39. Distributed and Cellular Robots

Zack Butler, Alfred Rizzi

This chapter is organized according to a number of broad classes of problem where modular robotic systems can prove beneficial. For each such problem, the benefits of modularity are described, along with the ways that particular systems or proposed systems have explored those benefits. In particular, we discuss locomotion in Sect. 39.1, manipulation in Sect. 39.2, modular robot geometry in Sect. 39.3, and robust systems in Sect. 39.4. The systems under consideration in general have some level of independent computation on each module, and this discussion will focus on systems in which modules maintain some sort of kinematic constraint between them during operation. Compared to the types of multirobot teams described in Chap. 40, the systems of interest here are generally much more tightly coupled, both physically and conceptually. That is, we are primarily concerned with systems which, though they have many processors and independent actuators, have a single goal or small set of goals which can only be achieved collectively, rather than a set of goals which can be apportioned to single (or a small number of) robots within the team.

For many families of tasks in robotics, including manipulation and locomotion, different instances of the task can be best solved by robots with different kinematics. For example, both legged robots and wheeled robots can be effective for locomotion, but each type is better suited to a particular type of environment (rough or generally smooth). Likewise, manipulator arms with different geometry can achieve good manipulability over different workspaces. The use of modularity in robots, whether by developing physically interchangeable parts for a single robot or creating completely independent robots that can join together as the situation demands, can make it possible for a single overall robotic system to achieve a variety of kinematic configurations that a fixed architecture can not.

39.1 Modularity for Locomotion

Modular robots can achieve a very wide variety of forms for locomotion over (or even through) rough terrain. A single robot system may be able to surmount obstacles many times the size of the modules itself, while also being able to navigate tunnels just larger than a single module. Some systems have been designed primarily

for multimodal locomotion, while other more general-purpose machines can also perform in this way, albeit less efficiently.

39.1.1 Self-Reconfigurable Robot Locomotion

Self-reconfigurable robots are a potentially very powerful type of system in which the modules move themselves with respect to each other to autonomously attain different configurations for different purposes. Systems constructed to date generally fall into two categories, *chain based* and *lattice based*, defined by how a collection of modules lies in space. Chain-based systems form essentially one-dimensional skeleton structures which can move in a continuous way, while lattice-based structures tend to have space-filling modules that form configurations that are relatively dense in two or three dimensions.

In chain-based systems, the modules generally connect as links in a chain, such that most modules connect to two other modules along an axis about which the module can rotate. Additional junction modules provide for more complex structures, such as legged spiders in addition to wheels and snakes. Some examples of such systems are the PolyBot [39.1] and the configurable robot (CONRO) system [39.4]. There are two major challenges when building chain-based systems. First of all, control of reconfiguration presents an interesting kinematic problem – turning a snake into a loop requires coordination of all the joints to bring the two ends of the snake close enough to allow physical connection. This is similar to traditional inverse kinematics with computation distributed over the joints of the system.

Once a chain-based system has achieved a given configuration, in order to progress over the terrain, the modules must produce a successful periodic gait. A gait is usually specified similarly to legged systems, that is, each joint has a schedule of angle with respect to time. The challenge here is that the gait will depend on the configuration – snakes of different lengths may require different gaits, and legged constructions with a varying number of legs are also possible. The work of *Zhang* et al. [39.5] on the Polybot system describes a set of scalable configuration types (loops, legs, snakes) and automated transitions between them as well as automatic gait generators. This work uses phase automata to generate the periodic angle for each joint, but specified in a way that can be easily recomputed as the configuration changes.

Modules in lattice-based systems fill space much like building blocks. This allows these robots to construct a much wider variety of shapes, and as such they may be able to surmount larger obstacles, but at the cost of less efficient locomotion. For example, as a wheeled or legged construct, a chain-based system cannot climb obstacles significantly higher than traditional wheeled or legged robots can, whereas a lattice-based robot can make scaffolding-like shapes limited only by the number of modules available, with modules then climbing the scaffolding and reforming a group on the opposite side of the obstacle. Here the hardware specifics become very important, as these systems perform locomotion by reconfiguring, that is, the modules will move over the remainder of the structure one at a time for the group to move forward. Figure 39.2 shows an example of lattice-based modules that actuate through compression and expansion climbing stairs using a hard-coded motion sequence.

Locomotion via shape change was formalized in a cellular-automata-like framework in [39.6] – in this work, each module independently runs a set of geometric rules and depending on the state of its neighborhood

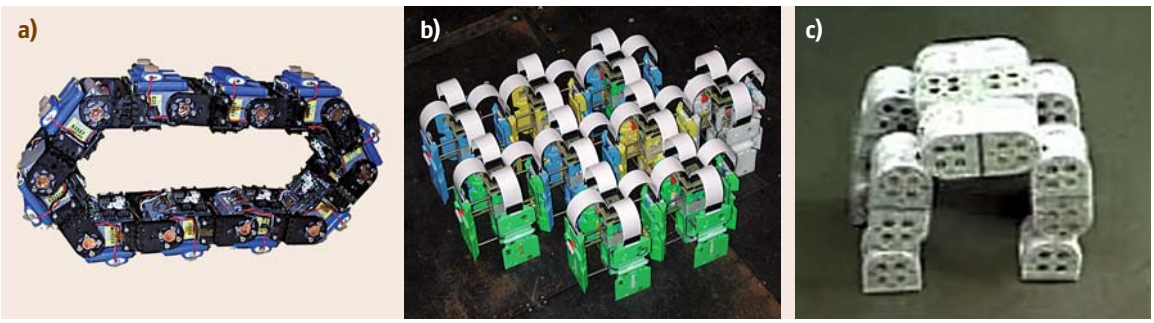

Fig. 39.1a–c Examples of self-reconfigurable robots: **(a)** the Polybot, a chain-based system [39.1]; **(b)** the Crystal, a two-dimensional lattice-based system [39.2]; **(c)** MTRAN [39.3], which can take on the characteristics of either type

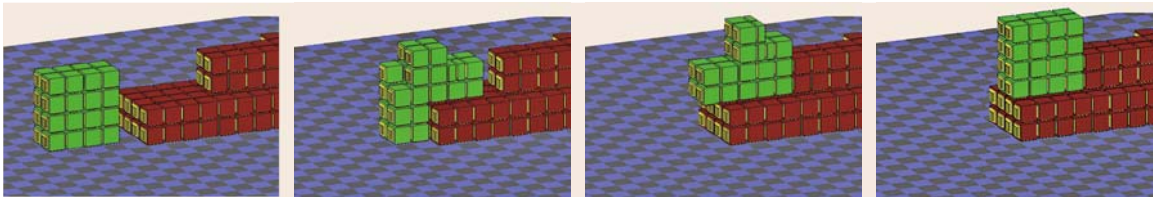

Fig. 39.2 A self-reconfigurable robot made of compressible modules climbing stairs that are two modules tall

(where other modules and obstacles are located), may move in a given direction. This technique was shown to ensure progress and avoid deadlock for straight-line motion over uneven terrain, traversing cliffs many modules high as well as narrow tunnels, and turning on flat ground, but is not trivially generalizable to more complex operations.

Lattice-based robots also have a wider variety of module types. Some systems are purely planar (e.g., [39.9, 10]) while others are three dimensional (e.g., [39.11])). They can be simple cubes or more complex shapes, though in general a single module is incapable of significant motion in isolation, instead requiring a substrate of other modules to connect to and walk across. Connections between modules can be mechanical [39.12] or magnetic [39.9]. The work of [39.6] proposed a generic model of a cubic module that can translate over other modules and make convex transitions, and showed that several different hardware systems can implement this model (though some require multiple hardware modules, termed metamodules, to implement a single generic module).

Recently this divide has been bridged with hardware than can operate in both chain-based and lattice-based modes. Modular transformer (MTRAN) and its successors [39.13] consist of 2-DOF modules, with each module containing two semicylindrical parts that rotate about a link between them. Each half of the module can make three connections to other modules through magnetic connectors that require energy only to disconnect. This robot can also operate completely untethered, including connecting and disconnecting, to effect true self-reconfiguration. This system has demonstrated self-reconfiguration in a lattice-based mode (where the discrete nature of the lattice makes the kinematics and therefore the docking problem much simpler) followed by conversion to a four-legged walker which uses the joints of the modules in a continuous fashion. The Superbot [39.14] is another recently developed system which promises to be at least as functional as the MTRAN due to an additional degree of freedom per module, however at present the ability to physically connect and disconnect autonomously is still under development.

In all of these systems, constructing functional and robust hardware is a critical challenge. The primary con-

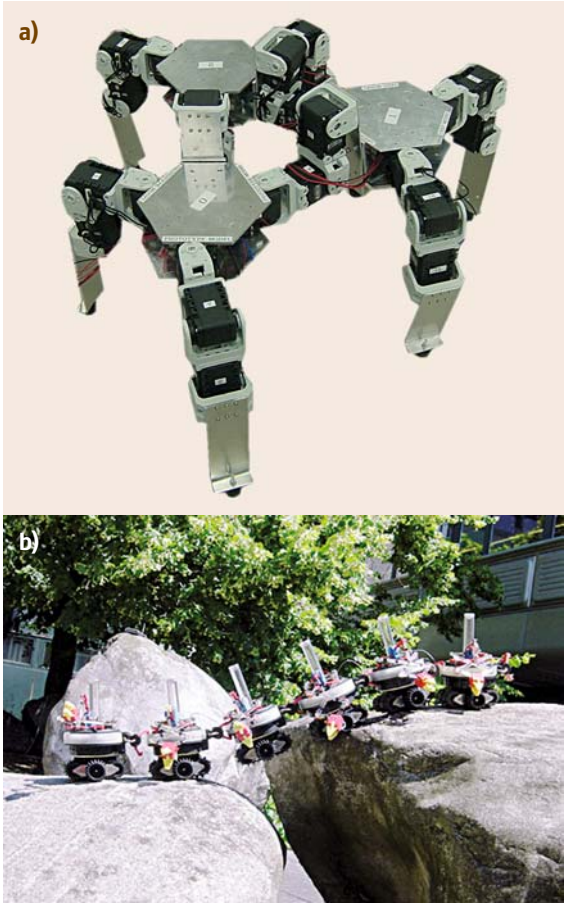

Fig. 39.3a,b Examples of independently capable mobile robots which can join together to perform more difficult tasks. **(a)** Three modules of the Ashigaru system [39.7] combined into a six-legged walker. **(b)** Swarmbots [39.8] connected together traversing a gap larger than a module in width

sideration is that of the connection mechanism – these will require fairly rigid connection, often including multiple electrical connections for communication and/or power transfer, while also allowing for physical connection in the face of position error. The position error can either be handled through sensing, such as the multiple beacons used in the CONRO modules, or through mechanical means, such as the grippers used in the Molecule robot. Connectors are especially critical for lattice-based systems, which must perform connection and disconnection to effect any significant geometric change.

39.1.2 Physically Cooperative Mobile Robots

In other situations, the locomotion requirements may be largely straightforward, with only certain specific areas requiring unusual abilities. In these cases, it may be more appropriate to use more traditional mobile robots as modules that can travel efficiently on their own but can also make rigid or semirigid connection to achieve greater capability.

One example of this is the Gunryu system proposed by *Hirose* et al. [39.15]. In this system, each module is a stand-alone tracked mobile robot approximately 0.5 m in length with a two-joint arm mounted on it. Each module can autonomously navigate moderately difficult terrain. Upon encountering very steep terrain or a hole in the ground larger than one module, the module can use its manipulator arm to hold onto another module. The authors were able to demonstrate that two modules connected together could traverse steeper ground than a single module. Key to this ability is the strength of the manipulator; it must be sufficiently strong to hold the weight of another module. A more recent robot called Ashigaru uses modules that each have three legs and can execute simple crawling gaits alone. Modules can grab each other in a variety of ways, as each leg can act as a grabber. Demonstrations have included forming quadrupeds and hexapods (such as in Fig. 39.3a), climbing steps taller than single modules are able to, and even using one module as a simple manipulator while it is held by other modules. It is currently manually reconfigured, though plans include the ability for self-reconfiguration [39.7].

More extensive work in this line is being carried out by in the SWARM-BOTS project [39.16]. The modules in this system are significantly smaller (circular robots 120 mm in diameter) but have a similar gripper that can grab other modules to allow for cooperative locomotion. In this case, the gripper is a 2-DOF wrist rather than a serial manipulator, which allows for greater rigidity. These modules have demonstrated the ability to cross a gap greater than one module diameter wide, as in Fig. 39.3b [39.8]. Each module also has a short serial manipulator to perform a wider variety of manipulation tasks, and the overall system has also demonstrated simple forms of cooperative manipulation.

39.2 Modularity for Manipulation

Modularity can also be a great advantage when manipulating a wide variety of payloads. In particular, the underlying kinematics of a manipulation system may be changed rapidly (either online or offline, depending on the system) so that the workspace and manipulability are most appropriate for a given task.

39.2.1 Independent Manipulators

One common task for which a group of robots can be beneficial is that of pushing or carrying a large object. If the object is significantly larger than a single robot can handle, a group is the only way to move it through the environment. When implementing such a system, the key issues include the frequency of data transfer and the effective sharing of the load. In order for the robots to maintain control over the object, the forces inherent must be shared so that no single robot is overwhelmed. *Khatib* et al. proposed the virtual linkage as an solution to this problem [39.17].

Force control is less of a concern in a quasistatic pushing problem – in this case, the challenge is in producing the appropriate trajectory for the pushed object given the limited range of forces that can be applied by each pusher. Synchronization is still important in such systems, however. *Mataric* et al. [39.18] demonstrated cooperative pushing with two legged robots moving a box that neither could move alone, and showed that in general, without synchronization, one would move faster than the other and misorient the box or move past the end of the box. In this work, the synchronization is done through explicit communication and time-slicing of control, that is, each robot has a time in which it can move, after which it passes control (and sensory information) to its colleague. This explicit synchronization is not always necessary, however – if the robots have suffi-

cient sensing and understanding of the mechanics of the pushing task, they can implicitly cooperate through the pushing task itself. That is, as the pushed object moves, each robot can infer the state of the overall system and choose its actions and role appropriately. This idea of information in the task mechanics was developed by *Donald* et al. [39.19], where they provide a set of transformations for information channels between the virtual and the physical.

Cooperative manipulation can also be achieved by modules that are not necessarily mobile. Some manufacturing-oriented systems have been proposed in which the manipulation modules can be much simpler than otherwise would be required because many modules can work together to perform the required tasks. One such system is the virtual vehicle of *Luntz* et al. [39.20]. In this system, each module is simply a wheel which can be rotated. By assembling a large number of them in a bounded region, a programmable conveyor is created. Multiple objects can be handled simultaneously at different points in the system, and can be handled differently (e.g., one being stopped and rotated in place while another is moved past). Again, synchronization is critical here – the motion of an object is determined by the total force and torque acting on it, which in turn is determined not only by the velocity and angle of each wheel under it, but also the relative location of each wheel. In order for the object to maintain the correct trajectory, all wheels must inject the appropriate force for their location, which can change as the object moves. One way to handle this complexity is to simplify the task such that the forces are not time dependent, for example, simply operating as a conveyor belt in which all wheels move the same direction, or having two portions of the system push toward each other to move objects to a given location.

Another system in which the use of multiple modules allows for the use of simpler robots is the minifactory of *Hollis* et al. [39.21]. Here, traditional 4-DOF assembly operations are performed by two robots, each with two DOF. One robot is an overhead two-axis (Z–Θ) manipulator and the other an X–Y planar motor. The simplification of the robots, in particular the elimination of a serial chain in the overhead manipulator, allows for greatly increased precision and the performance of micron-scale automated assembly. Here, however, the goal is to perform force-guided assembly, requiring the closing of a 1 kHz control loop between the two independent robots.

39.2.2 Reconfigurable Manipulators

A very different type of modularity for manipulation is that for a more or less traditional robot arm that can be easily rebuilt or reconfigured to achieve different kinematics. If the arm is constructed as a sequence of identical modular components (joints), these can then be attached to provide as many degrees of freedom as needed, with the appropriate workspace. The reconfigurable modular manipulator system (RMMS) was designed with these aims in mind [39.22]. The challenge here is to calculate the inverse kinematics and the dynamics automatically as the system changes, and in fact the RMMS does provide this functionality, though these systems usually have centralized computation with at most local joint controllers. This type of simplicity through modularity can be taken to a more extreme point, in which each module is a linkage with binary actuation, that is, only two possible states. Combining many of these binary modules with different geometry in different configurations creates robots with different workspaces from the same selection of components [39.23]. Here the kinematics and inverse kinematics are discrete, though the inverse kinematics are not closed form since the workspace is a set of discrete points in space. However, the control scheme is much simpler, since little or no feedback is required to direct the endpoint to a given location once the desired linkage positions are determined.

One of the earliest systems described in the field of self-reconfiguration is the cellular robot (CEBOT) of *Fukuda* et al. [39.24]. This system falls between the extremes of pure self-reconfiguring modules and mobile robots, as it does include modules which are independently mobile, but also includes nonmobile modules and expects that several modules will be used for most tasks. In one task described for CEBOT, individual modules can move through a small opening into a tank, and then reconfigure into a larger manipulator to perform operations across the entire tank.

39.3 Modularity for Geometric Reconfiguration of Robot Systems

The use of modularity allows for systems of many independent components to be reconfigured quickly between task operations with a minimum of effort. This is useful in manufacturing systems, where the modules may

be capable of stand-alone operation, as well as self-reconfiguring robots, where sensors can be repositioned as the robot performs its tasks. In both cases, there are several issues to consider: ease of reconfiguration, determination of the current configuration, choosing good configurations for a particular task, and the design of the underlying task-performance algorithms for use with arbitrary configurations.

39.3.1 Manually Reconfigured Systems

Traditional manufacturing can be performed in workcells, in which usually a single robot has access to several stations and performs a series of assembly operations. These workcells are designed to be fairly modular from a component point of view, swapping different portions of the product in and out as the manufacturing task changes. Several workcells can then be laid out along a conveyor or other line mechanism to create an assembly line. While they may be very similar in construction, they tend to be fairly independent in operation. Scheduling in the traditional sense may be important, but real-time interactions are less likely to be.

In the Minifactory project, on the other hand, the entire system including the conveyances is part of a single modular system. All modules conform to a common interface, both in hardware and software [39.21]. This allows rapid reconfiguration, with each robot able to be placed in the system with clamps and simple connectors. The robots share a common high-level language and a centralized simulation environment, so that tasks can be programmed into the system ahead of time. When the modules are assembled, the robots that ferry the products through the system can also be used to discover the exact locations of all of the components as placed. The task algorithms, written under a common framework, are then automatically updated with the exact locations to be visited. This allows for a smooth and efficient transition from a simulated assembly line to an operating set of hardware.

39.3.2 Shape Generation via Self-Reconfiguring Systems

Self-reconfigurable robots, as the name implies, can autonomously change their overall shape to fit the task at hand. In addition to using this capability to perform a variety of locomotion tasks, lattice-based systems in particular have the capability to make nearly arbitrary three-dimensional shapes. This may be useful for generating different sizes and shapes of payload carriers, manipulators, or other devices. In more recent research, systems of heterogeneous modules, where the capabilities of each module are not necessarily the same, have again received more attention [39.25]. These may be useful for positioning cameras or other specialized sensors at varying positions during the completion of a task even as the shape of the system remains the same. Figure 39.4 shows a simulation of a heterogeneous reconfiguration from a chair to a table – in this technique, the system first achieves the desired shape without regard to module type (denoted by module color), after which the modules physically sort themselves into the correct locations.

To perform self-reconfiguration, the modules in a system need to cooperate to plan their motions so that they can form the desired configuration. In common practice, the modules run without a central controller, but the global desired shape in some specification is given to all modules, at which point each module makes local decisions about if and where to move. To perform the reconfiguration, primarily local algorithms as well as more global planners have been developed.

Local shape-formation techniques tend to be connection based, that is, the desired shape is specified by

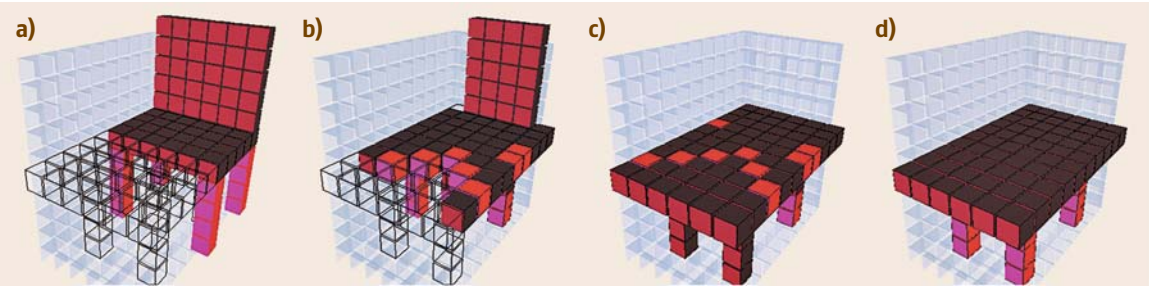

Fig. 39.4a–d Heterogeneous reconfiguration of a chair-shaped system to a table shape. **(a)** Initial configuration. **(b)** Modules moving in parallel to achieve the correct configuration. **(c)** Final system shape. **(d)** Final module locations based on heterogeneous module types

listing the connections to be made between modules. Each module is then responsible for ensuring it has the appropriate set of modules connected to it. *Tomita* et al. first proposed a shape generation scheme of this nature for two-dimensional systems [39.26], in which a single seed module collected modules around in successive layers, with each layer collecting appropriate modules as specified by the global design. Along a similar vein, work by *Lipson* et al. [39.27] constructs shapes out of modules that do not contain traditional actuation, but rather exist in a moving fluid medium. The modules can only control when and where to allow other modules to connect to them to build the desired shape, and so a connection-based scheme is natural. Recent work by *Klavins* [39.28] avoids the need for a seed module, and uses reaction network theory to allow modules that move stochastically (e.g., due to external forces) to intelligently detach and reattach and achieve many copies of the desired shape in an efficient and reliable way.

Global techniques tend to look more like traditional path planners – the modules discover local mismatches between the current configuration and the desired configuration, and paths are planned in a distributed fashion for modules to move into empty locations. One interesting feature is that the available paths are over the surface of other modules, so there is computation available wherever the search needs to take place, and distributed planning is quite natural. In addition, the paths are naturally discrete, so no artificial constraints need to be applied to perform the planning.

Here there are two main concerns: first, a matching problem must be solved, to ensure that each goal location is the destination of only one module and that each location is accounted for; and second, deadlock must be avoided and traversability maintained if and when multiple plans are being generated and executed within the system. The first can be solved by having each goal location spoken for by a particular module, and having that module initiate a path search for a module that can fill the location. In addition, these techniques also tend to build the goal shape one layer at a time, starting from the current shape and extending outward. Each individual path search must be filled by only one module, either by executing a depth-first search that terminates when a capable module is found (which is the approach taken in [39.29]) or having some other way to choose a module if multiple modules attempt to fill the requested location. Modules also must know to attempt motion only if they are not required at their current position and can move without disconnecting the overall structure. *MeltGrow* [39.30] and its heterogeneous successor, MeltSortGrow, handle this latter problem by moving all the modules to an intermediate shape (a line) from which the end of the line can be locally detected and only this module will move to a goal location. However, modules can detect whether they can will disconnect the structure through a depth-first search from their neighbors, and so this melting step is unnecessary. During motion, care must be taken to ensure that paths are still valid as other modules begin to move. This can be done through serialization or through a localized traffic control scheme [39.29], in which modules communicate to take turns passing through a particular area. Shape reconfiguration algorithms can be shown to require $O(n^2)$ module motions for n modules [39.31], but usually only $O(n)$ time if the motions are executed in parallel.

If the systems are heterogeneous, that is, the modules are of different types, modules must also determine which type is required at a given location, and only modules of that type can respond to the particular request. *Fitch* et al. [39.25] propose a planning system for identically sized but arbitrarily typed modules in which the correct shape is first achieved regardless of type, after which modules swap location. This algorithm still only requires $O(n^2)$ motion under most circumstances and can be efficiently executed in a distributed way.

39.3.3 Configuration Optimization

The determination of an appropriate configuration is another key issue in both manually reconfigured and self-reconfiguring systems, but has been little discussed in the literature. One common approach is an empirical one – to simulate a particular task or set of tasks (e.g., an assembly process) under different configurations and compare efficiency in each case. For example, in describing the RMMS system, [39.22] mentions that a simulator is available for determining the best configuration for a particular manipulation task. Similar work was performed for the CEBOT system, in which the location of the manipulator as well as the required number of modules and their configuration was optimized [39.32]. There is work with binary manipulators to find an optimal configuration for a given set of points to be reached, in which Lagrange multipliers are used to minimize the change from a base structure to a structure that can achieve the desired locations [39.23].

In systems such as self-reconfiguring robots, where hundreds of modules may be placed in fairly arbitrary ways, there are vast numbers of configurations to be considered, and the tasks themselves can be more complex. An evolutionary approach to optimizing con-

figurations of modular (though not self-reconfiguring) robots is described in Chap. 61. More general configuration optimization approaches from multirobot teams may be applicable for tasks such as surveillance, though they would need to be modified to handle the added capabilities of self-reconfiguring systems.

39.3.4 Self-Replicating Systems

One type of system that obviates the question of configuration determination is the self-replicating modular robot. In these systems, one group of modules builds a copy from a supply of modules. While the theory of self-replication was established by *von Neumann* [39.33], engineering such a robotic system is a significant challenge. This capability has been demonstrated in restricted form by *Suthakorn* et al. [39.34], in which the robots are constructed of four fairly complex modules, and one robot can construct a second, and by *Zykov* et al. [39.35] in a system of more traditional lattice-based self-reconfiguring modules.

39.4 Modularity for Robustness

Finally, the use of a large number of simple modules to complete a task allows for robustness in the event of failure. If the modules are completely homogeneous, then they can simply be swapped out and replaced. If the system is engineered with sufficient redundancy, modules may even be done without as they fail. This process of self-repair may be possible to execute in an autonomous fashion, as has been proposed for some self-reconfigurable systems [39.26, 36]. In these cases, the failed module is ejected from the system either by selective disassembly to get the failed module to the surface, or by a pushing gait that moves the module. The DRAGON connector [39.37] can always be detached from either side in the event of module failure, so that the system can continue with minimal disruption. However, despite the presumed importance of this self-repair capability of modular systems, this has not been a major focus of research.

On the other hand, systems which are manually reconfigured may not have the option to evict a nonfunctional module. Instead, they may be built with inherent redundancy so that nonfunctional modules can simply be handled, either actively or passively. For example, in the virtual vehicle, the failure of a single unit may not affect the overall performance if the objects in transport are large relative to the actuators. Likewise, redundant manipulators such as the RMMS [39.22] can continue to function if a single joint seizes, though this must be actively detected so that their changed kinematics and dynamics can be taken into account.

39.5 Conclusions and Further Reading

The benefits of modularity can be seen in many different task domains, whether the systems change their configuration on their own or require assistance from humans. The modularity can be used to simply create larger machines, for example, for more capable locomotion, or more complex machines that still retain a notion of simplicity at the module level. At present, hardware systems have largely remained in the research realm, while algorithmic progress has been notably more advanced in several areas, such as self-reconfiguration. It is hoped that the costs associated with modular robot development can be reduced as a byproduct of mass production, enabling more deployments that take advantage of the robustness and versatility inherent to such systems.

Interested readers may find more details of some self-reconfiguring robots in a survey paper [39.38], and workshops in this area are often held in conjunction with international conferences, such as the 2006 Robotics: Science and Systems conference, and the 2007 International Conference on Intelligent Robots and Systems. Distributed robotic systems in general are the subject of the biennial distributed autonomous robotics systems (DARS) conference, including many papers relevant to this topic.

References

39.1 M. Yim, D. Duff, K. Roufas: PolyBot: A modular reconfigurable robot, Proc. of IEEE ICRA (2000) pp. 514–520

39.2 Z. Butler, D. Rus: Distributed motion planning for modular robots with unit-compressible modules, Int. J. Robot. Res. **22**(9), 699–716 (2003)

39.3 S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura, S. Kokaji: Hardware design of modular robotic system, Proc. of IROS (2000) pp. 2210–2217

39.4 A. Castano, W.-M. Shen, P. Will: CONRO: Towards deployable robots with inter-robots metamorphic capabilities, Auton. Robot. **8**(3), 309–324 (2000)

39.5 Y. Zhang, M. Yim, C. Eldershaw, D. Duff, K. Roufas: Scalable and reconfigurable configurations and locomotion gaits for chain-type modular reconfigurable robots, Proceedings of the International Symposium on Computational Intelligence in Robotics and Automation (2003) pp. 893–899

39.6 Z. Butler, K. Kotay, D. Rus, K. Tomita: Generic decentralized locomotion control for lattice-based self-reconfigurable robots, Int. J. Robot. Res. **23**(9), 919–938 (2004)

39.7 M. Ohira, R. Chatterjee, T. Kamegawa, F. Matsuno: Development of three-legged modular robots and demonstration of collaborative task execution, Proc. of IEEE Int'l. Conf. on Robotics and Automation (2007) pp. 3895–3900

39.8 V. Trianni, S. Nolfi, M. Dorigo: Cooperative hole avoidance in a swarm-bot, Robot. Auton. Syst. **54**(2), 97–103 (2006)

39.9 S. Murata, H. Kurokawa, S. Kokaji: Self-assembling machine, Proc. of IEEE ICRA (1994) pp. 442–448

39.10 D. Rus, M. Vona: A physical implementation of the crystalline robot, Proc. of IEEE ICRA (2000)

39.11 C. Ünsal, P. Khosla: Mechatronic design of a modular self-reconfiguring robotic system, Proc. of IEEE ICRA (2000) pp. 1742–1747

39.12 K. Kotay, D. Rus, M. Vona, C. McGray: The self-reconfiguring robotic molecule: design and control algorithms, Proc. of the Workshop on Algorithmic Foundations of Robotics (1998)

39.13 S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, S. Kokaji: M-TRAN: Self-reconfigurable modular robotic system, IEEE/ASME Trans. Mechatron. **7**(4), 431–441 (2002)

39.14 W.-M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, J. Venkatesh: Multimode locomotion for reconfigurable robots, Auton. Robot. **20**(2), 165–177 (2006)

39.15 S. Hirose, T. Shiratsu, F.E. Fukushima: A proposal for cooperative robot "gunryu" composed of autonomous segments, Proceedings of the International Conference on Intelligent Robots and Systems (1994) pp. 1532–1538

39.16 F. Mondada, L.M. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneubourg, M. Dorigo: The cooperation of swarm-bots, IEEE Robot. Autom. Mag. **12**(2), 21–28 (2005)

39.17 O. Khatib, K. Yokoi, K. Chang, D. Ruspini, R. Holmberg, A. Casal: Coordination and decentralized cooperation of multiple mobile manipulators, J. Robot. Syst. **13**(11), 755–764 (1996)

39.18 M. Mataric, M. Nilsson, K. Simsarian: Cooperative multi-robot box pushing, Proceedings of the International Conference on Robotics and Automation (1995) pp. 556–561

39.19 B. Donald, J. Jennings, D. Rus: Information invariants for distributed manipulation, Int. J. Robot. Res. **16**(5), 673–702 (1997)

39.20 J. Luntz, W. Messner, H. Choset: Closed-loop operation of actuator arrays, Proc. of IEEE Int'l. Conf. on Robotics and Automation (2000) pp. 3666–3671

39.21 A.A. Rizzi, J. Gowdy, R.L. Hollis: Agile assembly architecture: An agent-based approach to modular precision assembly systems, Proc. of IEEE Int'l. Conf. on Robotics and Automation (1997) pp. 1511–1516

39.22 C. Paredis, H.B. Brown, P. Khosla: A rapidly deployable manipulator system, Proceedings of the International Conference on Robotics and Automation (1996) pp. 1434–1439

39.23 G.S. Chirikjian: A binary paradigm for robotic manipulators, Proc. of IEEE Int'l. Conf. on Robotics and Automation (1994) pp. 3063–3069

39.24 T. Fukuda, T. Ueyama: *Cellular Robotics and Microrobotic Systems* (World Scientific, Singapore 1994)

39.25 R. Fitch, Z. Butler, D. Rus: In-place distributed heterogeneous reconfiguration planning, Proc. of Distributed Autonomous Robotic Systems (2004)

39.26 K. Tomita, S. Murata, H. Kurokawa, E. Yoshida, S. Kokaji: Self-assembly and self-repair method for a distributed mechanical system, IEEE Trans. Robot. Autom. **15**(6), 1035–1045 (1999)

39.27 P. White, V. Zykov, J. Bongard, H. Lipson: Three dimensional stochastic reconfiguration of modular robots, Proceedings of Robotics: Science and Systems (Cambridge 2005)

39.28 E. Klavins: Tuning reaction networks for programmed self-organization, Proceedings of the Third Conference on the Foundations of Nanoscience (2006) pp. 34–37

39.29 S. Vassilvitskii, M. Yim, J. Suh: A complete, local and parallel reconfiguration algorithm for cube style modular robots, Proc. of IEEE ICRA (2002) pp. 117–122

39.30 M. Vona, D. Rus: Self-reconfiguration planning with compressible unit modules, Proc. of IEEE Int'l. Conf. on Robotics and Automation (1999) pp. 2513–2520

39.31 G. Chirikjian, A. Pamecha: Bounds for self-reconfiguration of metamorphic robos, Proceedings of the International Conference on Robotics and Automation (1996) pp. 1452–1457
39.32 T. Fukuda, S. Nakagawa, Y. Kawauchi, M. Buss: Structure decision method for self organizing robot based on cell-structure cebot, Proceedings of IEEE International Conference on Robotics and Automation (1989) pp. 695–700
39.33 J. von Neumann: *Theory of Self-Replicating Automata* (Univ. of Illinois Press, Chicago 1966)
39.34 J. Suthakorn, Y.T. Kwon, G. Chirikjian: An autonomous self-replicating robotic system, Proc. of the International Conference on Advanced Intelligent Mechatronics (2003) pp. 137–142
39.35 V. Zykov, E. Mytilinaios, B. Adams, H. Lipson: Self-reproducting machines, Nature **435**, 163–164 (2005)
39.36 R. Fitch, D. Rus, M. Vona: A basis for self-repair robots using self-reconfiguring crysta l modules, Intelligent Autonomous Systems 6 (2000)
39.37 M. Nilsson: Connectors for self-reconfiguring robots, Trans. Mechatron. **7**(4), 473–474 (2002)
39.38 D. Rus, Z. Butler, K. Kotay, M. Vona: Self-reconfiguring robots, Commun. ACM **45**(3), 39–45 (2002)