Swarm-bots to the Rescue

R. O'Grady[1], C. Pinciroli[1], R. Groß[2], A. L. Christensen[3], F. Mondada[2], M. Bonani[2], and M. Dorigo[1]

[1] IRIDIA, CoDE, Université Libre de Bruxelles, Brussels, Belgium
[2] LSRO, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
[3] DCTI, Lisbon University Institute, Lisbon, Portugal

Abstract. We explore the problem of resource allocation in a system made up of autonomous agents that can either carry out tasks individually or, when necessary, cooperate by forming physical connections with each other. We consider a group transport scenario that involves transporting broken robots to a repair zone. Some broken robots can be transported by an individual 'rescue' robot, whereas other broken robots are heavier and therefore require the rescue robots to self-assemble into a larger and stronger composite entity. We present a distributed controller that solves this task while efficiently allocating resources. We conduct a series of real-world experiments to show that our system can i) transport separate broken robots in parallel, ii) trigger self-assembly into composite entities when necessary to overcome the physical limitations of individual agents, iii) efficiently allocate resources and iv) resolve deadlock situations.

keywords: self-assembly, task allocation, swarm intelligence, search and rescue, cooperation, group transport, autonomous robots, multi-robot systems, swarm robotics

1 Introduction

Self-assembling robotic systems are made up of autonomous agents that are able to physically connect to one another to form larger composite entities. Self-assembling systems in which the individual agents are themselves simple independent mobile robots potentially share the benefits both of distributed multi-agent robotic systems and of more conventional monolithic robots. In common with distributed multi-agent systems, such self-assembling systems are well suited to parallel task execution, and tend to be relatively inexpensive, robust and scalable. However, by connecting to each other and working together, such systems can also overcome the individual physical limitations of the system's constituent agents and thus carry out tasks that in the past might have required a larger traditional monolithic robot.

However, there is an intrinsic problem of resource allocation that must be solved before self-assembling systems can achieve this kind of flexibility and realise their potential. In particular, a self-assembling system must be able to determine when parallel or collective behaviour is more appropriate, and then be able to distribute resources to reflect this analysis. This problem of resource allocation has been largely ignored in the self-assembly literature. There is a large body of literature on task allocation in non-self-assembling multi-robot systems. However, little of this work is directly applicable to self-assembling systems, where the parameters of the task must determine the nature and extent of physical cooperation at the expense of parallel execution.

In this paper, we explore the problem of resource allocation using a real world group transport scenario. In our scenario, dedicated 'rescue' robots must find broken, immobile robots and transport them to a designated repair zone. The broken down robots can either be single agents or pre-assembled composite robotic entities. A single broken down agent can be transported by a single rescuing agent, whereas a broken down composite entity is sufficiently heavy that multiple rescue robots must self-assemble in order to effect the rescue. The system has no a priori knowledge either of the number of rescue robots or of the number and size of the broken entities.

We present a distributed controller that solves the above task while efficiently allocating resources. Each rescue robot tries to move any broken robots that it finds, and independently determines whether or not the object is successfully being moved. Based on this determination, the rescuing robot uses local communication to either attract or repel other rescue robots. We present a series of experiments with real robots using our controller. The contributions of this study are as follows. Firstly, we demonstrate a new use of self-assembly as a response mechanism. Secondly, we demonstrate a distributed task allocation mechanism based on local attraction and repulsion that is applicable to groups of mobile self-assembling agents. Finally, we demonstrate a group transport control mechanism that improves on previous implementations in both efficiency and flexibility.

2 Related Work

Kube and Zhang [5, 6] conducted a series of experiments in which a group of physical robots was transporting a heavy object. They observed that the robots could end up pushing the same object in opposing directions. As a result of this, the transport object could become stuck. To resolve this problem, they integrated into their control policy a recovery mechanism that was inspired by the group transport of the ant species *Pheidole crassinoda* [9]. In this species, ants were reported both to undergo changes in their group transport arrangements and to recruit nest mates if an object resisted motion (Kube and Zhang's robots mimicked the group transport rearrangement behaviour).

In simulation, Perez-Uribe *et al.* [8] investigated a system in which a group of robots is required to find and transport multiple objects of two sizes: small and large. The allocation of robots to the objects was irreversible, thus creating a deadlock potential in the case of an immovable object.

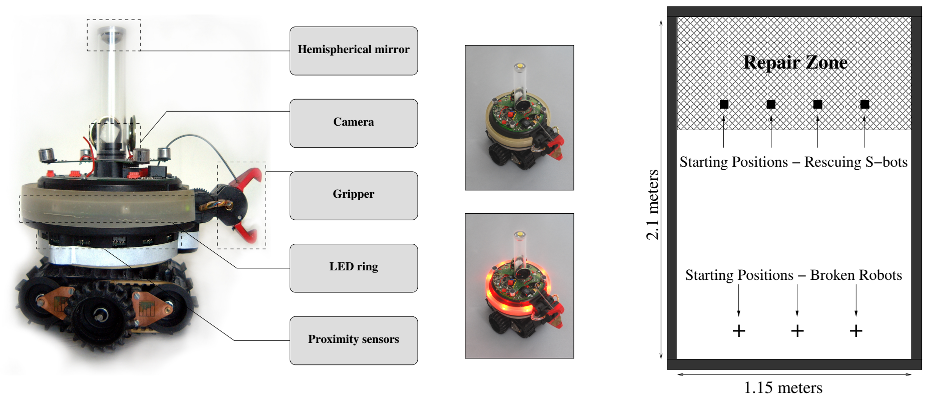


Fig. 1. Left: The s-bot. Centre: S-bot LEDs off and on. Right: The arena.

Ijspeert *et al.* [4] studied a system of physical robots in which two physically cooperating robots could pull long sticks out of the ground (the sticks could not be removed by a single robot). When a single robot tried to remove a stick, it would wait for another robot to arrive. The optimal waiting time was computed as a function of the environmental parameters.

Groß and Dorigo [1–3] used computer simulations to study the transport of heavy objects by groups of self-assembling robots. The control policies were designed by evolutionary algorithms. In [2, 3], it was assumed that only a single transportable object was present. The system in [1] could in principle cope with multiple objects, however, each robot would always attempt to transport the closest object within its perceptual field. The allocation of robots was irreversible and did not depend on the objects' resistance to motion.

Tuci *et al.* [10] conducted an experiment with physical robots that could self-assemble and transport a heavy object. The transporters were programmed to suspend their transport whenever they perceived an unconnected robot (allowing the latter to join the group). Thus, if a robot permanently failed to join the pulling structure, a deadlock occurred.

3 Platform and Experimental Setup

The platform we use is the swarm-bot robotic platform [7]. The swarm-bots platform consist of a number of autonomous robots called *s-bots*, see Fig. 1 (left). The s-bot is 12 cm high without its camera turret, and has a diameter of 12 cm. Each s-bot is equipped with a gripper that enables physical connections between s-bots. The turret holding the gripper can rotate independently of the chassis, that contains a differential drive system composed of combined tracks and wheels. This allows an s-bot to exert force in any direction when grasping another s-bot or object.

Each s-bot is equipped with torque sensors on the tracks motors that let the robot determine whether it is successfully moving a gripped object that it is trying to transport. If the s-bot is moving the object, the s-bot's tracks will be rotating and the motor torque will be low. If, however, the object is not moving (for example, because it is too heavy) the rescue robot's tracks will be blocked and the torque will be high.

The s-bot camera records panoramic images reflected in a hemispherical mirror mounted above the s-bot chassis in a transparent perspex tube. Each s-bot has a semi-transparent ring housing eight sets of RGB coloured LEDs. Depending on light conditions, the camera can detect illuminated LEDs on other s-bots up to 50 cm away. A strong external light source can be seen up to 4 m away.

Our experimental setup requires 'rescue' s-bots to search the arena shown in Fig. 1 (right), to find broken robots and then to transport them to the designated repair zone (darker floor). A light source is located two meters outside of the arena beyond the repair zone (not shown in Fig. 1). Transporting s-bots perform phototaxis to ensure that they transport the broken robots in the correct direction towards the repair zone.

All of our experiments involve either one or two rescuing robots and either one or two broken robots. We use two types of broken robot. We refer to a single broken s-bot as a *1-s-bot broken robot* (a single rescuing s-bot is able to transport a 1-s-bot broken robot). We refer to a broken robot that is a composite entity made up of two physically connected s-bots as a *2-s-bot broken robot* (the rescuing s-bots must team up in order to collectively transport a 2-s-bot broken robot). We consider broken robots to be immobile. However, we make the simplifying assumption that broken robots are still able to use their LEDs to signal that they require assistance.

4 Controller

The desired behaviour of our multi-agent system is for the rescue s-bots to locate and transport the broken robots to the repair zone. Ideally, based on the properties of the broken robots (size, weight), only the minimum required number of rescue s-bots should be allocated to transport each broken robot, thus leaving as many rescue s-bots as possible idle or available for other tasks.

We designed our distributed controller to use only local sensing and communication. The control logic executed independently by each rescue s-bot is shown in Fig. 2. A rescue s-bot searches for any broken robots by performing a random walk with its blue LEDs illuminated (`Random Walk`). Broken robots signal that they need help by illuminating their red LEDs. When a rescue s-bot detects red LEDs without any nearby green LEDs, it assumes that it has seen a broken robot that is not currently being rescued and heads towards the red LEDs (`Goto Red Entity`). If more than one red entity is seen, the rescuing s-bot picks one of the red entities at random. It then grips the broken robot (`Grip Red Entity`) and tries to pull the broken robot to the repair zone by heading towards the light source (`Pull Towards Light`) with its green LEDs illuminated.

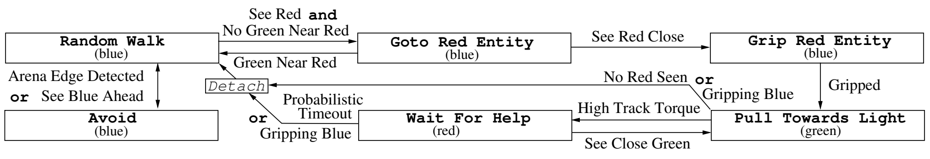


Fig. 2. Control logic that each rescue s-bot executes independently.

As long as a rescue s-bot is successfully pulling a broken robot (low track torque), it stays green. This tells other rescue s-bots within camera range not to go towards the broken robot it is pulling. However, if the rescue s-bot fails to move the broken robot (high track torque measured consistently for 5 seconds), the rescue s-bot stops pulling and illuminates its red LEDs (`Wait For Help`). At this point, the rescue s-bot is indistinguishable from the broken robot that it is still gripping. Thus, other rescue s-bots will approach and grip either the broken robot or any rescue s-bots that are attached to the broken robot and are in state `Wait For Help`. When a new rescue s-bot attaches, it will turn green as it attempts to pull the broken robot. Any attached s-bots in state `Wait For Help` are prompted by the sight of green LEDs to try to pull again. If the new larger number of attached rescue s-bots is now sufficient to move the broken robots, all of the rescue s-bots will stay green and thus prevent any further rescue s-bots from approaching. Otherwise, after 5 seconds of high torque, they will realise that the broken robot is still not movable, and will switch to state `Wait For Help`. This process repeats itself, with progressively more rescue s-bots attaching to the broken robot until there are enough rescue s-bots to move the broken robot. In this way, the system automatically finds the minimum group size capable of moving a broken robot.

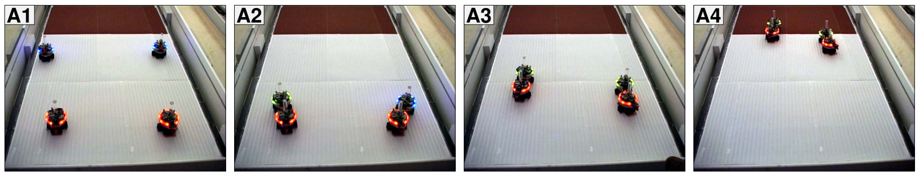
If there are several large composite broken robots present, and a small number of rescue s-bots, it is possible that the individual rescue s-bots might randomly distribute themselves among the broken robots in such a way that none of the large broken robots can be moved. To prevent this type of deadlock situation, a rescue s-bot in state `Wait For Help` has a low probability of detaching. A detached robot returns to state `Random Walk` and thus turns blue, which signals to any s-bot that might be gripping the detached robot that it should in turn detach. Rescuing robots also detach if they no longer detect any red LEDs—in our experiments, the broken robots detect arrival in the repair zone using ground sensors, and switch off their red LEDs. Together, these two detachment mechanisms ensure that as long as there are enough rescue s-bots present in the environment to move any given broken robot, the rescue s-bots will eventually combine forces to move the broken robot.

5 Results

We carried out four sets of real-world experiments. Each set of experiments is defined by the number of rescuing s-bots, the number of broken robots and the size of the broken robotic entities. The rescuing robots have no a priori knowledge of the experiment configuration. Videos of the experiments can be found on the web at: `http://iridia.ulb.ac.be/supp/IridiaSupp2009-011`.

Rescuing Broken Robots in Parallel

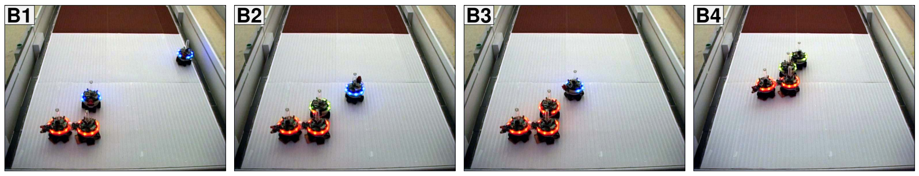
We conducted 5 trials of an experiment with two rescue s-bots and two 1-s-bot broken robots (photos A1-A4 are of a single trial). In each experiment, the system successfully allocated a single rescue robot to each of the broken robots and the broken robots were transported in parallel to the repair zone.



This experiment shows that when possible the system correctly 'chooses' parallel execution (the incorrect choice would be for two rescue s-bots to assemble to the same broken robot).

Physical Cooperation to Rescue a Broken Robot

We conducted 5 trials of an experiment with two rescue s-bots and a single 2-s-bot broken robot (photos B1-B4 are of a single trial). In each trial, one rescue s-bot found the broken robot, then tried and failed to move the broken robot alone. The attached rescue s-bot waited for help, and after the other rescue s-bot attached, together the two rescue s-bots succeeded in transporting the broken robot to the repair zone.

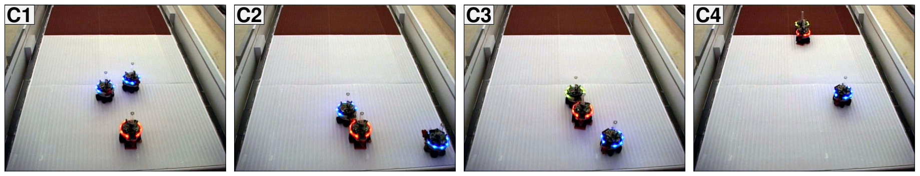


In one of the trials, the initial attachment configuration of the rescue s-bots failed to move the broken robot. One of the rescue s-bots detached based on the probabilistic time out twice, but reattached both times. In the final configuration the rescue s-bots succeeded in transporting the broken robot to the repair zone.

This experiment shows that the system correctly allows the rescue s-bots to physically coordinate in order to solve a task that is beyond the physical capacities of a single s-bot.

Efficiency Gains Through Group Size Regulation

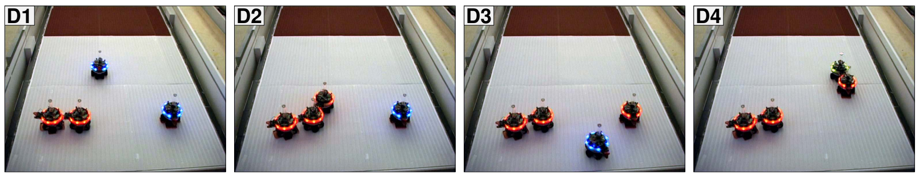
We conducted 5 trials of an experiment with two rescue s-bots and a single 1-s-bot broken robot (photos C1-C4 are of a single trial). In each trial, a single rescue s-bot connected to the broken robot, and successfully transported it to the repair zone. The other s-bot continued to explore the arena without attempting to attach to the moving broken robot.



This experiment shows that the system correctly allocates the minimum number of rescue s-bots (in this case 1 s-bot) to a task, and leaves other rescue s-bots free to potentially carry out other tasks.

Reallocation of Resources in a Deadlock Situation

We conducted 5 trials of an experiment with a single rescue s-bot and two broken robots: one 1-s-bot broken robot and one 2-s-bot broken robot (photos D1-D4 are of a single trial). In this experiment, we initially 'break' the 2-s-bot broken robot (i.e., illuminate its red LEDs). We allow the rescuing s-bot to find and attach to the heavy broken robot before we 'break' the 1-s-bot broken robot.



In four out of the five trials, the rescue s-bot probabilistically timed out, explored the arena, found and attached to the 1-s-bot broken robot and successfully transported it to the repair zone (in one of these successful trials the rescue s-bot first re-attached to the 2-s-bot broken robot and timed out again). In a single trial, the rescue s-bot failed to detach correctly due to a hardware failure.

This experiment shows that the system correctly resolves a deadlock situation where a rescue s-bot is attempting an impossible task, and manages to reallocate resources (the rescue s-bot) to another task that is feasible.

6 Conclusion

In this study, we presented a distributed controller that for the first time tackles the problem of resource allocation in a self-assembling robotic system. We conducted real world experiments in a group transport rescue scenario which showed that our distributed control logic displays several important properties. In particular, our system was able to maximise parallel execution (and hence efficiency) by allocating the minimum number of agents required to solve tasks of varying magnitudes. In addition, the system included a reset mechanism which allowed it to resolve potential deadlock situations (situations in which a given distribution of agents at a particular moment in time results in all attempted tasks being impossible to carry out). Interestingly, we discovered that this mechanism also allowed for potentially beneficial random reconfiguration of spatial arrangements within a single transport rescue group.

We are currently working on testing the scalability of the system with larger numbers of robots in simulation, and on abstracting the fundamental dynamics of our system so as to apply them to other self-assembly scenarios.

7 Acknowledgements

This work was supported by the Sixth Framework Programme of the European Commission in the form of the IST FET project SWARMANOID (contract IST-022888), of the Marie Curie Early Stage Research Training Site COMP2SYS (contract MEST-CT-2004-505079), and of a Marie Curie Intra-European Fellowship (contract MEIF-CT-2006-040312). Marco Dorigo acknowledges support from the F.R.S.–FNRS of the Belgian French Community, of which he is a research director, via the VIRTUAL SWARMANOID project. The information provided is the sole responsibility of the authors and does not reflect the European Commission's opinion. The European Commission is not responsible for any use that might be made of data appearing in this publication.

References

1. R. Groß and M. Dorigo. Group transport of an object to a target that only some group members may sense. In *Parallel Problem Solving from Nature – 8th Int. Conf. (PPSN VIII)*, volume 3242 of *LNCS*, pages 852–861. Springer Verlag, Berlin, Germany, 2004.
2. R. Groß and M. Dorigo. Evolution of solitary and group transport behaviors for autonomous robots capable of self-assembling. *Adapt. Behav.*, 16(5):285–305, 2008.
3. R. Groß and M. Dorigo. Towards group transport by swarms of robots. *Int. J. of Bio-Inspired Computation*, 1(1–2):1–13, 2009.
4. A. J. Ijspeert, A. Martinoli, A. Billard, and L. M. Gambardella. Collaboration through the exploitation of local interactions in autonomous collective robotics: The stick pulling experiment. *Autonomous Robots*, 11(2):149–171, 2001.
5. C. R. Kube and H. Zhang. Stagnation recovery behaviours for collective robotics. In *Proc. of the 1994 IEEE/RSJ/GI Int. Conf. on Intell. Rob. and Sys.*, volume 3, pages 1883–1890. IEEE Computer Society Press, Los Alamitos, CA, 1994.
6. C. R. Kube and H. Zhang. Task modelling in collective robotics. *Autonomous Robots*, 4(1):53–72, 1997.
7. F. Mondada, G. C. Pettinaro, A. Guignard, I. V. Kwee, D. Floreano, J.-L. Deneubourg, S. Nolfi, L. M. Gambardella, and M. Dorigo. SWARM-BOT: A new distributed robotic concept. *Auton. Robots*, 17(2–3):193–221, 2004.
8. A. Perez-Uribe, D. Floreano, and L. Keller. Effects of group composition and level of selection in the evolution of cooperation in artificial ants. In *Proc. of the 7th European Conf. on Artificial Life*, volume 2801 of *Lect. Notes Artif. Int.*, pages 128–137. Springer Verlag, Berlin, Germany, 2003.
9. J. H. Sudd. The transport of prey by an ant, *Pheidole crassinoda* Em. *Behaviour*, 16:295–308, 1960.
10. E. Tuci, R. Groß, V. Trianni, F. Mondada, M. Bonani, and M. Dorigo. Cooperation through self-assembly in multi-robot systems. *ACM Trans. Auton. Adapt. Syst.*, 1(2):115–150, 2006.