Modular Self-Reconfigurable Robots

Mark Yim, Paul White, Michael Park, Jimmy Sastra
School of Engineering and Applied Science, University of Pennsylvania, Philadelphia, USA

Article Outline

Glossary

Bonding mechanism A mechanism that allows modules to attach to other modules. Self-reconfigurable modules have the ability to selectively make and break attachments to other modules.

Configuration The connectivity arrangement of modules in a system which describes which modules is physically attached and adjacent to which.

Configuration recognition The process of automatically determining a modular robot's connectivity arrangement.

Decentralized control A control system in which the controller elements are not central in location (like the brain) but are distributed throughout the system with each component sub-system controlled by one or more controllers.

Enumeration algorithm A routine that counts and displays the number of unique, non-isomorphic configurations of a given modular robotic system.

Global bus Communication setup such that when one unit talks all other units can listen, as opposed to neighbor to neighbor communication in which communication occurs only between two units.

Isomorphic configurations Modular structures that have the same morphology but are arranged differently according to their module labels.

Morphology The form or structure of some entity, more specifically, the connectivity arrangement of modules in a system independent of module labels.

Reconfiguration algorithm A method that transforms a given robotic configuration to a desired configuration via a sequence of module detachments and reattachments.

Definition of the Subject

Modular self-reconfigurable (MSR) robots are robots composed of a large number of repeated modules that can rearrange their connectedness to form a large variety of structures. An MSR system can change its shape to suit the task, whether it is climbing through a hole, rolling like a hoop, or assembling a complex structure with many arms.

These systems have three promises:

Versatility The ability to reconfigure allows a robot to disassemble and/or reassemble itself to form morphologies that are well-suited for a variety of given tasks.

Robustness Since the system is composed of many repeated parts which can be rearranged during operation, faulty parts can be discarded and replaced with an identical module on-the-fly, leading to self-repair.

Low cost MSR systems can lower module costs since mass production of identical unit modules has an economic advantage that scales favorably. Also, a range of complex machines can be made from a set of modules saving the cost versus having multiple single-function machines for doing different tasks.

Introduction

Conceptually, the best known example of an MSR robot would be the fictional T1000 liquid-metal robot from the James Cameron film, *Terminator 2: Judgment Day*. In this movie, a robot made from a futuristic liquid-like metal, (possibly many million microscopic modules) can change its shape, copy forms, or reconstitute itself to carry out sinister aims.

Real robots that change their shape, made up of many identical modules have been created and are being studied by a wide variety of groups [33]. These robots are capable of more useful contributions to society than the T1000. They promise to be versatile, low cost, and robust. While these systems do not yet behave like liquid metal, systems on the order of 100 modules have been built and promise to be useful in search and rescue or space exploration.

The concept of modular self-reconfigurable robots can be traced back to the "quick change" end effecter and

automatic tool changers in computer-controlled machining centers in the 1970's. Here, special modules, each with a common connection mechanism, were automatically interchanged on the end of an electro-mechanical or robotic arm. The concept of applying a common connection mechanism to an entirely modular robot was introduced by Fukuda with the biologically-inspired CEllular roBOT (CEBOT) in the late 1980's [11]. Here each CEBOT module is 18 x 9 x 5 cm and weighs approximately 1.1 kg. These units have independent processors and motors, and can communicate with each other to approach, connect, and separate automatically.

In the early 1990's, modular reconfigurable robots were shown to have the ability to perform the task of locomotion. In 1994, Yim explored many statically stable locomotion gaits with Polypod. Polypod [28] is an MSR robot that is significantly lighter and smaller than CEBOT. A module by itself could not locomote, but through the collective behavior of the system of many modules it could move itself from place to place and achieve many different locomotion gaits [29] such as a slinky, caterpillar, or rolling track gait.

Through this work it became clear that controlling a system with a large number of modules is complex. Initial Polypod control used a gait control table to program simple gaits on a modular robot using prescribed motions. In addition to the complexity of coordinated control, the complexity of arbitrary configurations and the sequence of reconfigurations to attain those configurations quickly developed into an interesting computational problem.

Chirikjian and Murata developed lattice style configuration systems in [10,17]. As described in Sect. "Modular Self-Reconfigurable Robot Review", the lattice style robots have modules which sit on a lattice and make it easier to represent the configurations computationally. As a result this style of system quickly became popular among computational roboticists. This also presents the interesting issue of the tradeoffs between issues solved electro-mechanically versus computationally, which is developed further in Sect. "Mechanical/Electrical/Computational Interaction".

In the later 1990's Rus [14] and Shen [6], also developed hardware but their larger contributions came in the distributed programming aspects. This included seminal trends in developing provable distributed algorithms [4] and decentralized control based on local communication [23]. Two of the areas of research include configuration self-recognition and kinematic planning of the motions for rearrangement between configurations.

This paper is structured as follows: Sect. "Modular Self-Reconfigurable Robot Review" gives a classification scheme for MSR robots, potential applications, and an overview of robotic systems that are currently being developed. Section "Complexity in Robot Configurations" discusses issues regarding complexity in the configurations of MSR robots. In Sect. "Control Architectures", we present architectures used for control in MSR systems. In Sect. "Mechanical/Electrical/Computational Interaction", we discuss the interaction between mechanical, computer, and electrical disciplines within modular robots. Lastly, in Sect. "Future Directions" we present future directions in MSR research.

Modular Self-Reconfigurable Robot Review

Categories of MSR Systems

There are several ways of categorizing MSR robotic systems. One is based on the regularity of locations for attaching; lattice vs. chain vs. mobile, and another is based on the methods of moving between those locations; stochastic vs. deterministic.

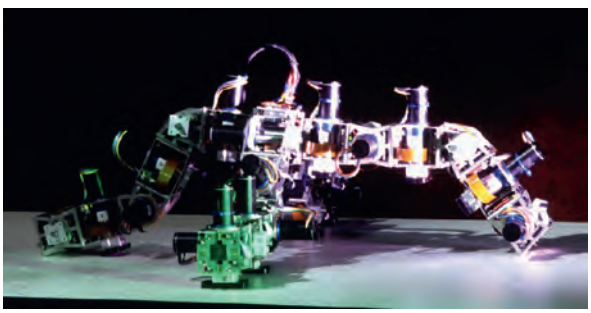
Lattice A lattice based MSR system has modules arranged nominally in a 2D or 3D grid structure. For this category, there are discrete positions that a given module can occupy. In contrast to chain-based architectures where modules are free to move in continuous space, the grid based structure of lattice systems generally simplifies the reconfiguration process. Kinematics and collision detection are comparatively simple for lattice systems. An example is shown in Fig. 1.

Chain A chain based MSR system consists of modules arranged in groups of connected serial chains, forming tree and loop structures. Since these modules are typically arranged in an arbitrary point in space, the coordination of a reconfiguration is complex. In particular, forward and inverse kinematics, motion planning, and collision detection are problems that do not scale well as the number of modules increases. An example is shown in Fig. 2

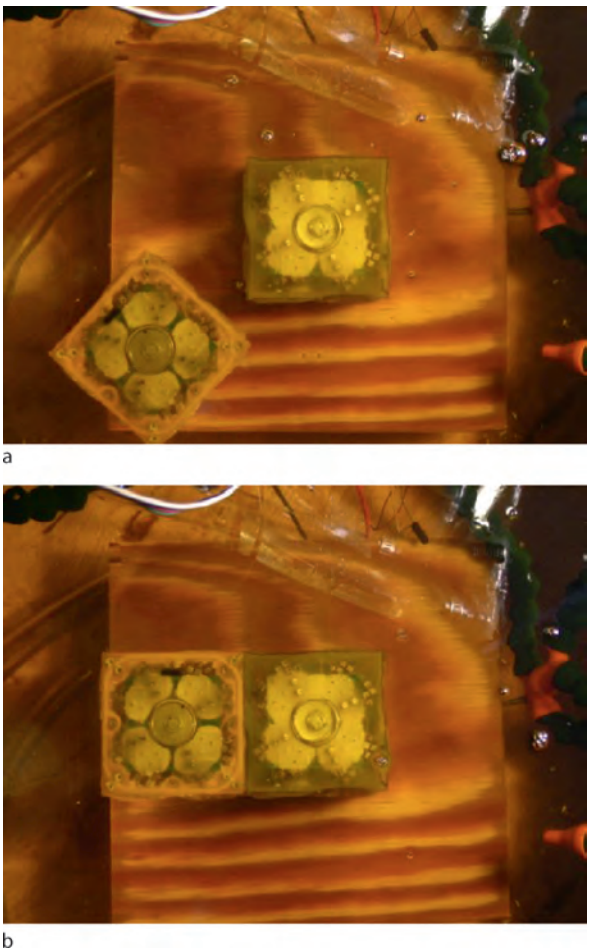
Mobile The mobile class of reconfiguration occurs with modules moving in the environment disconnected from other modules. When they attach, they can end up in chains or in a lattice. Examples of mobile reconfiguration devices include multiple wheeled robots that drive around and link together to form trains, modules which float in a liquid or outer space and dock with other modules.

Stochastic In a stochastic system, modules move in a 2D or 3D environment randomly and form structures by bonding to a substrate and/or other modules. Modules move in the environment in a passive state. Once a module

Modular Self-Reconfigurable Robots, Figure 1
Crystalline. The Crystalline system is a lattice style robot developed by Rus et al. [20] at Dartmouth University (then continued later at MIT) consists of modules that can expand and contract their shape in order to reconfigure and mobilize the robotic structure. Each module has three actuators: one rack-and-pinion device that allows all four sides to expand and effectively double the side length of the module and mechanical latches that allows the module to make and break bonds to its neighbors. Locomotion and shape metamorphosis was demonstrated experimentally both in simulation and with a physical implementation. The ability to self-repair a system with a malfunctioning module was demonstrated in simulation. The system was able to identify and relocate the damaged module. Both centralized and distributed planning algorithms were explored with Crystalline



Modular Self-Reconfigurable Robots, Figure 2
PolyBot, Yim et al. developed the PolyBot chain-type MSR system at Palo Alto Research Center (PARC, formerly Xerox PARC). Each 50mm cube shaped modules is equipped with a brushless flat motor and harmonic drive which provides a single rotational DOF. Sensors provide information about neighbor proximity and contact, orientation, joint position and force torque feedback. Two hermaphroditic (electrically and mechanically) faces of the module possess redundant spring contacts to transmit power and communication and an SMA actuated mechanical latch to bond to a neighbor module. PolyBot robotic systems have shown their versatility by demonstrating locomotion as a biped, as a snake, as a rolling tread, and by climbing stairs, poles, etc. The system has also demonstrated the ability to manipulate objects and self-reconfigure



Modular Self-Reconfigurable Robots, Figure 3
Stochastic 3D. A stochastic MSR robotic system has been demonstrated in both 2D and 3D by White et al. [26] at Cornell University. The 2D system consisted of planar, square-shaped modules with electromagnets on each face that allowed the modules to selectively bond and release other modules. The modules were shuffled about randomly on an oscillating air table. The modules do not have onboard power nor do they have the capability to influence their motion. One central module acts as a powered substrate to which other modules may attach to and build desired structures. The oscillating table causes the modules to move about randomly, and when two modules collide properly, they bond to one another via the electromagnets, determine if the new configuration is desired, and release from each other if the configuration is not desired. Stochastic reconfiguration was also demonstrated in 3D where cube-shaped modules floated about in an agitated oil environment. The first generation of 3D modules (shown above) used electromagnets to provide the bonding force; a second generation used the force caused by fluid that flowed through the faces of the modules

contacts the substrate or another module, it makes a decision about whether it will bond to the structure or reject a bond. The time that it takes for the system to reach a desired configuration is probabilistically bounded. The reliance on environmental forces allows the mechanical actuation to be simplified as only bonding actuation is required internal to the module. An example is shown in Fig. 3.

Deterministic In deterministic MSR systems, modules move or are manipulated directly from one position to another in the lattice or chain. The positions of each module in the system are known at all times. The amount of time it takes for a system to change from one configuration to another is determined. A module's reconfiguration mechanism requires a control structure that allows it to coordinate and perform reconfiguration sequences with its neighbors.

There are a growing number of existing physical systems that researchers are developing self-reconfigurable robots. One indication that this number is getting large is the development of a robot whose name is YaMoR (Yet another Modular Robot) [16]. Table 1 lists many of the other instantiated modular robot systems. In addition to the name, class, and author, the table lists DOF. This describes the number of actuated degrees of freedom for module motion (e. g. not latch degrees of freedom) as well as whether the system motion is planar (2D) or can move out of the plane (3D). The year is the estimated first public disclosure.

Applications

Compared with fixed morphology robots, MSR robots are flexible in that they can adapt to a wide range of tasks and environments. However, this flexibility may compromise performance or cost. Fixed morphology systems can be optimized for a particular known task, therefore, MSR robotic systems are particularly well-suited for tasks where the operating conditions and ability requirements are not known or not well specified a priori. The following set of application examples illustrate some areas that would benefit from the development of a mature MSR system.

Space The exploration of space presents numerous challenges, including an unpredictable environment and significant limitations on the mass and volume of equipment used to study that environment. Since one set of modules can be reconfigured to perform many tasks, MSR robots can solve both the unexpected challenges while occupy little space and weight as compared to multiple devices. Graceful degradation due to failure is particularly important for robots operating in space – a component malfunction can potentially lead to mission failure. The redundant nature of MSR systems gives them the ability to discard failed modules. Modules can also be packaged in a convenient way so as to meet the volume constraints of spacecraft. Once on site, modules can be used to build structures, navigate across terrain, perform scientific studies, etc.

Search and Rescue Disaster areas such as those around collapsed buildings or other structures present another type of highly unstructured unpredictable environment where the use of an MSR robot could be beneficial. For example, the MSR system could take the form of a snake which can more easily squeeze through small void spaces to find victims. Once found, the robot could emit a locator beacon and take the form of a shelter to protect the victim until rescued.

Bucket of Stuff The term "Bucket of Stuff" is futuristic idea coined by David Duff at the Palo Alto Research Center [33]. The system would be a consumer product comprised of a container of reconfigurable modules that would reconfigure to accomplish arbitrary household tasks. This application can be seen as the most general practical goal of MSR robotics: a system that can adapt to any task in real time. A bucket of MSR modules could be used to form the desired configuration for the end user such as cleaning gutters to folding laundry.

Current Modular Robot Systems

In the previous section we presented historical examples of MSR robotic systems. In the following section we present MSR robotic systems under experimentation and development at time of this publication.

Chain The CKBot system is a reconfigurable robotic system developed by Yim et al. at the University of Pennsylvania. The CKBot system shown in Fig. 4, is a chain based system building on earlier PolyBot work at the Palo Alto Research Center. These modules utilize a servo to rotate one portion of the module with respect to the other. In addition to statically stable locomotion gaits, Sastra et al. [22] have demonstrated a dynamic rolling gait for the CKBot system that has proven to be the fastest battery powered modular reconfigurable robot system. Global inter-module communication through CANbus as well as local neighbor-to-neighbor communication is incorporated on the modules. This system has also been used in some ini-

Modular Self-Reconfigurable Robots, Table 1
List of self-reconfigurable modular systems

System name	Class	DOF	Primary author	Affiliation	Year
CEBOT	mobile	various	Fukuda et al.	Nagoya	1988
Polypod	chain	2 3D	Yim	Stanford	1993
Metamorphic	lattice	3 2D	Chirikjian	JHU	1993
Fracta	lattice	3 2D	Murata	MEL	1994
Tetrobot	chain	1 3D	Hamlin et al.	RPI	1996
3D Fracta	lattice	6 3D	Murata et al.	MEL	1998
Molecule	lattice	4 3D	Kotay and Rus	Dartmouth	1998
CONRO	chain	2 3D	Will and Shen	USC/ISI	1998
PolyBot	chain	1 3D	Yim et.al	PARC	1998
TeleCube	lattice	6 3D	Suh et.al	PARC	1998
Vertical	lattice	2D	Hosakawa et al.	Riken	1998
Crystal	lattice	4 2D	Vona and Rus	Dartmouth	1999
I-Cube	lattice	3D	Unsal	CMU	1999
Pneumatic	lattice	2D	Inoue et.al.	TiTech	2002
Uni Rover	mobile	2 2D	Hirose et al.	TiTech	2002
MTRAN II	hybrid	2 3D	Murata et al.	AIST	2002
Atron	lattice	1 3D	Stoy et al.	U.S Denmark	2003
Swarm-bot	mobile	3 2D	Mondada et al.	EPFL	2003
Stochastic 2D	stochastic	0 2D	White et al.	Cornell U.	2004
Superbot	hybrid	3 3D	Shen et al.	USC/ISI	2005
Stochastic 3D	stochastic	0 3D	White et al.	Cornell U.	2005
Catom	lattice	0 2D	Goldstein et al.	CMU	2005
Prog. parts	stochastic	0 2D	Klavins	U. Washington	2005
Molecube	chain	1 3D	Zykov et al.	Cornell U.	2005
YaMoR	chain	1 2D	Ijspeert et al.	EPFL	2005
Miche	lattice	0 3D	Rus et al.	MIT	2006

tial experiments in self-repair in with experiments in self-reassembly after explosion described further in Sect. "Control Architectures".

Lattice The ATRON system, shown in Fig. 5, developed by Stoy et al. [13] at the University of Southern Denmark looks to combine the reliability of reconfiguration provided by a lattice based module architecture while maintaining some of the flexibility of motion of a chain based system. Modules can distribute power via their bonding mechanisms and use a power management system for voltage regulation and battery charge maintenance. A module consists of two hemispheres where one can rotate continuously relative to the other. The bonding mechanism is extremely robust; each module has 4 female metal bars and 4 metal clasps that can be actuated to grab hold of a neighbor's bar. Reconfiguration is performed by having one module grab another and then rotate some multiple of 90 degrees to another position in the lattice structure. The ATRON system has been used to explore the value of using clusters of multiple modules to increase the manipulation, reconfiguration and locomotion abilities of the system.

The Miche system developed by Rus et al. [12] at MIT has demonstrated the ability to form desired configurations from a collection of modules. In order to self-assemble, a cluster of modules disassembles by rejecting modules that are not part of the goal configuration. Each face of the cube module has a switchable magnet and a communication interface. After the user defines the desired shape of the robotic system through the interface, a distributed algorithm determines which modules should be rejected from the system. These modules simply let go of the structure and fall due to gravity. Like many of the stochastic systems, the hardware here de-emphasizes the actuation requirements easing the ability to scale up the numbers and scale down the size.

Modular Self-Reconfigurable Robots, Figure 4
CKbot module cluster. 4 CKbot modules and one CKbot camera module are joined together in a cluster. Several clusters can join together attaching magnetically

Hybrid

The M-TRAN system developed by Murata et al. [18] at AIST/Tokyo Institute of Technology combines the positive capabilities of chain and lattice based systems to implement a highly maneuverable and reconfigurable system, Fig. 6. A module consists of one passive and one active cube that can pivot about the link that connects them and can form chains for performing tasks. However during reconfiguration, each of a module's two cubes can occupy a discrete set of positions in space when attempting to align with another module and bond for reconfiguration as in a lattice system. The current generation of M-TRAN (III) modules utilizes a mechanical latch as a bonding mechanism which is considerably faster, stronger and more reliable than the previous generation's magnetic latch. A kinematics and dynamic simulator and a GUI have been developed to aid the user in planning a reconfiguration or motion sequence of operations. This system has demonstrated the largest number of unique self-reconfiguring parallel steps in a single demonstration at 14.

The SUPERBOT system developed by Shen et al. [21] at USC/ISI is another example of a hybrid system. Building on the M-TRAN design and Shen's earlier CONRO system, one of the primary goals of this project is to develop a system robust and flexible enough to operate in harsh and uncertain environments such as space. Each module has three degrees-of-freedom (two similar to M-TRAN with an added twist degree-of-freedom) and has the capability of sharing power through its bonding mechanism and communicating via high-speed infra-red light emitting diodes (LED). A software hierarchy separates low level device specific code from high level task driven routines. The modules are controlled using hormone-inspired distributed controllers as developed for the CONRO project. Various locomotive gates have also been demonstrated in which modules traverse along carpet, sand, up a slope and across a rope, and self-reconfiguration is planned for the future.

Stochastic

Klavins et al. [3] at the University of Washington has developed a 2D stochastic MSR system named Programmable Parts. Modules are shuffled about randomly on a air hockey table by air jets. When a module collides with another module it bonds using switchable permanent magnets, communicates with the other module and decides whether or not to remain attached. The group has demonstrated that local rules can be developed that allow the system to tend toward and equilibrium of desired configurations. Using theory from statistical mechanics the group is working to develop methods for controlling stochastic MSR systems at various different scales.

Complexity in Robot Configurations

Since MSR systems are designed to be versatile, with numerous configurations for a set of modules, the problem of recognizing and choosing useful configurations is a central area of research. The organized control of modular structures is often a complex task, involving coordinated communication between modules (each which has a processor), central controllers, and in some cases, a human user.

The computational complexity of controlling existing MSR systems varies. Factors such as processor organization (centralized or decentralized), inter-module communication schemes (i. e. global bus, local neighbor-to-neighbor, both global and local), module labeling (unique module IDs vs. unlabeled), and structural symmetry all determine a modular systems' complexity for control and coordinated computation. Ultimately, these hardware parameters determine how computationally complex the control schemes will be.

When a modular robot is controlled with a central controller it is natural to employ identifying labels so the central controller can designate explicit commands over a global bus. Since all processors and modules access a bus

Modular Self-Reconfigurable Robots, Figure 5
Atron system. The Atron system reconfigures in a lattice system, but can form chains as well. This image shows a four "legged" or "wheeled" configuration depending on how the modules are actuated

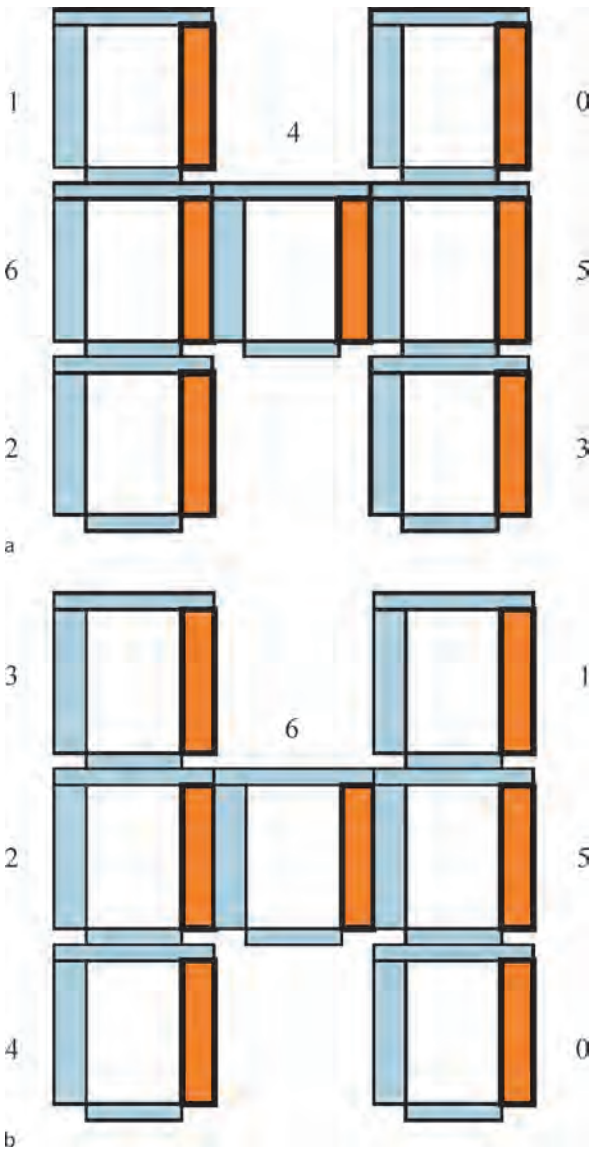
Modular Self-Reconfigurable Robots, Figure 6
MTRAN III four legged configuration. MTRAN modules appear similar to two cubes, one black one white. Walking occurs with chain-like motions, but reconfiguration occurs with modules at specific lattice positions. The cube half-modules checkboard space so white modules only attach to black. This eases the manufacture as one can be male and the other female. (The copyright National Institute of Advanced Industrial Science and Technology (AIST))

equally there is no indication of the relative location of modules within a configuration. Some other mechanism (e. g. the user who constructs the system, or a self-discovery mechanism) must be used to locate each module in a structure and thus map control to each module accordingly.

In the case where a system contains both a global bus and neighbor-to-neighbor communication capabilities (CEBOT, M-TRAN, CKBot, PolyBot), the system can determine a representation of the configuration (e. g., an adjacency matrix). However, for most modular systems adjacency is not enough to represent the full kinematics of the relationship between two modules as two modules maybe be attached together in different ways; for example, two cube-shaped modules may be attached face-to-face on different faces, or with different orientations on each face.

Variants of adjacency matrices [19,27] that take into account how structures are put together add essential structural information, such as inter-module port connections. While this explicit representation is not required for control, it is needed for things like simulation and any type of autonomous behavior that relies on knowing its configuration and state. For example, self-repair or any type of capability reasoning requires this explicit representation.

When doing self-discovery (automatically determining a configuration based on neighbor sensing/communication) it is often useful to see if a configuration is the same



Modular Self-Reconfigurable Robots, Figure 7
Two isomorphic CKBot configurations Modules are rearranged, but the connectivity is the same (similar to nodes on a graph relabeled)

as another configuration; for example, matching a configuration to one in a library of configurations. This problem is related to finding the automorphism group of graph representations, which is known to be a hard problem with no known polynomial time algorithm [7].

The eigenvalues of the port-adjacency matrix (a generalized adjacency matrix that contains port connection numbers to designate how modules are connected) is invariant under any of the $n!$ ways a structure with unique module IDs can be rearranged or relabeled. In graph theory terminology, each relabeling is graph isomorphic to one another. Module ID mappings between isomorphic configurations can be found with a heuristic program such as *nauty* [15], a sequential search through a three-dimensional linked-list representation of the system, or with eigenvectors corresponding to the shared eigenvalues of the isomorphic structures [19].

$$\mathbf{A}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 6 \\ 1 & 0 & 0 & 7 & 6 & 0 & 0 \\ 0 & 1 & 7 & 0 & 4 & 0 & 0 \end{bmatrix}$$

$$\mathbf{A}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 1 & 7 & 0 & 4 \\ 0 & 0 & 7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 7 & 1 & 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 6 & 0 & 0 & 4 & 0 \end{bmatrix}$$

$$\mathrm{Det}\,(\mathbf{A}_1 - \lambda\,\mathbf{I}) = \mathrm{Det}\,(\mathbf{A}_2 - \lambda\,\mathbf{I}) = \lambda^7 - 76\,\lambda^5 + 868\,\lambda^3$$

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

Consider the two isomorphic arrangements of the H-shaped modular CKBot structure and their corresponding port-adjacency matrices, $\mathbf{A}_1$ and $\mathbf{A}_2$, in Fig. 7. Note that the two configurations share the same characteristic polynomial (and hence eigenvalues), as expected since the configurations are rearrangements of the same shape. The property that swapping two columns and two rows of a square matrix does not change its determinant (row swap and column swap each change the determinant by a minus sign) corresponds to rearrangements or relabelings of module IDs. The permutation matrix $\mathbf{P}$ that maps the module IDs between the two isomorphic configurations such that $\mathbf{A}_2 = \mathbf{P}\mathbf{A}_1\mathbf{P}^{-1}$ can be determined using the methods described in [19].

In cases where neighbor-to-neighbor communication is only present (ATRON, Conro hormone studies, Crystalline, Claytronics Atom), distributed algorithms that employ the processors of modules interacting together in parallel divide the computation required for configuration

recognition and motion planning. These MSR systems typically use token-type messages where aggregate configuration information is passed from module-to-module. Complexity scaling is a critical issue for these distributed systems as the number of inter-module messages for goal configuration recognition and planning is immense for as few as 10 modules. A major benefit of the decentralized approach is that for such systems unique module IDs are not necessary since each unit can only communicate with an adjacent unit and thus the system is not limited by an address space. Decentralized approaches also promise to scale as computational resources scale with the number of modules.

For a modular robotic system composed of n homogeneous units, each with c ports, and w ways of connecting modules, an upper-bound number of structurally unique configurations is $(cw)^n$ [19]. For example, given 11 CKBot modules, each unit has 7 ports, each of which can be uniquely connected to another module in 10 ways (3 rotations for the each of the 3 top faces, and 1 orientation for the bottom connection). Therefore an upper-bound to the number of unique configurations is $(10*7)^{11} \cong 2.0*10^{20}$. This number is an upper-bound since there are inherent physical symmetries in certain structures that this approximation double counts.

An enumeration algorithm that more precisely counts the number of non-isomorphic configurations of a modular robot was developed by Chen [8]. Structural and kinematic symmetries were taken into account to find a precise number of unique configurations for a given number of modules. In this method, Polya's Enumeration Theorem is employed to count a structural state only once. For example, two cubes modules that can connect to one another on any of the six faces (each in one orientation) has 36 ways of connecting. Chen's approach takes into account the 3-fold symmetry of the cubes to determine that there is only one unique way of connecting the two modules. This assumes that all six ports on the cubes are all the same; if this symmetry is broken and one or both modules can have multiple types of ports (revolute, helical, cylindrical, etc.) then the algorithm takes these variations into account to find the number of unique ways (greater than one) that the two cubes can be connected.

Another challenge in the field of MSR robotics is the development of *reconfiguration algorithms*: a method that transforms a given robotic configuration to the desired configuration via a sequence of module reconfigurations. A naive centralized method is to perform an exhaustive search of the configuration space (all reachable configurations) beginning with the initial configuration until a path (reconfiguration sequence) to the goal configuration is found. Because it is possible for modules in an MSR system to move in parallel, the branching factor for the search tree is $O(m^n)$ with n being the number of modules free to move and m being the number of ways the module can move. Finding an optimal path might require searching the whole space which is clearly intractable for large n.

Many groups have developed methods and tools for doing self-reconfiguration planning [9,32] that include centralized algorithms. Several groups [2,4,24,25] have presented distributed reconfiguration algorithms. In these cases, the reconfiguration algorithm is embedded on processors running on every module. Each module has an identical program with implicit or explicit knowledge of the required goal states, but only local information about the current state of neighboring modules.

One trend in developing these distributed systems is the use of "meta-modules" – groups of modules together considered to be a subset. As described further in Sect. "Mechanical/Electrical/Computational Interaction", there is a tight coupling between mechanical properties, the constraints on motion and the ease of programming such things as the reconfiguration problem. In many cases, idealized cubes that move in known and unconstrained fashions are used to develop algorithms. However, in the physical world, there are added mechanical constraints that enable the manufacture and motion of the devices. Examples of these constraints include, blocking constraints where modules may block motions under certain conditions [35], checkerboard constraints in lattice configurations where modules may only move to alternate positions (as if they were bi-partite) [18]. By grouping small numbers of modules into a meta-module, many of these constraints can be eased. For example the checkerboard constraint and the blocking constraints can removed for a group of modules moving in concert. However, the system as a whole loses resolution based on the size of the meta-module.

In [1] Abrams and Ghrist introduce the state complex as an extension of the concept of the configuration space. They present an algorithm that uses the added structure defined by the state complex to optimize (with respect to total reconfiguration time) a reconfiguration sequence generated by local planner (such as the aforementioned distributed algorithms.)

Control Architectures

Review of Existing Architectures

A design philosophy behind modular robots is that each module is very simple. In fact, one group [5] proposed the *Ensemble Axiom* "A [module] should include only enough

functionality to contribute to the desired functionality of the ensemble." A module by itself cannot achieve much, but modules arranged together in a system can achieve complex tasks such as manipulation and locomotion. Similarly, the control of a single module is usually simple whereas controlling a system of many modules becomes difficult very quickly. For the overall system, different control architectures have been implemented which we will describe in more detail.

In large part, the implementation of a control architecture depends on the communication structure upon which it is built. Communication between modules can be achieved through a global bus such as CANbus (Controller Area Network, a popular automotive and more recently robotics communications protocol) and/or locally using neighbor-to-neighbor communication such as infra-red (IR) emitter/detector pairs. Many systems use both (Polybot, CKBot, M-TRAN, CONRO and Superbot). Wireless communication is also possible which is architecturally similar to a global bus. In the YaMoR system [16], Bluetooth wireless is the sole means of inter-module communication. ATRON and Crystalline modules [13,20] use only local nearest neighbor IR communication.

As mentioned earlier, control architectures can be implemented in either a centralized or decentralized fashion. In most cases it is easier to develop and analyze a centralized approach. The advantage of decentralized control architecture is that computation is shared among modules. No single unit needs to do all the heavy computation. This is also thought to be more robust and more easily lends itself to scaling to large numbers of modules. It is easier to implement centralized control has using global communications and decentralized using local and there are many examples of such. However, it is possible to implement centralized on a local bus and decentralized on a global bus.

An example of centralized control architecture is implemented on [30]. Each module has its own controller that positions its local actuator. In addition, a master controller communicates to the module controllers to set local behaviors such as setting desired joint angles under position control. In other words, a designated unit sends commands to all the individual modules and synchronizes the action of the whole system. A simple method of implementing this control is to use a *gait control table*. The gait control table is an nxm matrix where m is the number of modules and n is the number of steps of the gait. Each cell in the table holds the desired joint angle for a module. Each column of angles corresponds to the sequence of joint angles for a given module. The controller steps through this table row by row and sends these angles to the corresponding module. Typically stepping through the table occurs at a specified rate, so the vertical axis can represent time. Each module takes the next desired joint angle in the table and interpolates in joint space. The time between steps sets the joint velocity so desired motions have C1 continuity in joint space.

Shen et al. propose a control that is based on biological hormone systems in [23]. The basic idea is that an inter-module "hormone" message is a signal that triggers different actions in different modules while leaving the low-level execution of these actions to the individual modules. The obvious biological analogy occurs when a human experiences sudden fear, and adrenaline hormones released by the brain trigger fight-or-flight behaviors in the body (i. e., the mouth opens, skin gets goose bumps, and the legs jump). Based on this principle, Shen et al. designed a control mechanism that lies somewhere between master and master-less control in that typically one or more modules need to start the hormone messages. It reduces the communication cost for locomotion controls, yet maintains some degree of global synchronization and execution monitoring.

At its root, the hormone is a local message passing system where modules can receive, act on or change messages as they are passed from module to module. An advantage of this type of control is that modules are treated identically without labels or identification numbers; instead the topology of a configuration is the differentiator and thus has a great bearing on the implementation. This lends itself well to simple locomotion control such as undulating gaits however, developing arbitrary motions can be more difficult to implement.

A fully decentralized planning system has been developed by Rus et al. In [4] an algorithm modeled after cellular automata is described. Cellular automata (CA) control uses local rules that are the same for all modules. A rule can be viewed as having a set of pre-conditions. If all those preconditions are satisfied, then a certain action is applied. For example, for a given cell, the pre-conditions could be whether a cell exists at a certain location, whether a cell does not exist at a certain location, and whether a cell is empty. If all preconditions are satisfied, the cell moves itself in a certain direction. Rather than having one master controller being in control of the whole system, modules think for themselves in a parallel distributed fashion. All modules run on the same rules and all modules are programmed with the same code. Just as the hormone method of control adds some complexity to the development of arbitrary motions, it is also difficult to do in the CA case.

Completely centralized control architecture is relatively straightforward to implement. But issues arise when

dealing with millions of modules such as reaching the limits of bandwidth when using a global communication bus. On the other hand it is hard to achieve complex tasks with a completely decentralized architecture that requires only local communication because it is hard to implement behavior in a distributed fashion.

Self-Assembly After Explosion

An example of a hybrid architecture in which global as well as local communication is used is given in [34]. In this work the ability for a modular robot to repair itself is demonstrated by having the robot reassemble into one connected component after disassembly from a high energy event. As a system assembles itself, the connectivity of the robot changes many times. Having disparate disconnected pieces requires a level of decentralized control, however as the system comes together, the modules must act in a coordinated manner as well.

In [34] a demonstration is shown with 15 modules. Modules are grouped into three clusters of five modules. Clusters move as physically separate units, search for and localize each other, and crawl toward each other to connect using magnet faces and form one aggregate unit.

Within each cluster, the modules are attached using screws and an electrical header is included in between these modules to facilitate a global CANbus. The clusters connect to each other using magnet faces without an electrical header so communication is only achieved through IR communication. Thus, this hybrid architecture consisted of a global CANbus within a cluster and local IR communication in between clusters.

The hardware in SAE work is hierarchical – modules form clusters – clusters form systems – the control architecture follows that architecture as well. Each module has an onboard controller that controls the position of the local actuator. Within each cluster a controller communicates on the CANbus to all the modules in that cluster. The master cluster controller gives commands similar to a gait control table to implement behaviors such as crawling, detecting a fallen condition and self-righting, searching for other modules etc. Once clusters dock to each other magnetically, the cluster controllers can communicate to other cluster controllers using a combination of CAN and IR. For this work, one cluster controller is designated as the master whereas the other controllers are designated as slaves and follow the coordinated messages sent by the master cluster controller. For example, in the walking state with all clusters connected, the master cluster controller sends precisely timed messages to the other clusters to coordinate tasks like walking and turning.

Mechanical/Electrical/Computational Interaction

MSR systems sit at an interesting junction between mechanical, electrical, and computational interaction. Robotics in general, is highly interdisciplinary since it requires expertise in all three of those areas. However, the configurability aspect of MSR systems adds to the intertwining of these disciplines. Enabled with electronic technologies (such as communication architecture), modular robotic structures introduce unique mechanical properties, which often require novel computational processes.

Electro-Mechanical Solutions to Computational/Information Problems

The reconfiguration planning problem consists of determining the motions of individual modules to attain a global shape under a variety of constraints. One common constraint is that the MSR system must maintain one connected component (for example, if power is shared from one module to the next if a module disconnects from the group it loses power). Determining whether a module detachment will sever the system into two or more pieces is often computationally and communication bandwidth intensive, as modules may be required to communicate with every other module for this analysis (e. g., the system may be in the shape of a large loop, in which case a disconnection motion between two modules will still leave a single connected component; however, that cannot be determined until every module has communicated at least once). If every module were to simultaneously check whether a disconnection would violate the connectedness constraint, there would be $O(n^2)$ messages and the communications system would quickly become saturated.

One electro-mechanical solution for power distributed systems (shared power between two or more modules) is to use power distribution to determine connectivity. One way this could be achieved would be to develop an intra-module connector where power between connected faces can be temporarily severed, to simulate an actual physical disconnection. If power to either module is lost, it could be concluded that that disconnection violates the connectedness constraint.

Computational Solutions to Mechanical Problems

Applying large forces or torques over some motion path is usually solved mechanically by designing stronger motors or leverage mechanisms. However, with modular robots, this problem can be moved to the computational planning domain. Robot systems with many redundant degrees-of–freedom, such as those typically found with chain style

modular robots, can exploit configurations which have large mechanical advantage [31].

The idea is to utilize the very large mechanical advantage that can be obtained when a system's Jacobian is near a singularity. For example, when using a set of modules that have parallel chains, one chain can be moved to be near a singularity and have large mechanical advantage (e. g. when a human knee is straight the Jacobian representation of the leg loses rank and becomes singular, he can carry much more weight then when it is bent). Consequently, this chain then has a large mechanical advantage which can in turn apply a large force in the desired direction to move the system to a new position. Meanwhile, another parallel chain can be reoriented to be near a singularity at the new position, and then apply large forces yet again to a new position. By repeatedly switching a subset of the motors supporting the load, a ratcheting kind of action can be used to move links to desired positions while under large external forces. If the size of each ratchet motion can be made arbitrarily small, it can be arbitrarily close to the singularity with very large mechanical advantage. Thus, weak motors can be used to provide large forces. Of course, there are practical limits to this method, e. g., sensing accuracy, material strengths, joint precision etc.

The sequence of ratcheting actions that move the end points through desired trajectories is potentially a computationally intensive problem. Hence, the problem of providing sufficiently large torques has now been solved not from a mechanical viewpoint but computationally instead.

This tight integration of computational and electromechanical complexity can be viewed as a wider space from which to find solutions, or as a more complex problem in finding optimal solutions. This is particularly interesting in that it is likely that optimal solutions will not be found by experts in one field, but by the interaction of experts in several fields.

Future Directions

The grand challenges for MSR robotic systems were the results of a workshop where a group of researchers in the MSR robot community gathered and then presented in [33]. A proposed ultimate goal for these systems would be to one day use them in vast numbers for practical applications where un-supervised, adaptive self-organization is needed. Five grand challenges that, if overcome, would enable a next-generation of modular robots with vastly superior capabilities are summarized here:

Big systems Most systems of modular robots have been small in number, especially compared to, for example, the number of components in a living cell (which many researchers view as the best example of a self-organizing, modular system). The demonstration of a system with at least 1000 individual units would suggest that modular robots have come of age.

Self-repairing systems A demonstration of a self-healing structure made up of many distributed, communicating parts would require rethinking algorithms for sensing and estimation of the global state, as well as truly robust hardware and algorithms for reconfiguration that work from any initial condition. A concrete example would be having a system blown up (randomly separated into many pieces) then self-assembling, or recovering from failure of a certain percentage of faulty units.

Self-sustaining systems A demonstration of a system actively running for, say 1 year, in an isolated self-sustaining robotic ecology would require new techniques in power management and energy harvesting, as well as the ability to cope with the inevitable failures.

Self-replication and self-extension While simple robotic self-replication has been demonstrated using few high-level modules, a significant challenge remains to demonstrate self-replication from elementary components and raw materials. The demonstration of a "seed" group of modular robots that can build copies of themselves from raw materials would require advancing beyond a level of complexity that Von Neumann identified as the equivalent of breaking the sound barrier for engineered systems.

Reconciliation with thermodynamics If modular robots are to be miniaturized to micro and/or nano-scale, or if the ideas discovered in this community are even to be tied to nanotechnology, the stochastic nature of nanoscale systems must be addressed. Most existing modular robot systems overcome entropy through brute force and unreasonable amounts of energy. Molecular systems, on the other hand, employ random diffusive processes and are robust to the intrinsic noise found at the nanoscale. The demonstration of a system where stochastic fluctuations are the dominant factor would represent a fundamental advance.

Bibliography

Primary Literature

1. Abrams A, Ghrist R (2004) State complexes for metamorphic robot systems. Int J Robot Res 23(7–8):809–824
2. Bhat P, Kuffner J, Goldstein S, Srinivasa S (2006) Hierarchical Motion Planning for Self-reconfigurable Modular Robots. In: Proceedings of the IEEE/RSJ International Conference on In-

telligent Robots and Systems (IROS), Beijing, October 2006. pp 886–891
3. Bishop J, Burden S, Klavins E, Kreisberg R, Malone W, Napp N, Nguyen T (2005) Self-organizing programmable parts. In: Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), August 2005, pp 3684–3691
4. Butler Z, Kotay K, Rus D, Tomita K (2002) Generic decentralized control for a class of self-reconfigurable robots. In: Proceedings of the 2002 IEEE International Conference on Robotics & Automation (ICRA), Washington DC, May 2002, pp 809–816
5. Campbell J, Pillai P, Goldstein SC (2005) The robot is the tether: active, adaptive power routing modular robots with unary inter-robot connectors. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Edmonton Alberta, August 2005, pp 4108–4115
6. Castano A, Shen W-M, Will P (2000) CONRO: Towards Deployable Robots with Inter-Robots Metamorphic Capabilities. Autonom Robot 8(3):309–324
7. Chartrand G, Lesniak L (1986) Graphs and Digraphs. Wadsworth Publ Co, Belmont
8. Chen I, Burdick J (1993) Enumerating the Non-Isomorphic Assembly Configurations of Modular Robotic Systems. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Yokohama, July 1993, pp 1985–1992
9. Chiang CJ, Chirikjian G (2001) Modular Robot Motion Planning Using Similarity Metrics. Auton Robot 10(1):91–106
10. Chirikjian G (1994) Kinematics of a Metamorphic Robotic System. In: Proceedings of the 1994 IEEE International Conference on Robotics & Automation (ICRA), San Diego 1994, pp 449–55
11. Fukuda T, Nakagawa S (1988) Dynamically reconfigurable robotic system, Robotics and Automation. In: Proceedings 1988 IEEE International Conference, Philadelphia, 24–29 Apr 1988, pp 1581–1586, vol 3
12. Gilpin K, Kotay K, Rus D (2007) Miche Modular Shape Formation by self-Disassembly. In: Proceedings of the 2007 IEEE International Conference on Robotics & Automation (ICRA). Rome, April 2007, pp 2241–2247
13. Jørgensen M, Østergaard E, Lund H (2004) Modular ATRON: modules for a self-reconfigurable robot. In: Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2004, pp 2068–2073
14. Kotay K, Rus D, Vona M, McGray C (1998) The Self-reconfigurable robotic molecule. In: Proceedings of the 1998 IEEE Intl. Conf. on Robotics and Automation (ICRA), May 1994, Leuven, Belgium, May 1998, pp 424–431
15. McKay B (1981) Practical graph isomorphism. Congressus Numerantium 30:45–87
16. Moeckel R, Jaquier C, Drapel K, Dittrich E (2006) YaMoR and Bluemove – an autonomous modular robot with Bluetooth interface for exploring adaptive locomotion. Climbing and Walking. Springer, Berlin Heidelberg, pp 285–692
17. Murata S, Kurokawa H, Kokaji S (1994) Self-Assembling Machine. In: Proceedings of the 1994 IEEE International Conference on Robotics & Automation (ICRA), San Diego, May 1994, pp 441–448
18. Murata S, Yoshida E, Kamimura A, Kurokawa H, Tomita K, Kokaji S (2002) M-TRAN: Self-Reconfigurable Modular Robotic System. IEEE/ASME Trans Mechatron 7(4):431–41
19. Park M, Chitta S, Teichman A, Yim M (2008) Automatic Configuration Recognition in Modular Robots. Int J Robot Res 27(3–4):403–421
20. Rus D, Vona M (2000) A physical implementation of the self-reconfiguring crystallinerobot. In: Proceedings of the 2000 IEEE International Conference on Robotics & Automation (ICRA), San Francisco, April 2000, pp 1726–1733
21. Salemi B, Moll M, Shen W-M (2006) SUPERBOT: A Deployable, Multi-Functional, and Modular Self-Reconfigurable Robotic System. In: Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Beijing, October 2006, pp 3636–3641
22. Sastra J, Chitta S, Yim M (2006) Dynamic Rolling for a modular loop robot. In: Proceedings of the International Symposium on Experimental Robotics, Rio de Janerio, July 2006
23. Shen W-M, Salemi B, Will P (2002) Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots. IEEE Trans Robot Autom 18(5):700–712
24. Vassilvitskii S, Yim M, Suh J (2002) A complete, local and parallel reconfiguration algorithm for cube style modular robots. In: Proceedings of the 2002 IEEE International Conference on Robotics & Automation (ICRA), Washington DC, May 2002, pp 117–122
25. Walter J, Welch JL, Amato NM (2004) Distributed reconfiguration of metamorphic robot chains. Distrib Comput 17(2):171–189
26. White PJ, Kopanksi K, Lipson H (2004) Stochastic self-reconfigurable cellular robotics. In: Proceedings of the 2004 IEEE International Conference on Robotics & Automation (ICRA), New Orleans, April 2004, pp 2888–2893
27. Will P, Castano A (2001) Representing and Discovering the Configuration of Conro Robots. In: Proceedings of the 2001 IEEE International Conference on Robotics & Automation (ICRA), Seoul, May 2001, pp 3503–09
28. Yim M (1994) Locomotion with a Unit Modular Reconfigurable Robot. PhD Thesis, Stanford University
29. Yim M (1994) New locomotion gaits, Robotics and Automation. In: Proceedings 1994 IEEE International Conference, San Diego, 8–13 May 1994. pp 2508–2514, vol 3
30. Yim M, Duff DG, Roufas KD (2000) PolyBot: a modular reconfigurable robot. In: Proceedings of the 2000 IEEE International Conference on Robotics & Automation (ICRA), San Francisco, April 2000, pp 514–520
31. Yim M, Duff DG, Zhang Y (2001) Closed-chain motion with large mechanical advantage. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Maui, October 2001
32. Yim M, Goldberg D, Casal A (2002) Connectivity Planning for Closed-Chain Reconfiguration. In: Proceedings of the SPIE Vol 4196, Sensor Fusion and Decentralized Control in Robotic Systems III, October 2002, pp 402–412
33. Yim M, Shen W-M, Salemi B, Rus D, Moll M, Lipson H, Klavins E, Chirikjian GS (2007) Modular Self-Reconfigurable Robot Systems: Grand Challenges of Robotics. IEEE Robot Autom Mag 14(1):43–52
34. Yim M, Shirmohammadi B, Sastra J (2007) Towards Self-reassembly After Explosion. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 2007, pp 2767–2772
35. Yim M, Zhang Y, Lamping J, Mao E (2001) Distributed control for 3D metamorphosis. Auton Robot 10(1):41–56

Books and Reviews

Butler Z, Fitch R, Rus D, Wang Y (2002) Distributed Goal Recognition Algorithms for Modular Robots. In: Proceedings of the 2002 IEEE International Conference on Robotics & Automation (ICRA), Washington DC, May 2002, pp 110–16

Fukuda T, Kawauchi Y (1988) Dynamically Reconfigurable Robotic System. In: Proceedings of the 1988 IEEE International Conference on Robotics & Automation (ICRA), Philadelphia, April 1988, pp 1581–86

Fukuda T, Nakagawa S, Kawauchi Y, Buss M (1989) Structure decision method for self organising robots based on cell structures-CEBOT Robotics and Automation, 1989. In: Proceedings of 1989 IEEE International Conference, Scottscale, 14–19 May 1989. vol 2. pp 695–700

Murata S, Yoshida E, Kamimura A, Kurokawa H, Tomita K, Kokaji S (2002) M-tran: self-reconfigurable modular robotic system. IEEE/ASME Trans Mechatron 7(4):431

Ostergaard EH (2004) Distributed control of the ATRON Self-Reconfigurable robot. PhD, Univ. of Southern Denmark

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

Yim M, Zhang Y, Duff D (2002) Modular Reconfigurable Robots, Machines that shift their shape to suit the task at hand. IEEE Spectr Mag 39(2):30–34

Zhang Y, Roufas K, Yim M (2001) Software Architecture for Modular Self-Reconfigurable Robots. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), Hawaii, October 2001

http://www.springer.com/978-0-387-75888-6

Encyclopedia of Complexity and Systems Science
Editor-in-chief: Meyers, R.A.
2009, MCXX, 10398 p. In 14 volumes, not available separately., Hardcover
ISBN: 978-0-387-75888-6