Comparing Coordination Schemes for Miniature Robotic Swarms: A Case Study in Boundary Coverage of Regular Structures

Nikolaus Correll, Samuel Rutishauser, and Alcherio Martinoli

Swarm-Intelligent Systems Group, École Polytechnique Fédérale Lausanne
Station 14
CH-1015 Lausanne, Switzerland
firstname.lastname@epfl.ch

We consider boundary coverage of a regular structure by a swarm of miniature robots, and compare a suite of three fully distributed coordination algorithms experimentally. All algorithms rely on boundary coverage by reactive control, whereas coordination of the robots high-level behavior is fundamentally different: random, self-organized, and deliberative with reactive elements.

The self-organized coordination algorithm was designed using macroscopic probabilistic models that lead to analytical expressions for the algorithm's mean performance. We contrast this approach with a provably complete, near optimal coverage algorithm, which is due to its assumption (noise-less sensors and actuators) infeasible on a real miniature robotic platform, but is considered to yield best-possible policies for an individual robot.

Experimental results with swarms of up to 30 robots show that self-organization significantly improves coverage performance with increasing swarm size. We also observe that enforcing a provably complete policy on a miniature robot with limited hardware capabilities is highly sub-optimal as there is a trade-off between coverage throughput and time spent for localization and navigation.

1 Introduction

We consider the multi-robot boundary coverage problem [10], which is motivated by a case study aiming at autonomous inspection of a jet turbine by a swarm of miniature robots (Figure 1, left), but is also relevant for various other inspection/coverage tasks such as painting or mowing. The jet turbine environment imposes drastic constraints on the robotic platform (e.g., miniaturization, only local communication), and therefore emphasizes a distributed approach.

In the boundary coverage problem, a group of k robots is required to completely inspect all points on the boundary of objects in a specified environment. In this paper, we consider a specific case of boundary coverage concerned with regular structures.

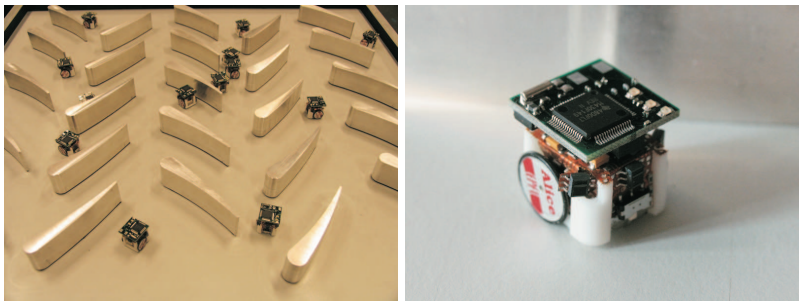

Fig. 1. *Left:* A swarm of miniature robots *Alice* [5] executing boundary coverage in a simplified 2D model of a jet turbine's engine. *Right:* The miniature robot *Alice* with its extension module.

The boundary coverage problem was formally introduced by Easton and Burdick [10], who also provided a provably complete, near optimal algorithm for coordinating a team of holonomic point robots, whereas we introduced a probabilistic algorithm that was experimentally validated using a swarm of miniature robots in [7]. Obviously, the feasibility gap between the two approaches is large: In [10] trajectories for robot coordination are calculated off-line, assuming perfect navigation/localization abilities of the robots. In [7] instead we use no planning, but coordination is fully decentralized and reactive, enabling execution by minimalist robots with crude sensors and limited localization capabilities. While [17] extends the algorithm of [10] to work in dynamic environments with distributed path re-planning, we raise the level of coordination of our minimalist approach: we implement and compare two algorithms on the *Alice* platform [5] that rely on orthogonal paradigms, deliberative planning with minimal reactive parts [2] and self-organization [3]. Whereas in the deliberative approach robots plan their trajectories based on an algorithm leading to provably complete coverage, in the self-organized approach robots follow simple heuristics that govern their behavior upon interaction with other robots or the environment.

1.1 Related Work

Random versus deliberative strategies for the coverage problem in theory and simulation have been addressed previously by for instance [11, 16]. In this paper, we address this problem experimentally by large scale robotic experiments. Due to the limitations of real miniature robots, we do not expect

complete coverage. However, we would like to study whether it is indeed a good policy to always choose the next robot's action assuming sensing and actuation were perfect, which is considered best practice [16].

Although boundary coverage is distinct from distributed coverage path planning [1, 4], which considers coverage of every accessible point in the environment by a robot team, boundary coverage of a *regular* structure, and thus visiting every one of its elements, is comparable with visiting every cell of a grid as for instance in [1, 4].

1.2 Self-Organization as Coordination Mechanism

Self-organization is emerging from the interplay of four ingredients: Positive and negative feedback (e.g., amplification or saturation, respectively), randomness, and multiple interactions among individuals [3]. While self-organization might achieve less efficient coordination than other distributed control schemes, it can provide extremely high levels of robustness and can be applied to miniature robotic platforms such as those mentioned in this paper.

One of the major drawbacks of self-organization in an engineering context is its lack of analytical tractability of the resulting collective behavior. We try to overcome this limitation by combining reactive control (e.g., [2]) on the individual level with probabilistic modeling [12], that allows us to calculate the analytic mean of arbitrary swarm performance metrics based on the (probabilistic) behavior of the individual agent. Modeling can hence be used to guide the design process [6, 8] (see below), which lead to an improved communication scheme that is experimentally studied in this paper.

2 Experimental Setup

Experiments are conducted in a 60cm×65cm arena populated with 25 blades in a regular pattern (Figure 1, left), mimicking the rotor and stator blades in a turbine. Stator blades can be distinguished from rotor blades as their curvature is concave whereas the curvature of a rotor blade is convex when looking at the edge following the round tip (considering coverage of the boundary in clockwise direction), compare Figure 2. The *Alice* robot (Figure 1, [5]) has a size of 2cm×2cm×2cm, a differential wheel drive with reaching speed of up to $4\frac{cm}{s}$, and four infrared distance sensors for obstacle detection (up to 3cm), and 4Bit/s local communication up to 6cm. It is endowed with a PIC micro controller with 368bytes of RAM.

For implementing more sophisticated collective navigation algorithms and enhancing both on-board computation and communication capabilities, we use an extension module measuring 2cm×2cm. The extension module is inspired by [14] and provides 2.4Ghz wireless communication (Chipcon CC2420), 512kB Flash, and a TI MSP430 processor (4KByte RAM) running TinyOS. The extension module is connected to the Alice's serial port (Figure 1). In

this paper, the extension module is exclusively used as provider for additional computational power, and communication between the robots is solely based on the on-board infrared distance sensors.

Systematic experiments involving 5 to 30 robots are monitored using an overhead camera and the tracking software *SwisTrack*[1] [9].

3 Self-Organized Approach: Interactive Random Coverage

In [7] we implement a very simple distributed algorithm: robots are searching randomly through the arena; on encountering a blade, a robot attaches to it and circumnavigates it for a certain time (10s), and finally leaves it at its tip. By this, we exploit the structure of the environment to bias the robots' trajectories. Although this makes sense in a continuous environment such as a real cylindrical turbine, a bounded arena leads to sub-optimal performance due to non-uniform distribution of the robots [7]. In [6] we introduce the concept of robots acting as "beacons" preventing other robots from finishing the inspection of a blade, and find an optimal (dynamic) policy for employing the beacon state in [8]. Using probabilistic modeling we show theoretically that turning the beacon behavior on after a certain time can lead to a 5% improved performance, but only if there are more robots than blades. In this contribution we combine lessons learned from [6–8], and have the robots perform an additional movement along the blades contour for 50% of the blades on average (the robot's decision to leave a blade at its tip or sweep along its contour for leaving at the other end is taken randomly with a 50/50 chance). By this, the spatial distribution of the robots is uniform, and at the same time robots communicate that this blade has already been inspected while moving along its contour. Additionally, we exploit low bit-rate local communication (via the *Alice's* on-board infrared sensors) for decreasing redundant inspection by having robots abandon an inspection if they are following or encounter another inspecting robot (in this case only the robot having the blade to its right will leave). Finally, searching robots will not attach to a blade if there is an inspecting robot nearby (6cm max.). These additional steps were necessary, as it is difficult to show a 5% performance increase, as predicted in [8] for static beacons, experimentally in a significant way.

4 Complete Approach: Spanning-Tree Coverage

Exploiting the regularity of the structure, the *Alice* constructs a spanning tree with the blades as nodes, and possible routes between nodes as edges. Hereby we consider the 4-neighborhood of each blade as possible routes (Figure 2,

[1] http://swistrack.sourceforge.net

left). Edges are numbered from 0 to 3, where the direction 0 is given by the direction of the face that follows on the round tip when considering clock-wise coverage of the boundary. Nodes are indexed with 2D coordinates relative to the root, where the x-axis is given by the direction of edge 1, and the y-axis by the direction of edge 0 (compare Figure 2, left).

The spanning tree is constructed on-line and systematically explored by a Depth-First-Search (DFS) algorithm. The DFS algorithm is setup such that the direction of exploration is not biased in order to promote uniform coverage of the environment (see Section 3), even if the robots restart exploration occasionally due to failure. This is achieved by selecting the edges in clockwise or counter-clockwise order depending on the coordinate of the node.

An edge is considered as fully explored when all nodes connected to it have been visited. Once all edges of a node are explored, the DFS algorithm makes the robot physically return to its parent node (known as *backtracking*) and explores remaining unexplored edges of this node. The algorithm goes on until it reaches the spanning tree's root, a policy leading to provably complete coverage. Notice that the algorithm explores *all* possible edges, including those ending at a wall. As DFS will visit every node at least twice during backtrack-

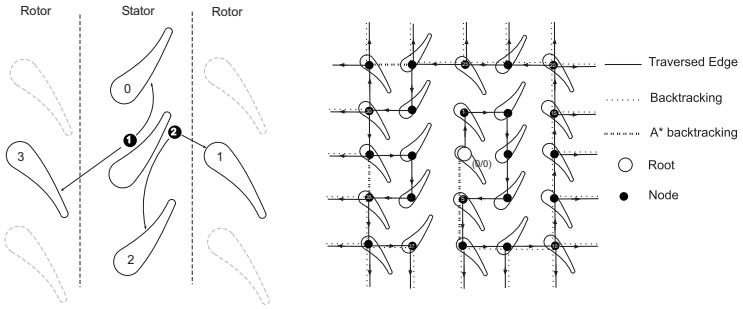

Fig. 2. *Left:* Possible routes between a blade and its four neighbors. A behavioral algorithm drives the robot to one of two launch-points (black circles), whereas the blade-to-blade transition is executed by open-loop control. *Right:* Possible trajectory for a single robot along a spanning-tree in a 5x5 blade environment (bold line). Dotted lines are paths the robot is backtracking, dash-dotted line are "short-cuts" provided by the A^* algorithm.

ing, we use the A^* algorithm for calculating the shortest path to the first node that has unexplored edges along the backtracking path (see [15] for a similar algorithm and analysis). Once the first node that has unexplored edges on the robot's path is determined, A^* finds the provably optimal path from the robot's current location to this node by searching its spanning-tree. Also, A^* allows the robot to terminate if there are no unexplored edges left, whereas the standard DFS algorithm would require the robot to physically return to the spanning tree's root in order to terminate.

Figure 2, right, shows a possible spanning tree constructed by the DFS algorithm and short-cuts provided by A^*.

5 Low-Level Reactive Robot Control

The random and the self-organized coverage algorithms rely on three basic behaviors [2]: obstacle avoidance, wall following (left and right), and assessing an objects type (blade, arena boundary, or another robot). Performing boundary coverage and exploring the spanning tree instead requires the following additional behaviors: determining the blade's type (rotor or stator) at the spanning tree's root, navigating to one of two distinct way-points, traversing 8 possible edges (4 for rotor and 4 for stator blades), and finally backing up non-navigable edges (i.e. those ending in a wall). The flow-chart of the robot's controllers is summarized in Figure 3, left, for the random/self-organized coverage algorithm, and in Figure 3, right, for the deliberative-reactive approach.

The type of a blade is determined by measuring the curvature of the blade between its round and its sharp tip. This is achieved by counting the number of increments of the wheels's stepper motors: the round and the sharp tip can be distinguished by the amount of sharp turns necessary for surrounding them. In order to reach a certain level of confidence, a robot might need to circumnavigate a blade multiple times. For instance, for determining the blade type, the difference of "votes" for either type needs to be equal to two, whereby a vote is based on a certain threshold. Parameters determining the termination criteria for the behavioral algorithms have been determined experimentally, and aim at a trade-off between accuracy and time needed.

Action-selection is hard-coded in the *Alice* for the random/self-organized approach, whereas the behavior is selected by the extension module for the deliberative algorithm. After each behavior is terminated, the *Alice* stops, and reports to the DFS algorithm, which in turn selects the appropriate behavior for physically guiding the robot along the spanning tree.

If a behavior has obviously failed (termination criteria not reached within a given time, 10s in our experiments, or a mismatch between the location of the *Alice* and the belief of the DFS algorithm occurred, e.g. being at a wall while the DFS algorithm expects a blade), the *Alice* and the extension module are reset and 10s of obstacle avoidance is executed, in order to restart from a random position.

6 Results

We first compare performance of random exploration, self-organized coordination, and deliberative-reactive coverage for a swarm of 10 real robots. Over 10 experiments, the DFS algorithm needed 788±375s to complete (results with standard deviation), whereas random exploration and self-organization

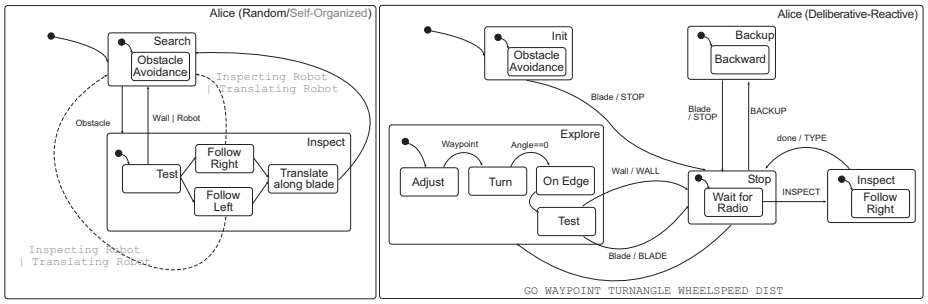

Fig. 3. Flowchart of the robot controller implementing the behavioral layer of the random/self-organized algorithm (left) and deliberative-reactive algorithm (right). *Left*: State transitions requiring communication (self-organized coordination) are dashed / labeled gray. *Right*: On state transitions the *Alice* sends a message to the DFS algorithm on the extension module (capitalized and separated by a "/" from the event). State transitions from the STOP state are always caused by the DFS algorithm.

led to 303±112s and 336±192s, respectively. Compare also Figure 4, left. We also measured the mean time to failure (MTF) in the deliberative-hybrid approach. This is the mean time until the robots reboot due to navigation error, and was measured in terms of distinct blades covered as $MTF(k) = 2.8 \pm 1.4$ blades, and $MTF(t) = 138s \pm 73.3$s in terms of time, leading to an average coverage time of 49.4s per blade.

Inspection performance (time to completion) using non-communicating robots for swarms of 20, 25, and 30 robots (100 experiments each) are contrasted with inspection time for communicating swarms (Figure 4, left, error bars represent the standard error). The absolute improvement of the self-organized approach is given in Figure 4, right. Given the relatively small difference between the results, we applied a non-parametric test for statistical significance (Kruskal-Wallis). Here, experiments involving 30 robots are most significant (p-value equal to 0.07%), whereas results obtained with 20 robots have an estimated chance of 6% to be a random artefact. Additional experiments (32 repetitions) for swarms of 5, 10, and 15 robots did not allow for drawing a significant conclusion (p-value from 62% to 30% for 5 to 15 robots). We conjecture however that the trend—the benefit of communication increases linearly with the swarm size—holds also for smaller team sizes.

7 Discussion

We observe that communication can significantly improve performance of the self-organized coverage algorithm, and performance seems to grow at least linearly with swarm size. Also, the deliberative-reactive approach is outperformed even by half the number of robots performing probabilistic coverage.

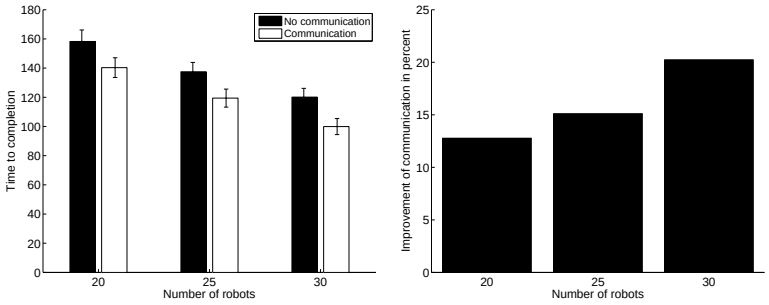

Fig. 4. *Left:* Time to completion for swarms of 20, 25, and 30 robots running self-organized coordination algorithms with and without communication. *Right:* Relative improvement of communication over the probabilistic coverage algorithm without communication for 20 to 30 robots.

This results is mainly due to the fact that the effective time-per-blade in the deliberative approach is very large (49.4s vs. around 15s in the self-organized approach). This difference can be explained by the additional circumnavigations that are necessary in order to determine the blade type, extra time needed for navigating to distinct way-points (due to the crude sensors, a robot might need multiple circumnavigations until it has enough evidence to determine the way-point), and finally due to necessary exploration of dead-ends (arena boundary).

Although the time-per-blade is three to four times higher in the deliberative approach, coverage performance is only by a factor two worse than the probabilistic approach, showing the benefit of planning the trajectories according to a near optimal, deliberate policy.

We notice that in practice the reliability that can be achieved is a function of the time that one is ready to invest to collect sensory information (as we do for determining the blade type that depends on a majority vote). In fact, extensive simulations in *Webots* [13] have shown that relaxing the policies that are necessary for guaranteeing complete coverage, i.e. re-booting on facing a wall instead of exploring the edge and returning to the blade—a sequence of actions that is likely to fail—yielded performance in the range of the self-organized approach with communication.

8 Conclusion and Further Work

We show how concepts from self-organization can be used to design highly robust distributed coordination schemes for boundary coverage. In particular, performance indeed benefits from multiple interactions, which is a key concept in self-organization. Designing such a controller is an iterative process, which is supported by modeling on different abstraction levels [7,8]. This approach is

in strong contrast with a classical design that starts from a provably optimal policy, which is enforced as good as possible.

Acting according to a deliberative scheme might lead to worse performance than a random policy, as the time needed for assessing the environment (navigating on a blade, exploration of dead-end edges) is preventing the robot from actually performing the task. In particular, the complete algorithm proposed in this paper is not feasible to be implemented on the *Alice* platform due to limited computational resources, but needs additional hardware that in itself exceeds the capabilities of the *Alice* robot by an order of magnitude. We thus conjecture that size constraints might make a self-organized approach the sole choice, for instance in inspection tasks inside the human body or micro machinery. We also conclude that evaluating an algorithm's performance solely based on theoretic completeness criteria is misleading as deterministic complete approaches necessarily degenerate to probabilistic completeness under real world constraints. In the future, we would like to analytically assess the trade-off between probabilistic completeness and inspection time for deriving optimal policies knowing the constraints of a particular platform. For instance, by using probabilistic modeling we will be able to estimate how many robots will fail on average, and also how likely it is for a failed robot to meet a robot that is still "on track". We can then show, how much accuracy is needed for executing the behavioral part, so that complete coverage can be achieved by collaboration.

Acknowledgments

The authors would like to thank Jonas Fritschy, Xavier Raemy, Vlad Trifa, Peter Brühlmeier, and André Badertscher for their help with developing the radio modules hard- and software. Both authors are sponsored by a Swiss NSF grant (contract Nr. PP002-68647).

References

1. N. Agmon, N. Hazon, and G. Kaminka. Constructing spanning trees for efficient multi-robot coverage. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1698–1703, Orlando, FL, USA, May 2006.
2. R. Arkin. *Behavior-Based Robotics*. The MIT press, Cambridge, MA, USA, 2nd edition, 2000.
3. E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. SFI Studies in the Science of Complexity, Oxford University Press, New York, NY, USA, 1999.
4. W. Burgard, M. Moors, C. Stachniss, and F. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–378, 2005.
5. G. Caprari and R. Siegwart. Mobile micro-robots ready to use: Alice. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3295–3300, Edmonton, Alberta, Canada, August 2005.

6. N. Correll and A. Martinoli. Modeling and analysis of beacon-based and beaconless policies for a swarm-intelligent inspection system. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2488–2493, Barcelona, Spain, April 2005.
7. N. Correll and A. Martinoli. Collective inspection of regular structures using a swarm of miniature robots. In *Int. Symp. on Experimental Robotics (ISER)*, pages 375–385, Singapore, June 2006. Springer Tracts for Advanced Robotics (STAR), Vol. 21.
8. N. Correll and A. Martinoli. Towards optimal control of self-organized robotic inspection systems. In *8th Int. IFAC Symp. on Robot Control (SYROCO)*, Bologna, Italy, September 2006.
9. 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 research. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Beijing, China, Oct. 2006.
10. K. Easton and J. Burdick. A coverage algorithm for multi-robot boundary inspection. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 727–734, Barcelona, Spain, April 2005.
11. D. Gage. Many-robot MCM search systems. In A. Bottoms, J. Eagle, and H. Bayless, editors, *Proc. of the Autonomous Vehicles in Mine Contermeasure Symp.*, pages 9.55–9.63, 1995.
12. A. Martinoli, K. Easton, and W. Agassounon. Modeling of swarm robotic systems: A case study in collaborative distributed manipulation. *Int. J. of Robotics Research*, 23(4):415–436, 2004.
13. O. Michel. Webots: Professional mobile robot simulation. *Journal of Advanced Robotic Systems*, 1(1):39–42, 2004.
14. J. Polastre, R. Szewczyk, and D. Culler. Telos: Enabling ultra-low power wireless research. In *IEEE/ACM Int. Conf. on Information Processing in Sensor Networks (IPSN-SPOTS)*, Los Angeles, CA, USA, April 2005.
15. L. Shmoulian and E. Rimon. Roadmap-A*: an algorithm for minimizing travel effort in sensor based mobile robot navigation. In *Proc. of the 1998 IEEE Int. Conf. on Robotics and Automation*, pages 356–362, 1998.
16. I. Wagner, M. Lindenbaum, and A. Bruckstein. MAC vs. PC — determinism and randomness as complementary approaches to robotic exploration of continuous unknown domains. *Int. J. of Robotics Research*, 19(1):12–31, 2000.
17. K. Williams and J. Burdick. Multi-robot boundary coverage with plan revision. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1716–1723, Orlando, FL, USA, 2006.