Metamodule Control for the ATRON Self-Reconfigurable Robotic System

David Johan Christensen   Esben Hallundbæk Østergaard
Henrik Hautop Lund
The Maersk Mc-Kinney Moller Institute for Production Technology
University of Southern Denmark
{david, esben, hhl}@mip.sdu.dk

Abstract.
This paper presents an approach for distributed control for the individual modules in the ATRON self-reconfigurable robotic system. The ATRON is a lattice-type module, which has a single degree of freedom. The system is reconfigured using metamodules, which are simple, consisting of only three modules. But they have far higher capabilities than a single module, because they can crawl relatively freely on the surface of other modules. We let the metamodules emerge from the structure and migrate from one place to another, before they die and become structure again. Then, by controlling the flow of metamodules, we can change the shape of the structure. We introduce three different algorithms for migration of metamodules on the structure, and show that by using these, we can make the structure change its shape significantly in a controllable manner.

1 Introduction

The field of self-reconfigurable robots has received an increased interest since Fukuda's ground-breaking research in 1988 [4]. As biological organisms are built from cells, so are self-reconfigurable robots built up from simple modules. A module has the ability to communicate locally with nearby modules and to do simple motor actions. The modules are connected to form a structure, for example a lattice or a chain structure. The hope is that such robot systems will inherit the robustness and flexibility properties of biological organisms.

One central research question is how to control such individual modules, in order to achieve the desired global behaviour of the system, for example reshaping into a given form like a box, a sphere or a snake. There are two approaches to the control problem. 1) Centralized control, where a single entity knows everything that goes on in the system, and, on the basis of this global knowledge, plans how the modules should act [5, 13, 14, 17]. 2) Distributed control, where every module is autonomous and there is a lack of an all-seeing central entity [2, 6, 9, 12, 18]. Both centralized and distributed control is a possibility, but some effects of centralized solutions are not desirable. For example, planning does not scale well with respect to the number of modules. The strongest argument for distributed control is perhaps that nature seems to prefer this solution to control cells [7].

When controlling a self-reconfigurable robot, the global behaviour of the system must arise from simple motor actions performed by individual modules. This coordination problem

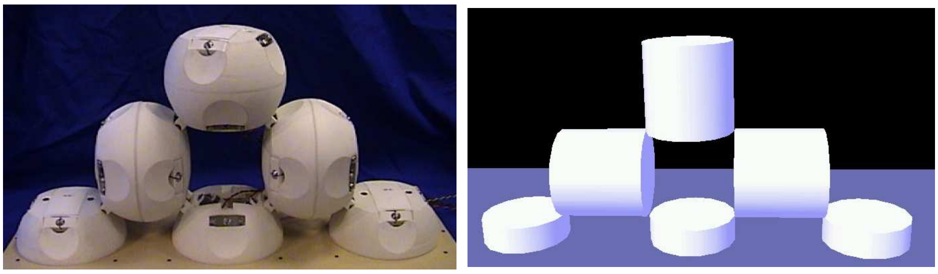

Figure 1: A physical prototype of three whole ATRON modules and three half ones on the left and their representation in the simulator on the right.

can be solved ”top down”, where a given task is divided into smaller problems that can be solved by the individual modules, or ”bottom up” where controllers in the individual modules make the global solution emerge. The approach presented here is a mix between the two approaches, but arises from the ”bottom-up” approach.

In this paper we try to deal with some of the control complexity of distributed controllers, by having a small number of modules cooperate in goups, to form “metamodules”. In [8, 10, 11, 15], a metamodule is a group of modules working together to reduce the control complexity of the system. In existing work metamodules cause the granularity of the system to increase, because the metamodules form lattices. In our approach we consider metamodules that do not neccesarily sit in lattice structures, but rather metamodules that can crawl freely on a surface of other modules. Also, the metamodules, which we consider, are formed during the lifetime of the system, when they emerge (from the structure), migrate, and die (become structure).

2 ATRON self-reconfigurable robot system

The ATRON self-reconfigurable robot system consists of a number of ATRON modules. The module is designed for simplicity. It has a single degree of freedom, rotation around an axis that goes through the centre of the module. The module is built up by two half-spheres that are able to rotate relative to each other and able to connect to eight other modules, four on each half, see figure 1. As can be seen, the modules function in a three-dimensional space and sit in a kind of grid. In this paper they will always be rotated in steps of 90 degrees.

The simplicity of the ATRON module results in positive as well as negative aspects in terms of production and control. Amongst the positive aspects is the ability to act as building-block in more complex building block systems that may be simpler to control. Also, there is a better chance that the simplicity of the modules will make them useful in a variety of tasks and environments. The trouble and cost of production will also be a dominant factor in the success of a particular module, and simple modules will probably be cheaper to mass-produce.

Amongst the negative aspects is the reduced mobility of the individual module, which will make the control system more complex. A module cannot move on its own, therefore, a second module must always be involved to rotate the first module. This means that modules must cooperate with other modules to achieve even simple things. Because of the limited sensing ability individual modules can only sense whether they have a neighbour on a particular connector. This makes it hard for the modules to avoid making illegal actions, like crashing into other modules, rotating when not possible - because of connection to other modules or dis-

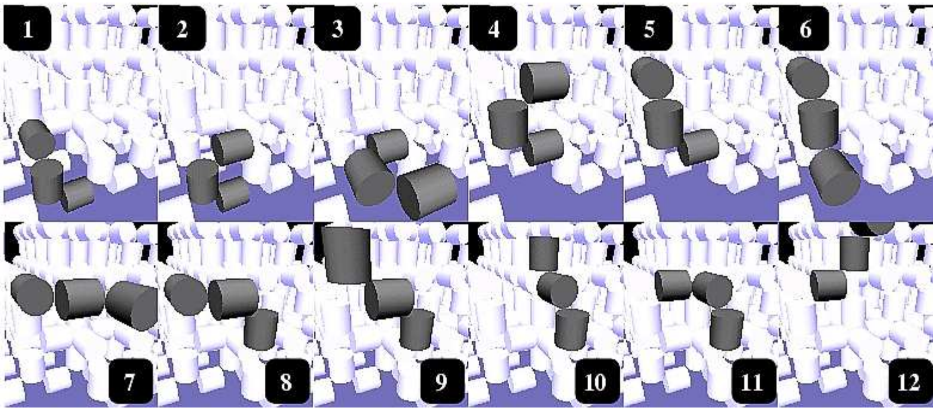

Figure 2: A metamodule demonstration of good mobility capabilities which enable it to to crawl away from its starting position without help from its environment.

connection from the rest of the structure. Amongst the physical limitations of the modules is a single module's ability to rotate a maximum of two other modules and still fit into the grid of modules.

3 Controlling the modules

It seems that there is no simple and obvious way to solve the general problem of self-reconfiguring. Specific solutions to specific problems can be found in some cases. But to make mechanisms that solve the general problem of self-reconfiguration in a distributed way on "real" modules can be very complex. Finding the local rules that produces the desired reconfiguring while avoiding illegal actions, is hard or impossible. In general, we would like to have a generic mechanism, which will enable the system to go from one shape to another with few or no limitations. One way we could imagine doing this, is by having very powerful/movable modules that are able to crawl freely on the surface of other modules. Then, if we are able to control the "flow" of modules from one place to another, we can also obtain the desired shape. This idea is similar to the approaches of [1] and [16], which are done with the very movable Proteo module. A single ATRON module can not fulfil the criteria of movability, but a well-chosen metamodule made up of a number of ATRON modules might.

3.1 Metamodule Design

We wish to define a collaboration between a number of modules to get an abstraction of these into a metamodule. This new metamodule should have better capabilities, especially in terms of mobility without compromising the physical constraints.

The chosen metamodule consists of three individual modules. It has a central module with one outer module connected to each of its halves. Figure 2 demonstrates some of the metamodule abilities, crawling from one place to another. Furthermore, it can be positioned in such a way that the three modules are orientated with rotation axes in the x, y and z direction, such that a structure consisting only of metamodules can be packed optimally tight. To simplify the control of the modules, we make the constraint that only the outer half of the outer modules must be connected itself to the structure. With this constraint the metamodule can perform a total of eight different actions: It can stand on one of the outer modules and make a rotation one way or the other using either the module it is standing on or the central module.

3.2 Controlling the Metamodules

We wish to let the metamodules emerge from the structure and then during their life time migrate from one place to another on the structure, before they again die becoming structure again. This approach models division, migration and death of biological cells. In addition, it is expected to be more flexible, because the metamodules can emerge more freely whenever and wherever there is an opportunity or a need. And, finally, it does not compromise the granularity of the system since every metamodule can be decomposed into its parts, which otherwise is a problem as pointed out in [15].

Controlling the metamodules is then the job of deciding when they should emerge, which choice of actions they should perform when alive, and when they should die.

Emergence A metamodule comes to life by probability: There is a slight chance that a module will try to become the centre part of a metamodule and that it will succeed if there is a possibility i.e. if it has "willing" modules on each half. Otherwise there are no constraints on the emergence of metamodules, which means that a metamodule can emerge from very different starting orientations, and that a module, at different times, can be part of different metamodules.

Life In this paper we explore three different approaches for controlling a living metamodule. Each of them are given the orientation of the metamodule, and the local configuration; then the set of all legal metamodule actions are computed. The algorithm differs depending on the way the metamodule chooses from the set of legal actions.

Death To make the metamodule die, a virtual energy is introduced, which is lost as time goes, and lost faster if it is not able to move. When a metamodule runs out of energy, it ceases to exist.

The three "life" strategies explored in this paper are:

Virtual Force Algorithm In this algorithm the metamodule is attracted towards a particular point in space. This can be done by having the modules share a common coordinate system. Then the algorithm picks a random action in the set of legal metamodule actions, but with a higher probability for actions that lead towards the goal than for those which do not. Adjusting this probability is a compromise between, on the one hand, making the metamodule more likely to get stuck in local minima, and, on the other hand, being inefficient.

Gradient Algorithm In this algorithm the metamodule is attracted in a particular direction. A module may send out a gradient(its metaphor could be *scent* as in [1]), which will travel on the surface of the structure. A metamodule can then, by following the gradient, move towards the module or group of modules that are sending out the gradient. Similar to the virtual force algorithm, the metamodule tries to follow the gradient with some probability. But still it can be seen from experiments that it has a higher tendency to get trapped in local minima.

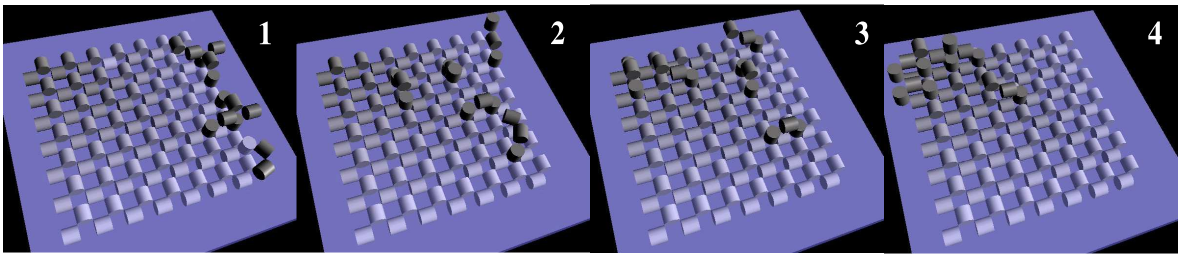

Figure 3: Experiment 1: Module, in the top left corner, sending out a gradient and thereby attracting metamodules. The gradient intensity is indicated by the colour of the modules. Darker structure-modules have a higher gradient intensity while the metamodules are black.

Gradient and Pheromone Algorithm This algorithm expands the gradient algorithm by using a mechanism, which is similar to pheromone. Pheromone is a chemical, which is used by animals to communicate with each other. For example ants will find the shortest way from the nest to a food-source by using an pheromone laying system [3]. The motive is to reduce the tendency of metamodules to get stuck in local minima. To do this, we let the metamodules lay pheromone on every module they connect to. If no metamodule crawls on a particular module, the pheromone on it will evaporate. This can be seen as a form of distributed memory that remembers if the metamodule keeps circling around in the area; a continuous circling-around may suggest that the metamodule is stuck in a local minima. If the pheromone level then reaches a certain threshold, the metamodule starts to move randomly until it is free of the local minima.

4 Experiments

In this section we present experiments which are done using metamodules and the three different algorithms.

4.1 Experiment 1: Calling Module

This experiment shows how a single module can attract metamodules by sending out a gradient. The experiment can be seen in figure 3. There is a floor of structure-modules, and the metamodules walk on it. This attraction (or repelling) of modules can be used as a tool for controlling the flow of metamodules from one place to another. The randomness in the algorithms enables the metamodules to avoid deadlock situations.

4.2 Experiment 2: Building a Tower

By applying different virtual forces to the system, we are able to manipulate self-reconfiguration. The experimental setup and result are shown in figure 4. The starting configuration of the structure is a pyramid. By using the ”Virtual Force” algorithm to attract the metamodules towards a point high above the top of the structure, we are able to make the structure stretch towards that goal. In similar ways one could make different structures emerge, by pulling and pushing the metamodules.

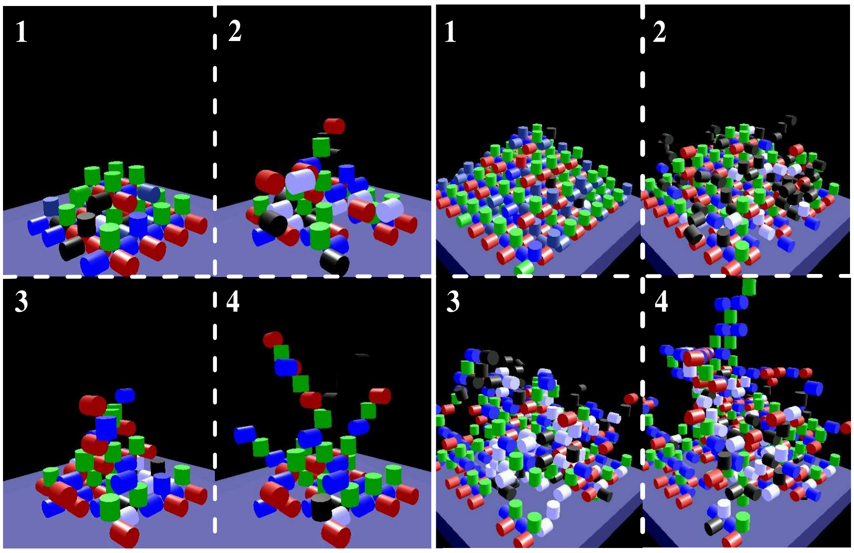

Figure 4: Building a tower by using the "Virtual Force" algorithm, pulling the metamodules upwards. On the left the starting configuration is a pyramid made up of 60 modules and on the right 360 modules. The programs run on them are identical.

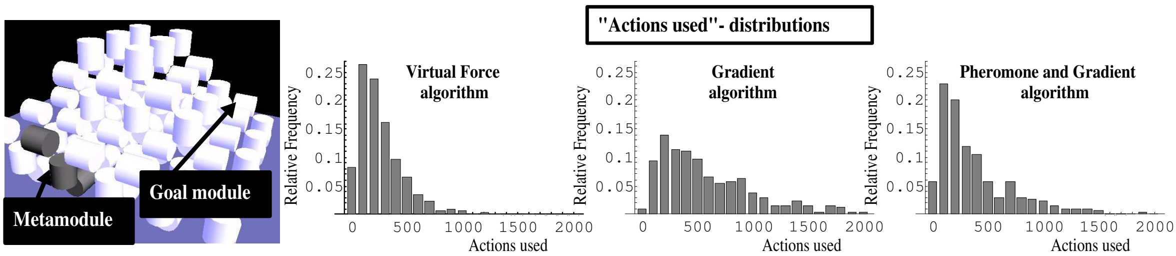

Figure 5: On the left the experimental setup is shown. On the right diagrams that shows the distribution of the number of actions it took to solve the task for the different algorithms. Therefore, the better the algorithm performs, the higher the columns in the left end of the graph.

4.3 Experiment 3: Comparing the Algorithms

The performance of the algorithms on a sample experiment can indicate how they compare to one another. The experimental setup can be seen in the left of figure 5. The module must travel from its starting position to the other side of the structure indicated on the figure. The performances of each of the three algorithms are measured as the number of actions needed for performing the simple task. A number of experiments is run, and the result is summarized in table 1 and on the right of figure 5. From this it can be seen that the "Virtual Force" algorithm outperforms the others, but also that the "Gradient and Pheromone" algorithm performs very reasonably.

5 Discussion and Further Work

The control complexity of the ATRON module is hard to handle. It is a simple module, with not many capabilities of its own. Also, it is enormously easy to perform illegal actions that

Table 1: Experiments results from the comparison of the three algorithms.

Algorithm	Experiments	μ	σ	Min	Max
Virtual Force	500	315	215	49	1696
Gradient	500	669	478	46	2809
Gradient & Pheromone	500	464	464	56	4268

may break the modules. In this light the metamodule approach presented in this article is quite successful in encapsulating the dangers and improving the self-reconfiguring ability. Also, the principle of letting the metamodules emerge and die seems to make the system more flexible. In other experiments with constraints on the starting and ending orientation of the metamodule, the ability to self-reconfigure was much more limited. But the lack of orientation constraints also results in situations where parts of or the entire structure is not able to produce a sufficient number of movable metamodules to obtain a reasonable speed of the self-reconfiguring process. This could perhaps be handled by introducing new well-considered constraints on the metamodules, which would keep the structure tighter packed so that it would be easier to crawl on, would hold a larger number of potential metamodules and reduce the unaddressed problem of physical stability.

The control algorithms for moving the metamodules on the structure can be improved significantly, which can be seen from the performance experiment. The less time a metamodule spends on crawling around, the less time it will block the way for other metamodules, so in that sense the time to perform a given reconfiguring can be drastically lowered by having more efficient algorithms.

Further work will also go into more experiments on how to build more complex structures with desired functionality from the existing algorithms. One experiment could for instance be aimed at building a bridge, where a particular module could be pointed out as the basis of the bridge, which could then attract the building material by sending out gradients.

The principle of having simple modules and using a scheme with simple metamodules to control them makes both the control and production of self-reconfigurable robotics systems much more manageable. This could be thought of as an alternative to more complex modules, which have comparable characteristics to metamodules, but may be harder to build in practice, and lacks the ability to be divided into smaller parts.

6 Conclusion

We have designed a metamodule for the ATRON self-reconfigurable robotic system, which has much better capabilities than a single module for crawling on the surface of other modules. We let the modules emerge as metamodules for movement and let them die becoming normal modules again when the metamodule no longer should move. This gives a more flexible system, where each module can become part of different metamodules during its lifetime without compromising the granularity. Three ways of controlling the metamodules are introduced using the metaphors of force, gradient and pheromone. These algorithms manage to move the metamodules around on the structure, making it reconfigure into different structures, showing that ”self-reconfiguration” is possible using the presented metamodule approach.

7 Acknowledegements

This work was performed as part of the HYDRA project funded by the EUs Information Society Technologies (IST) Programme (Future and Emerging Technologies).

References

[1] H. Bojinov, A. Casal, and T. Hogg. Emergent structures in modular self-reconfigurable robots. In *Proceedings, IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 1734–1741, 2000.

[2] H. Bojinov, A. Casal, and T. Hogg. Multiagent control of self-reconfigurable robots. In *Fourth International Conference on MultiAgent Systems (ICMAS)*, pages 143–150, 2000.

[3] M. Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, 26(1):29–41, 1996.

[4] T. Fukuda and S. Nakagawa. Dynamically reconfigurable robotic system. In *Proceedings, 1988 the IEEE Int. Conf. on Robotics & Automation*, 1988.

[5] K. Kotay and D. Rus. Algorithms for self-reconfiguring molecule motion planning. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2000.

[6] J. Kubica, A. Casal, and T. Hogg. Complex behaviors from local rules in modular self-reconfigurable robots. In *Proceedings of the IEEE Int. Conference on Robotics and Automation*, volume 1, pages 360–367, May 2001.

[7] H. Meinhardt. A model for pattern formation of hypostome, tentacles, and foot in hydra: how to form structures close to each other, how to form them at a distance. *Developmental Biology*, 157(2):321–333, June 1993.

[8] A. Nguyen, L. Guibas, and M. Yim. Controlled module density helps reconguration planning. In *In Proceedings of 4th International Workshop on Algorithmic Foundations of Robotics(WAFR)*, 2000.

[9] E. H. Ostergaard and H. H. Lund. Evolving control for modular robotic units. In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, 2003.

[10] K. C. Prevas, C. Unsal, M. O. Efe, and P. K. Khosla. A hierarchical motion planning strategy for a uniform self-reconfigurable modular robotic system. In *IEEE International Conference on Robotics and Automation (ICRA)*, May 2002.

[11] D. Rus and M. Vona. Crystalline robots: Self-reconfiguration with compressible unit modules. *Autonomous Robots*, 10(1):107–124, Jan. 2001.

[12] K. Tomita, S. Murata, H. Kurokawa, E. Yoshida, and S. Kokaji. Self-assembly and self-repair method for a distributed mechanical system. *IEEE Transactions on Robotics and Automation*, 1999.

[13] C. Unsal and P. Khosla. A multi-layered planner for self-reconfiguration of a uniform group of i-cube modules. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.

[14] C. Unsal, H. Kilite, and P. K. Khosla. A modular self-reconfigurable bipartite robotic system: Implementation and motion planning. *Autonomous Robots*, 10(1):23–40, 2001.

[15] S. Vassilvitskii, J. Kubica, E. Rieffel, J. Suh, and M. Yim. On the general reconfiguration problem for expanding cube style modular robots. In *Proceedings of the 2002 IEEE Int. Conference on Robotics and Automation (ICRA)*, pages 801–808, 11-15 May 2002.

[16] M. Yim, Y. Zhang, J. Lamping, and E. Mao. Distributed control for 3d metamorphosis. *Autonomous Robots*, 10(1):41–56, 2001.

[17] E. Yoshida, S. Murata, and A. Kamimura. A motion planning method for a self-reconfigurable modular robot. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001.

[18] E. Yoshida, S. Murata, K. Tomita, H. Kurokawa, and S. Kokaji. Distributed formation control for a modular mechanical system. In *Proceedings of IEEE-RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1090–1097, 1997.