Self-replicating Robots for Monitoring and Surveillance

Zack Butler, Daniela Rus*

1 Introduction

Self-reconfigurable robots are modular robots that can morph into different shapes without outside assistance. The ability to change shape allows these robots to adaptively execute a variety of different tasks, such as locomotion through small passages, climbing tall obstacles, and moving while supporting large payloads, that a single fixed-architecture robot would be incapable of. Self-reconfiguring robots can also function as parallel distributed machines.

We consider the self-replication ability in which a large robot can divide itself in several independent smaller robots with the same basic functionality (but not identical size). For example, a system consisting of 100 modules could function as one large robot or 10 smaller robots each consisting of 10 modules, or any number of other configurations. Self-replicating robots are useful in tasks where the overall effectiveness and task completion time is improved by parallelism, such as distributed surveillance or exploration. Consider, for example, the distributed surveillance and monitoring task — a single robot can only survey the portion of the environmnet currently in its field of view; a robot that can self-replicate will enlarge the overall field of view. Similarly, a single robot requires longer to traverse a path in the environment. A robot that can split itself into several pieces can take advantage of parallelism to complete the exploration task faster.

We address the vision of self-replicating robots by developing distributed algorithms for controlling how to divide a self-reconfiguring robot into smaller independent robots, and recombining several smaller robots into a larger robot. Our algorithms are homogeneous and purely distributed, in that each module runs the same algorithm (with no central controller) and uses only local information to determine its actions. This is important here, as groups of different sizes will be created, and the algorithms must work correctly regardless of the overall group size. These algorithms have been implemented in simulation and also in hardware on the Crystal robot, which is a modular self-reconfiguring robot system. The poster will detail the algorithms and show simulations. There will be a hardware demonstration to accompany the poster.

2 Algorithms

We have developed distributed algorithms that allow modular robots to split up when they need to explore different directions, or when the surveillance task is enhanced by parallelism. The *division* operation splits up a robot into two smaller groups. This kind of robotic self-replication can be controlled in a distributed fashion, using the same approach as for locomotion. The key is to define a set of rules that select which robot modules belong to which smaller group by the local neighborhood only. The end result is local control that is guaranteed to split an undifferentiated modular robot into differentiated groups.

The basic division control algorithm requires a state variable corresponding to the moving direction. All robot modules begin in an undifferentiated state and divide into two groups: left-facing and right-facing The module differentiation is done by propagating of an integer variable called a *signal* through the system. As the modules finish dividing (which they can detect by comparing signals in a local neighborhood), they begin to locomote using the a distributed locomotion approach based on local rules. Merge is similar.

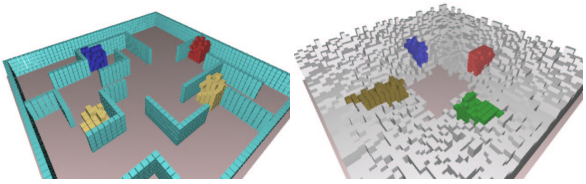

Figure 1: (Left) Four groups of modules exploring in parallel a building. (Right) Groups of modules exploring rough terrain.

We can use the division and merging algorithms within a locomotion task, which can lead to application in more general domains where the ability to split is useful for speeding up task completion, and the ability to merge is useful for climbing over high obstacles. Consider the tasks displayed in Fig. 1. The left image shows a simulation snapshot of a system exploring a maze-like environment, with groups splitting up to more efficiently cover the maze. Here, the number of groups could be increased as the complexity of the maze increases. The right image shows a simulation snapshot from a system in which several groups are covering an open piece of terrain. In this case, a larger number of groups is desirable to speed up exploration, but small groups cannot navigate large obstacles. Therefore, the system can use the ability to dynamically change group size and number to more efficiently conduct exploration.

3 Hardware

The physical implementation of these algorithms has been done on the Crystal Robot. This robot use a novel actuation mechanism, scaling, which gives more robust motion than the previous rotation-based actuation systems.

Figure 2: A Crystal robot consisting of nine modules.

The Crystal Robot (Figure 2) has some of the motive properties of muscles and can be closely packed in 3D space by attaching units to each other. Each module is actuated by expansion and contraction. By expanding and contracting the neighbors in a connected structure, an individual module can be moved in general ways relative to the entire structure. Crystal atoms never rotate relative to each other; their relative movement is actuated by sliding via expansion/contraction. This basic operation leads to new algorithms for global self-reconfiguration planning. Each module has on-board processing, sensing, communication, and power.

We have implemented distributed rule-based locomotion and self-replication on a 12 module Crystal robot.

*Department of Computer Science, Dartmouth,rus@cs.dartmouth.edu.