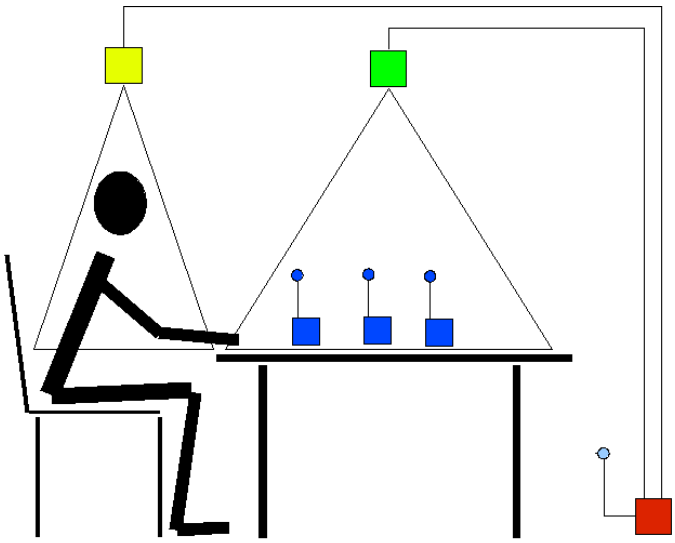

Fig. 2. User-swarm interaction setup. Robot and user positions are tracked using two different tracking systems. The information is then combined and sent via radio packets to the robots, which take action accordingly.

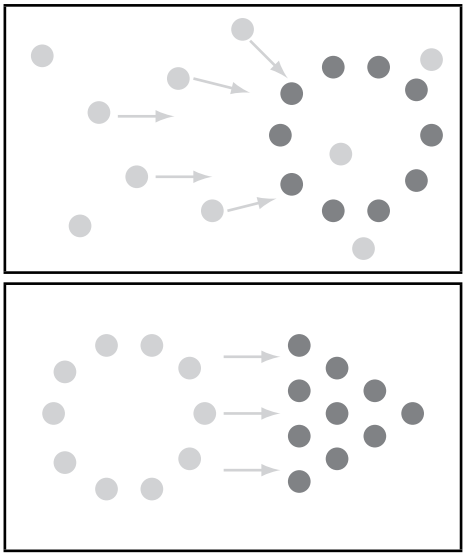

Fig. 3. The two classes of experiments performed: (left) From a random distribution to an ordered configuration, and (right) from an existing ordered configuration to another one.

an overhead camera for tracking the positions of the robots (using the SwisTrack multi-agent tracking software [6], [14]), and a human-computer interface which senses and indicates which regions of the workspace are currently occupied (see Figure 2). The target configuration of the aggregate "lamp" is controlled through crude tracking of users's positions and postures around the table. User and robot tracking are then integrated in software, and configuration and positioning information is sent to the robots.

Considering the specific task of ordered aggregation as a benchmark, this study developed a simple algorithm able to control the geometry of an aggregate consisting of multiple miniature mobile robots endowed with limited computational and communication capabilities. The robots use only infrared proximity sensors (both for avoiding teammates and detecting the boundary line on the surface of the table) and wireless communication.

In this scenario, since the objective driving the aggregation problem centers around humans (not typically known for their extreme patience), time-to-completion becomes critical. The user cannot be made to wait too long for the robots to aggregate. Therefore, the system should react quickly to changes in the environment. For this reason, the preliminary versions of the setup described here were built using a partially centralized algorithm where global positioning information is provided to the robots.

III. Experimental Setup

For the sake of repeatability and rigorousness, the test cases presented here on the multi-robot network are performed without the user tracking subsystem; the configurations sent to the robots are pre-specified. We then analyze the performance of the system based on how quickly it is able to construct a configuration requested of it.

A. Test Classes

Two distinct test classes are considered; building a configuration from an initially random arrangement of the robots, and transitioning from one established configuration to another (see Figure 3). Results of these tests, from both real hardware and simulation, will be presented in Section VI.

Test Class One: Random Initial Distribution

In the first set of experiments, we consider the problem of driving robots from an initially scattered state into a structured configuration. To accomplish this, the robots initially perform obstacle avoidance for a period of 30 seconds, yielding a sufficiently randomized distribution, and then in a second phase attempt to construct the configuration.

Test Class Two: Structured Initial Conditions

The second set of experiments considers the task of switching from one existing completed configuration to another, either by trying to reach the same configuration in a different location, or a different configuration. The lateral offset between the initial and final configurations is 60 centimeters.

The time to reach the configuration and individual robot trajectories are recorded, as well as message loss. Two specific target configurations are considered: circles and packed triangles, as shown in Figure 3. This task relates to other theoretical work on assembly of robotic configurations, as in [12], but few such approaches have yet been attempted on real platforms.

In the case of reforming the same configuration in a different location (translation), it is certainly true that one could attempt to complete the task using a movement in tight formation, similar to what is done in [10]. However, in the context considered, such a direct translation task is relatively unlikely to occur in an environment with real users. Our solution will therefore neglect movement in formation in favor of the simpler procedure of re-building the new configuration based

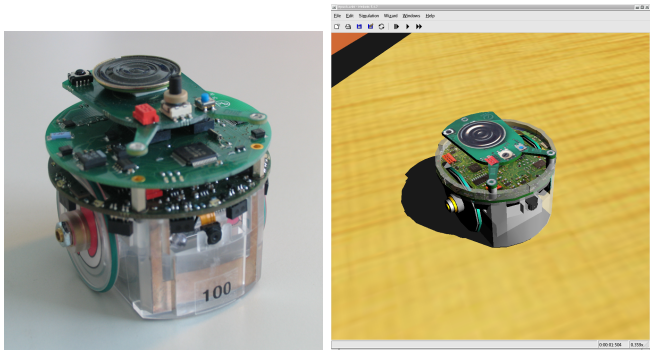

Fig. 4. (left) The e-puck robotic platform. Shown here with the radio communication board stacked between the basic module and the jumper board. (right) A simulated model of the e-puck in Webots.

on the same rules as the first one. Nevertheless, the fact that the starting positions are ordered may have an effect on the efficiency of convergence to the subsequent configuration.

B. Hardware Implementation: The e-puck Miniature Robot

The e-puck robot (shown in Figure 4) is a small-scale robotic platform, designed in collaboration with the Laboratory of Robotic Systems[2], the Distributed Intelligent Systems and Algorithms Laboratory[3], and the Laboratory of Intelligent Systems[4] at the École Polytechnique Fédérale de Lausanne (EPFL). It is a differential drive robot (with each wheel driven independently by a stepper motor), and is equipped with eight infrared proximity sensors (approximately 5cm range), a 3-axis accelerometer, three microphones, and a color camera (640 x 480 pixels). Stackable modules can be attached, enabling a maximum of flexibility for extensions.

In this experiment, ten e-puck robots are used. In addition to the above mentioned standard sensors and actuators, the e-pucks are each equipped with a floor sensor and, in order to create a networked robotic system, a local radio communication turret [4]. The radio board was constructed to be a low-power system operating on standardized protocols. The physical radio is a Chipcon CC2420, an IEEE 802.15.4 and ZigBee compliant transceiver, enabling hybrid communication between this module and other platforms running TinyOS [8].

In the setup, the robots are placed on a table, where they are free to move about in a 150 $\times$ 90 centimeter area, marked with black lines on the surface of the table; the floor sensors allow the robots to detect and avoid the lines, thus staying within their designated section on the table. An overhead camera provides low-resolution tracking information (deliberately down-sampled to 1cm granularity in order to limit message size), via a desktop computer in the corner of the room which is capable of relaying positioning information to the robots in radio packets from a MICAz basestation.

IV. Module-based Simulation

Performing systematic experiments directly on the target hardware system can be cumbersome, costly, time-consuming, or even impossible for logistical reasons such as safety or availability. However, by demonstrating correspondence with a higher abstraction layer representation of the system, we can gather and analyze additional information which may eventually be applied back to the control of the target system. In this context, simulation can therefore be a very useful tool for bridging the gap between theory and experiment. It is not intended to be a substitute for real experiments, but rather a supplement, allowing additional flexibility and diversity in the tests performed.

Here we highlight the relationship between the physical implementation of the target system and a realistic simulation, which can be seen as the first abstraction layer in a hierarchical spectrum of models, as shown in [15], [3], though this is not the primary focus of the present paper. While obviously a simplified version of the real world, this class of simulator still maintains as much realism as possible by preserving intra-node details, such as the individual sensors, actuators, transceivers, etc.

A. Modules

1) The e-puck in Webots: Webots[5] [16] is a versatile robotic simulation platform, capable of managing a wide range of simulated environments and systems, from 2-dimensional kinematic interactions to 3-dimensional worlds with full physical dynamics. As a function of the system being studied, this flexibility can be leveraged to yield the most advantageous trade-off between realism and execution speed. Figure 4 shows a fully functional model of the e-puck robot that has been developed and calibrated for use in Webots, including its two stepper motors, eight infrared proximity sensors and light sensors, camera, speaker, three microphones, and radio communication module.

2) SwisTrack in Webots: The functionality provided by SwisTrack and the overhead camera in the physical setup is also reproduced in simulation, using measured values from the real system for parameters such as position error and latency.

3) OMNeT++: Radio communication in Webots is modeled using a plug-in module built by replacing the provided interface of the open-source network simulation engine OMNeT++ [18] with a wrapper conforming to the Webots plug-in API (additional details can be found in [3]). Positions and instructions are passed from Webots to OMNeT++, which then handles the channel coding, fading signal propagation, and other wireless network dynamics in a realistic manner.

OMNeT++ was selected for its modularity, relative simplicity, and active user base. Indications are that it is becoming a widely accepted standard among researchers. An 802.15.4 module was also written for the OMNeT++ Mobility Framework, along with a partial implementation of the ZigBee standard, corresponding roughly to the subset which is used on the MICAz [9] in TinyOS [8].

B. Calibration and Correspondence with Reality

As mentioned above, results obtained in simulation are all but meaningless unless adequate correspondence can be

[2]http://lsro.epfl.ch/
[3]http://disal.epfl.ch/
[4]http://lis.epfl.ch/
[5]http://www.cyberbotics.com/

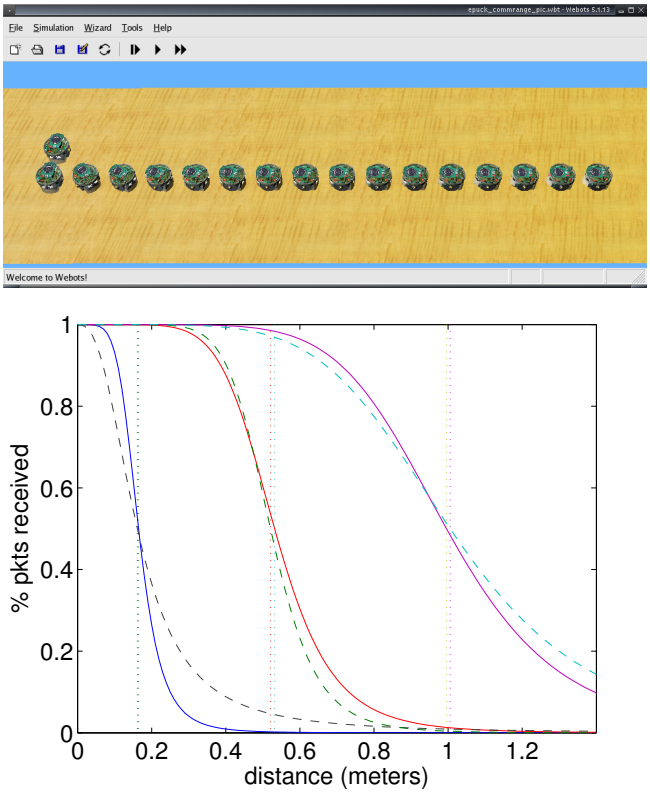

Fig. 5. (top) The experimental environment used for the range test experiments in [4], re-created in Webots, using the OMNeT++ plug-in module. A single transmitter sends a known number of packets, which are counted (if received) at each of the 16 receivers stationed at known distances from the transmitter. (bottom) Comparison of data taken from the physical system with analogous data produced by the realistic simulation, for three typical transmission power settings. Curves are regressions of 256 measurement points each (not shown, for clarity). Dashed curves represent data taken from the real system, and solid curves the realistic simulation. Vertical dotted lines mark 50% packet loss.

established between the target system and the simulated model. To this end, we have reproduced the setup used to determine the transmission range of the e-puck communication module in [4] in Webots. Sixteen robots are arranged in a line near a seventeenth which transmits a large known number of packets, which is compared with the number received at each of the sixteen receiving nodes, to give an estimation of the probability of message reception as a function of distance.

Figure 5 shows the results of the simulated experiments compared with corresponding data taken from the physical system; for each of three selected transmission power settings, 16 runs of 250 messages (2 messages per second) were performed both on the real platform and in simulation. Each curve shown is a regression over 256 points (16 runs $\times$ 16 robots), and vertical dotted lines are drawn at 50% packet loss (a threshold commonly used as an approximate "communication radius" for reference) which are within a centimeter in all cases.

C. Simulated Model of the Case Study

The system described in Section III is also faithfully reproduced as a Webots simulation (see Figures 6 and 7). This

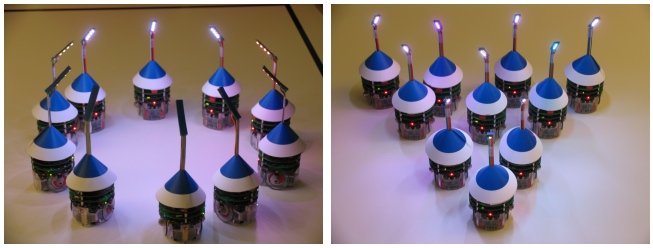

Fig. 6. The physical system: Ten e-pucks with floor sensors, radios, and lamp attachments. The 'arena' (section of the table marked off with black lines) is 150 $\times$ 90 centimeters.

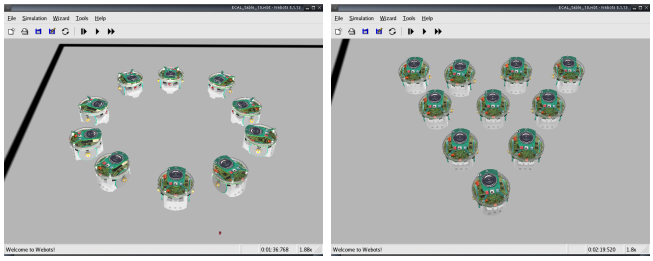

Fig. 7. The simulated system: An identical setup is reproduced in Webots, including line detection/avoidance and network dynamics (using the OMNeT++ plug-in).

allows us to experiment with the effectiveness of configuration forming algorithms while performing many more iterations, varying the amount of available and leveraged information at the individual robots, and testing different strategies for communication and information sharing among the robots. Because of the two-dimensionality inherent to the tabletop (as well as OMNeT++), we use a 2D kinematic simulation in the Webots environment to ease computational requirements and increase speed.

With these tools we can first accurately and precisely reproduce the outcome of parallel experiments in simulation and with real hardware to validate the use of simulation in this context. This can then justify later use of the simulation to explore other aspects of the parameter space with significantly greater ease and speed than performing all of the corresponding experiments on real hardware.

V. Control Algorithm

Here we present a possible method for coordination of the robotic swarm: a simple partially distributed algorithm for layered construction of configurations using nearest-neighbor allocation.

We have chosen to centrally specify the configurations and explicitly send them to the robots, instead of relying on a distributed strategy where configurations emerge from individual behaviors and multiple interactions. Interesting research addressing the specific problem of automatic generation of local behaviors from high-level directives can be found in work such as [2] or [20], but typically neglecting the impacts of embodiment, sensor and actuator noise, and speed limitations. While realistic implementation of such techniques (or similar) is planned in further experimentation, the current incarnation of the physical system combined with the requirement for

ALGORITHM I
LAYERED NEAREST-NEIGHBOR CONTROL (BROADCAST)

```
Receive target configuration from tracker.
repeat
    Receive position information (x, y) from tracker.
    Select nearest unfilled configuration location (x̂, ŷ, θ̂) as target.
    Attempt to move straight towards selected target.
    if Obstacle detected before target reached then
        Perform random turn / random backoff.
    else
        Estimate current orientation (θ) from difference between
         current and previous positions.
    end if
until Target location reached.
```

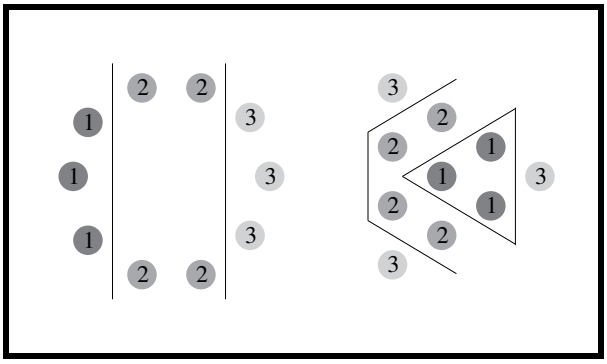

Fig. 8. Examples of configurations constructed from layers, so as not to unintentionally isolate robots from the positions they need to fill. (left) An open circle, and (right) a packed triangle. Positions are numbered by the layer that they belong to (e.g. all positions marked "1" must be filled before the positions marked "2" become available).

precise and absolute positioning under time pressure has led us to begin with a simpler, more straightforward approach.

The control algorithm should ensure that the robots proceed steadily towards their assigned positions and do not interfere with each other, and can be decomposed into three subtasks: position allocation, collective motion, and low-level control.

A. Layered Nearest-Neighbor Control

The *position allocation* subtask is an assignment problem. We have implemented a nearest-neighbor allocation algorithm: each robot attempts to move towards the closest unfilled position in the target configuration. This strategy may result in allocating more than one robot to the same destination within a given target configuration, however, once the position is filled the others will be automatically reassigned, and obstacle avoidance will prevent collision in the event that two robots arrive simultaneously. In order to prevent potential deadlock resulting from unfilled interior positions which may no longer be accessible due to other already well-placed robots, a *collective motion* directive is established by separating the list of target positions in the configuration into *layers*, such that exterior positions may not be filled (are not considered available) until the interior ones are already complete.

Two example configurations divided into layers are depicted in Figure 8. Layers for a specific configuration have been currently defined by hand, exploiting symmetry in the shape to be constructed; future algorithmic enhancements may involve automatically partitioning the configuration into layers and a possible distributed implementation on the robotic platform.

For *low-level control* we have implemented a simple control layer combining obstacle avoidance with movement towards the target position location, with both behaviors computed locally on the robots. Motor commands are determined simply by attempting to drive directly to the currently allocated target position; if an obstacle is encountered along the way, the robot will execute a random turn (in place) and a random backoff before re-attempting to drive straight towards its target. This turn/straight approach serves to minimize the accumulation of error in odometry, which is used to interpolate between reception of successive position messages (sent at approximately 1Hz, but not always received by all the robots). It is also noteworthy that there is no explicit path planning in this approach; it is merely the stochastic nature of the control primitives (random turn / random backoff) which guarantees that a robot will eventually be able to move around any potential blockade between it and its target position. A pseudocode representation of the complete controller is shown in Algorithm I.

B. Networking Variations

The implemented algorithm described above makes use of 'global' positioning in a shared coordinate frame common to all of the robots, and is therefore 'partially centralized.' In order to assess the effect of the networking setup on system performance, we test two simple strategies for information dissemination. First, the positioning information (extracted from the overhead camera feed) is periodically sent to the robots via the radio directly from the basestation attached to the tracking computer. Then, in the second setup, as an attempt to move gradually towards a more decentralized solution, the positioning information is sent to only one of the robots, which is then responsible for relaying this message locally to all of the others. As the robots are constantly moving, tracking information will rapidly become outdated, so we choose not to repeat the relayed packet more than once. Therefore, when the relay robot misses a message from the tracker, none of the robots will receive an update, and when it successfully receives and re-broadcasts a message, the others (being in closer proximity to the relay robot than they are to the basestation) have a slightly higher probability of receiving the update than they did in the previous scenario.

Coordinates in the position messages are quantized to approximately one centimeter granularity, due to limitations on message length and the desire (for simplicity and robustness) to transmit a single message containing all the necessary information, rather than fragmenting it. Using the tracking information, the robots are able to re-calibrate their odometry, and more accurately maneuver themselves closer to their target position. It is also worth noting that such position messages should not be sent with an excessively high frequency, so as to allow the robots to move more than one centimeter between updates.

VI. RESULTS

The test cases described in Section III-A, with direct message communication (Algorithm I), were performed in

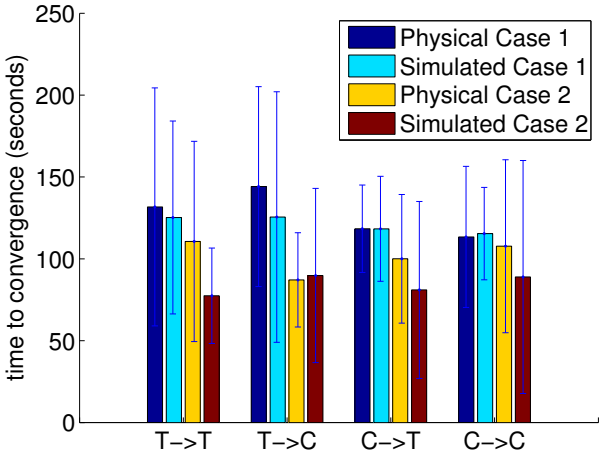

Fig. 9. Results of experiments in both the real and simulated system. Error bars represent standard deviation. The left two bars in each group of four represent Case 1, constructing a configuration from randomized initial conditions; the remaining two bars on the right side of each group represent Case 2, re-forming a configuration starting from an already achieved configuration (see Figure 3).

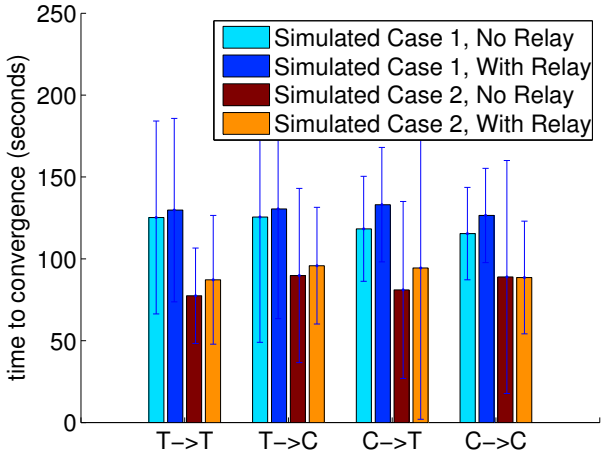

Fig. 10. Comparison of results in the simulated system between standard (direct) and relayed messaging. Error bars represent standard deviation. Though the differences are relatively small, the relayed version is consistently slightly slower. This is likely due to the fact that when the relay robot misses a message from the tracker, none of the robots will receive a message in that timestep, effectively lowering the update rate (and hence the accuracy) of the collective system.

sequence (random → configuration → configuration) ten times for each combination ($\triangle \rightarrow \triangle$, $\triangle \rightarrow \bigcirc$, $\bigcirc \rightarrow \triangle$, $\bigcirc \rightarrow \bigcirc$) on the physical system, and 100 times each in simulation (shown in Figure 9).

Additionally, a second full suite of simulated results was prepared using the relayed messaging scheme, where only one robot receives information from the tracker and subsequently disseminates it to the others (comparison of the two simulated experiments is shown in Figure 10).

As can be seen in Figure 9 with direct communication, test class one consistently takes slightly longer, which may be due to the time taken by the randomization phase before starting to build the configuration. Another factor appears to be that the configurations were specified on opposite sides of the table (one of which was farther away from the basestation in the corner), resulting in a smaller percentage of messages being successfully received (55% vs. 67%)[6]. Nonetheless, the results are relatively consistent, and the times achieved (approximately two minutes) are acceptable for the purported application, albeit with deviations somewhat larger than optimally desired.

In the relayed communication scenario (Figure 10), as expected, the completion times are slightly longer than those with direct communication. This may be explained by the fact that a single failed packet transmission (from the tracker to the relay robot) results in missed messages for all ten robots in that timestep. With the reception rate thus lowered, additional error in odometry (and therefore more frequent recalibration maneuvers) results in a negative impact on system performance. Curiously, while one might expect the standard deviations to be systematically larger in the relayed scenario, this does not appear to be the case. The reason for this is not immediately clear, though it may be related to the balance between the reduced probability of receiving a relayed message when the relay robot misses the tracker message and an increased probability of receiving a message when a message is sent from the relay robot on the table instead of the basestation in the corner.

While the results presented here are encouraging, they unfortunately come at a cost, in the form of "partial centralization" (the reliance on 'global' positioning in a shared coordinate frame). Nonetheless, the tools developed for this study, particularly the module-based simulation, will enable us to further explore possibilities for increased distribution and autonomy in control of the individual agents—the most promising candidates of which can then be evaluated on the target system as well. In this way, additional means to increase the efficiency and the robustness of the reconfiguration strategy will be investigated.

VII. Remarks, Conclusions, and Future Work

Even in what may appear to be a relatively simple setup, such as the one described here, a surprising number of possibilities, parameters, and even concerns can arise. As we have seen in the experiments presented, the inclusion of some global information (positioning provided by the tracking system) does not necessarily guarantee trivially optimal performance. Among other factors, the staggering amount of packet loss in the real system should be cause for reflection, as it shows to what extent some theoretical approaches may be untenable, and that success under such conditions would both require and demonstrate a significant degree of robustness.

[6]Clearly, efforts could be made to improve this, including revisiting the placement of the analog circuitry and the choice of antenna in the radio board described in [4], or implementing a richer subset of the ZigBee extensions. Though it should be noted that even in the absence of such optimizations, the system as presented is already robust to heavy packet loss, which does not prevent it from completing its task.

In a system which should be not only robust, but also scalable, it seems natural to look towards distributed control, as centralization will not only continue to be inefficient, but even may become ineffective under worsening conditions or increasingly many agents. Some possible avenues for distributing this particular system will likely include leveraging direct node-to-node local communication and localized relative positioning (using on-board sensors) between the agents. However, these methods come with their own complications which must be addressed, such as the accumulation of uncertainty at the collective level due to partial perception at any one node in the system. Other challenges could be to distribute the partitioning of a configuration into layers, or even the definition of the configuration locations themselves; one can imagine asking the system simply to produce the best triangle that it can, using however many robots happen to be available (such that failure of individual agents would have minimal impact).

Many of these issues may be addressed with a networked robotic system, in new and intriguing ways which promise potential solutions for a diverse set of problems and applications.

VIII. Acknowledgments

Christopher Cianci and Amanda Prorok currently receive support from the Swiss National Science Foundation (Contract Nr. PP002-116913). This work was also partially supported by a joint initiative between the École Polytechnique Fédérale de Lausanne (EPFL) and the École Cantonale d'Art de Lausanne (ECAL). The authors would like to acknowledge Nicolas Henchoz, Patrick Keller, Christophe Guignard, Christian Babski, Clément Hongler and Xavier Raemy for their support and contribution to this work.

References

[1] G. D. Abowd and E. D. Mynatt, *Designing for the human experience in smart environments*. Wiley, 2005, pp. 153–174.

[2] S. Berman, A. Halasz, V. Kumar, and S. Pratt, "Algorithms for the analysis and synthesis of a bio-inspired swarm robotic system," in *Simulation of Adaptive Behavior (SAB-2006), Swarm Robotics Workshop*, vol. 4433, Rome, Italy, October 2006, pp. 56–70.

[3] C. M. Cianci, J. Pugh, and A. Martinoli, "Exploration of an incremental suite of microscopic models for acoustic event monitoring using a robotic sensor network," in *IEEE International Conference on Robotics and Automation (ICRA)*. Pasadena, CA, USA: IEEE Press, May 2008, pp. 3290–3295.

[4] C. M. Cianci, X. Raemy, J. Pugh, and A. Martinoli, "Communication in a swarm of miniature robots: The e-puck as an educational tool for swarm robotics," in *Simulation of Adaptive Behavior (SAB-2006), Swarm Robotics Workshop*, Rome, Italy, October 2006, Lecture Notes in Computer Science (2007), vol. 4433, pp. 103–115.

[5] N. Correll and A. Martinoli, "Collective inspection of regular structures using a swarm of miniature robots," in *International Symposium on Experimental Robotics (ISER)*, M. H. A. Jr. and O. Khatib, Eds., Singapore, June 2004, pp. 375–385.

[6] N. Correll, G. Sempo, Y. L. de Meneses, J. Halloy, J.-L. Deneubourg, and A. Martinoli, "SwisTrack: A tracking tool for multi-unit robotic and biological systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Beijing, China, October 2006, pp. 2185–2191.

[7] K. Dantu, M. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme, "Robomote: enabling mobility in sensor networks," in *International Symposium on Information Processing in Sensor Networks (IPSN)*. Los Angeles, CA, USA: IEEE Press, April 2005, pp. 404–409.

[8] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for network sensors," in *Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, Cambridge, November 2000.

[9] J. L. Hill and D. E. Culler, "Mica: A wireless platform for deeply embedded networks," *IEEE Micro*, vol. 22, no. 6, pp. 12–24, November/December 2002.

[10] G. A. Kaminka and R. Glick, "Towards robust multi-robot formations," in *International Conference on Robotics and Automation (ICRA)*. Orlando, FL, USA: IEEE, May 2006, pp. 582–588.

[11] E. Klavins, "Self-assembly from the point of view of its pieces," in *American Control Conference*, Minneapolis, MN, USA, June 2006.

[12] E. Klavins, S. Burden, and N. Napp, "Optimal rules for programmed stochastic self-assembly," in *Robotics: Science and Systems*, G. S. Sukhatme, S. Schaal, W. Burgard, and D. Fox, Eds. Philadelphia, PA, USA: MIT Press, August 2006.

[13] K. Klues, G. Xing, and C. Lu, "Link layer support for unified radio power management in wireless sensor networks," in *International Conference on Information Processing in Sensor Networks*. Cambridge, MA, USA: ACM Press, April 2007, pp. 460–469.

[14] T. Lochmatter, P. Roduit, C. M. Cianci, N. Correll, J. Jacot, and A. Martinoli, "Swistrack - a flexible open source tracking software for multi-agent systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nice, France, 2008, p. to appear.

[15] A. Martinoli, K. Easton, and W. Agassounon, "Modeling of swarm robotic systems: A case study in collaborative distributed manipulation," *The International Journal of Robotics Research*, vol. 23, no. 4-5, pp. 415–436, 2004.

[16] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotic Systems*, vol. 1, no. 1, pp. 39–42, 2004.

[17] P. Song and V. Kumar, "A potential field based approach to multi-robot manipulation," in *International Conference on Robotics and Automation (ICRA)*. Washington DC, USA: IEEE, May 2002, pp. 1217–1222.

[18] A. Varga, "Software tools for networking: "OMNeT++"," *IEEE Network Interactive (2002)*, vol. 16, no. 4, July 2002.

[19] A. Wagner, Y. Endo, P. Ulam, and R. Arkin, "Multi-robot user interface modeling," in *Distributed Autonomous Robotics Systems (DARS)*, M. Gini and R. Voyles, Eds. Minneapolis, MN, USA: Springer-Verlag, July 2006, pp. 237–248.

[20] P. Yang, R. A. Freeman, and K. M. Lynch, "Multi-agent coordination by decentralized estimation and control," *IEEE Transactions on Automatic Control (To appear)*, 2008.