M-TRAN: Self-Reconfigurable Modular Robotic System

Satoshi Murata, *Member, IEEE*, Eiichi Yoshida, *Member, IEEE*, Akiya Kamimura, Haruhisa Kurokawa, Kohji Tomita, *Member, IEEE*, and Shigeru Kokaji, *Member, IEEE*

***Abstract*—In this paper, a novel robotic system called modular transformer (M-TRAN) is proposed. M-TRAN is a distributed, self-reconfigurable system composed of homogeneous robotic modules. The system can change its configuration by changing each module's position and connection. Each module is equipped with an onboard microprocessor, actuators, intermodule communication/power transmission devices and intermodule connection mechanisms. The special design of M-TRAN module realizes both reliable and quick self-reconfiguration and versatile robotic motion. For instance, M-TRAN is able to metamorphose into robotic configurations such as a legged machine and hereby generate coordinated walking motion without any human intervention. An actual system with ten modules was built and basic operations of self-reconfiguration and motion generation were examined through experiments. A series of software programs has also been developed to drive M-TRAN hardware, including a simulator of M-TRAN kinematics, a user interface to design appropriate configurations and motion sequences for given tasks, and an automatic motion planner for a regular cluster of M-TRAN modules. These software programs are integrated into the M-TRAN system supervised by a host computer. Several demonstrations have proven its capability as a self-reconfigurable robot.**

***Index Terms*—Distributed autonomous control, metamorphic robotics, modular robot, reconfiguration planning, self-reconfiguration.**

I. Introduction

A SELF-RECONFIGURABLE robot belongs to a class of robotic system that can change its shape and functionality without external help. Such robots are composed of many robotic modules where the different types of the modules are much less than the number of modules. Each module is equipped with computational and communication capability, sensors and actuators. The component modules change connective relations among themselves. Self-reconfigurability means that the system can metamorphose into various shapes by changing connectivity among the modules without external help. Many configurations, such as a manipulator, a crawler, a legged robot or other robotic configurations, can be built by the combination of (identical) modules. This kind of flexibility is highly desirable for robotic systems used in unstructured and unpredictable environments such as space and deep-sea exploration, or rescue operations in earthquake stricken areas. Another benefit from self-reconfiguration is self-repair by replacing damaged modules by spare modules.

Recently, this flexibility and robustness of self-reconfigurable robots has attracted many researchers and numerous systems (some hardware and some in simulations) have been proposed to demonstrate the feasibility of the concept. In the early development of self-reconfigurable robots, most such systems were two-dimensional. *Fracta* [1] and *metamorphic robot* [2] are two such two-dimensional systems with mechanical hardware modules. Both of them are homogeneous in hardware (all the modules are identical) and the modules have a hexagonal shape. The former system is homogenous in software (all the modules are equipped with the same processor with the same program) and can do primitive self-repair [1], [3]. Meanwhile, three-dimensional (3-D) systems have been developed for more realistic applications.

There are two classes of three-dimensional systems, a class based on a space filling polyhedron (lattice systems) and a class of linear (or string) systems. In the former class, the shape of a module is determined by a space-filling polyhedron, such as a regular cube (*3-D self-reconfigurable system (3-D SRS)* [1]; *crystalline* [4]; *I-Cube* [5]; or a rhombic dodecahedron (*Proteo)* [6]). Other systems are based on a cubic lattice too, but having a module fill in three adjacent cells; diagonal cells and a connection arm in the off-diagonal cell (*Molecule*) [7]. Naturally, modules of this class are homogeneous like atoms in a crystal. The advantage of these systems is easy self-reconfiguration. Modules are always placed at regular grid points, thus relative positioning control among the modules is not necessary. However, hardware realization of these systems tends to be complicated because geometrical symmetry requires many DOFs for actuation and connections for each module. So far, experiments with hardware have been limited to small-scale systems consisting of only a few modules. The latter class of system is a linear or string-type system, where a series of actuated joint modules forms a robotic tentacle (*CEBOT* [8], *PolyBot* [9], [10], *CONRO* [11]). Usually, some connector modules or fork modules are introduced in these systems, thus modules in this class are heterogeneous. Relatively small DOFs per module are required in this class, enabling it to realize dexterous motion as a multijoint robot. Self-reconfiguration, especially reconnecting detached modules, tends to be difficult in these kinds of systems. For instance, an automatic reconnection method using optical sensors is necessary and the inverse kinematics of a multijoint system must also be considered.

Manuscript received March 31, 2002; revised October 1, 2002. Recommended by Technical Editors W.-M. Shen and M. Yim.

S. Murata is with the Tokyo Institute of Technology, Yokohama, 226-8502 Japan (e-mail: murata@dis.titech.ac.jp).

E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji are with the National Institute of Advanced Industrial Science and Technology (AIST), Ibaraki, 305-8564 Japan (e-mail: e.yoshida@aist.go.jp; kamimura.a@aist.go.jp; kurokawa-h@aist.go.jp; k.tomita@aist.go.jp; s.kokaji@aist.go.jp).

Digital Object Identifier 10.1109/TMECH.2002.806220

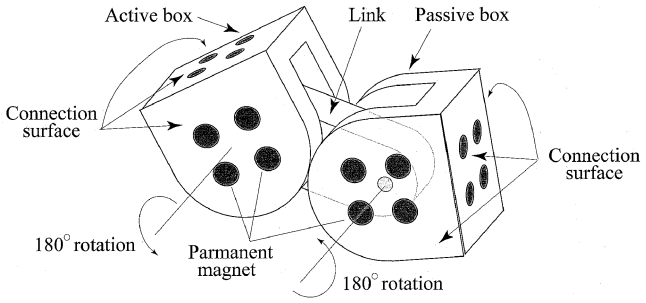

Fig. 1. Schematic view of the module.

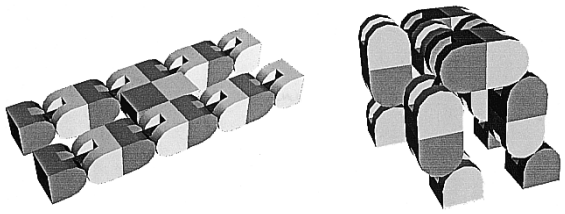

Fig. 2. Static structure (left) and its motion (right). In the left, all the rotational angles of the modules are restricted to $-90°$, or $0°$, or $90°$ and consequently, the modules are placed on lattice points. In the right, rotation angle has no such restriction to realize robotic motion.

In this paper, we propose a novel, self-reconfigurable robot called modular transformer (M-TRAN) [12]. Its special design of the module hardware realizes both reliable and quick self-reconfiguration and versatile robotic motion. It is a kind of double-cube module like Molecule, but requires relatively fewer DOFs for connection and actuation per module than other systems. M-TRAN behaves as a lattice type system for self-reconfiguration and then acts as a string-type robotic system. For instance, it is able to metamorphose into robotic configurations such as a legged machine and then generate coordinated walking motion without any human intervention. An actual system with ten modules was built and its self-reconfiguration and motion generation were examined by experiments. A series of software programs has also been developed to drive M-TRAN hardware, including a simulator of M-TRAN kinematics, a user interface to design appropriate configuration and motion sequences for given tasks [13] and an automatic motion planner for a regular cluster of M-TRAN modules [14], [15]. These software programs are integrated into the M-TRAN system. Several demonstrations have shown its ability as a self-reconfigurable robot.

This paper consists of six sections. Mechanical design and hardware of M-TRAN is described in detail in the next section and software for self-reconfiguration and motion generation is described in Section III. Experimental results are shown in Sections IV, and V briefly compares M-TRAN and other systems. Concluding remarks are given in the last section.

II. Hardware

This section describes the design of the M-TRAN module, presenting hardware details and the control architecture of the M-TRAN system.

A. M-TRAN Module

Fig. 1 shows a schematic view of the M-TRAN module. The module is composed of two semi-cylindrical boxes and a link linking them together. The curved part of each box is a semi-cylinder touching a regular cube, thus each box can rotate from $-90°$ to $90°$ around the axes at both ends of the link independently by two servomotors embedded in the link. Each module has two kinds of boxes: active and passive. Both boxes have three connection surfaces utilizing permanent magnets. Polarities of permanent magnets in active and passive connection surfaces are the same on the same box but opposite in the other box. By checkerboard parity, active boxes always meet passive ones as long as the angles of link motors are set to be multiples of a right angle. Connection surfaces of an active box are equipped with special actuation mechanisms for detachment, while the passive surface consists of a simple plate with magnets. An on-board processor and other circuitry are installed in the passive box. The connection surface gives not only mechanical connection between modules but also provides electrical connection for inter-module communication and power supply. Connection in four directions is possible by symmetrical design of electrodes on the surface. In the self-reconfiguration phase, the link angles must be multiples of a right angle. In the motion generation phase, however, they can be set to arbitrary angles as ordinary robotic joints (Fig. 2).

Reconfiguration is achieved by repeating basic operations such as detaching a surface from the neighbor, rotating a semi-cylindrical box and reconnecting the surface to another neighbor. Fig. 3(a)–(c) illustrate simplified schemes on how to move a module in the M-TRAN system. A module on the floor tiled with passive and active connection surfaces, can rotate around the horizontal axis [Fig. 3(a)], or rotate around the vertical axis [Fig. 3(b)]. Although a single M-TRAN module does not have enough DOFs to switch from one posture to another, this is possible by using a partner module [Fig. 3(c)]. Here, we assume the module has enough torque to lift one module in any posture. The actual reconfiguration process is not as simple as shown in this illustration and we need to combine these actions. This issue will be discussed in detail in Section III.

B. Mechanical Design and Connection Mechanism

Fig. 4 shows the developed module and its inner structure. The frame of an M-TRAN module is made from a block of engineering plastic (delrin) to ensure structural strength and lightness. The box shape of the module is also suitable for stacking in a cluster. The link part includes two sets of a precision-geared motor, reduction gear and servo circuit. Cables connecting active and passive boxes run inside of the link. The connection surfaces are made of glass-epoxy fiber circuit boards to decrease the number of electric wires and total weight.

For the mechanical connection, four Samarium-Cobalt permanent magnets are embedded in each connection surface. For electrical connection, two pairs of electrodes for power supply and one electrode for serial communication are also embedded in the surface. As mentioned before, symmetrical arrangement of electrodes allows connection in four directions.

The active connection surface has a connection mechanism based on the internally balanced magnetic unit [16]. The mechanism is composed of nonlinear springs, shape memory alloy (SMA) coils and magnets fixed on a moving part called the connecting plate (Fig. 5). In this mechanism, the magnetic potential energy is not dispersed but changed into internal elastic energy

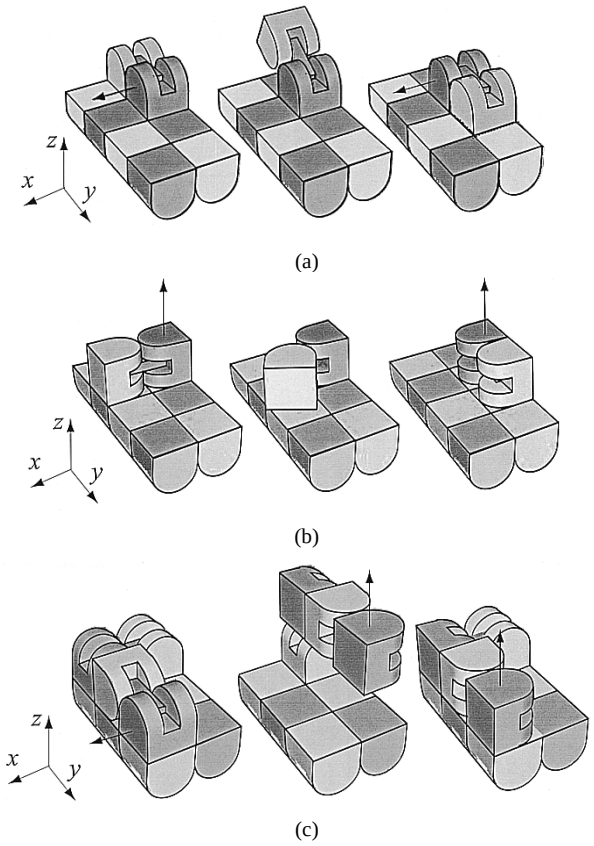

Fig. 3. Basic self-reconfiguration schemes. Arrows in this figure show rotational axes used in the self-reconfiguration schemes. (a) Forward roll: By rotation around the x axis, the module travels on a line. Although it cannot change the direction of the line, the module can change vertical level by climbing over another module. (b) Pivot translation: By rotation around the z axis, the module traverses the plane. The module can head to any direction but cannot change its vertical level. (c) Mode conversion: A module attached with an arrow is lifted up by a module behind (converter module) and placed back at the same position but in different posture.

of springs. The nonlinear spring is designed so that the repulsion by the spring is slightly weaker than the attraction by the magnets and the difference between these two forces is designed to be constant for any position of the connecting plate (Fig. 6). Note that when the magnets are attached, the repulsive force of the nonlinear springs is internally balanced inside the active box (A in Fig. 5) and does not affect the connecting force between two boxes (A and P). Connection is easily released when the SMA coils generate a larger force than the difference between magnetic attraction and spring repulsion (Fig. 6). (If the connection between adjacent surfaces is undesired for robotic motion, the SMAs should be heated to avoid the connection.) In the experiments, we need to heat SMA coils for 5–15 s for detachment. Once SMA is heated, it cannot reconnect immediately; we need to allow 20–30 s for cooling before reconnection. Connecting force (P in Fig. 6) by the magnets was 25 N which is enough to support another module in any posture (see also Section V).

C. Electrical Design and Control System Architecture

The passive box contains the onboard processor and other circuitry (Fig. 7). The onboard circuitry includes an onboard microprocessor (BASIC STAMP II, Parallax, Inc.), a power supply circuit and a pulse generator circuit for driving the SMA coils. The microprocessor has a dipswitch input to set an ID number.

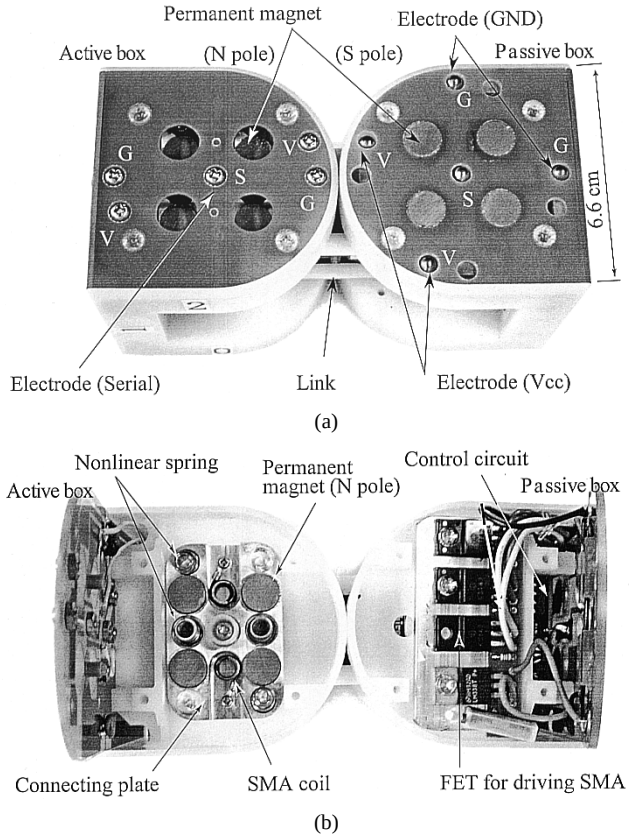

Fig. 4. Photos of the developed module and its inner structure. (a) Appearance. (b) Inner structure. Opened top lid of both boxes.

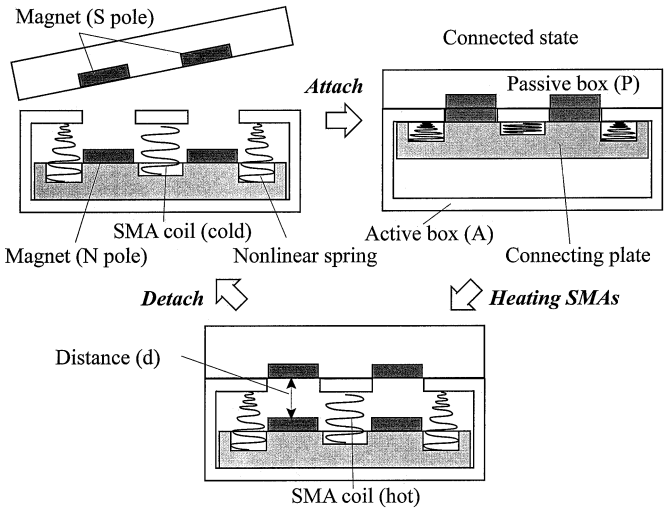

Fig. 5. Connection mechanism.

The rotational angles of the two servomotors are controlled by pulsewidth modulation (PWM) signals generated by the microprocessor and sent to the servo circuits. To break a connection to another module, the microprocessor drives the SMA coils in the active connection surface by pulsated current, while electric contact points embedded in the connecting plate and the back of connection surface check the connection status (connected or detached).

Fig. 8 shows a schematic diagram of the control system architecture. It consists of the host PC, relay PIC (BASIC STAMP II, the same one as in the module) and module processors. All

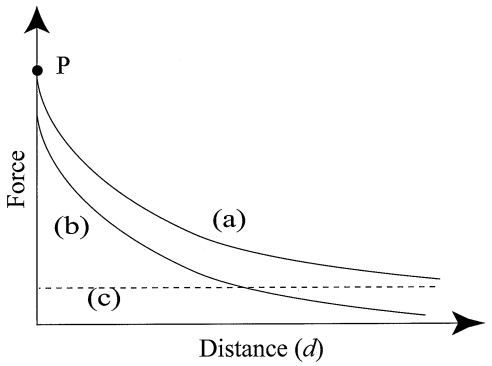

Fig. 6. Magnetic force and design of nonlinear spring. Relation between magnetic force and repulsive force of nonlinear springs is shown: (a) magnetic force, (b) repulsive force of nonlinear springs, (c) difference between (a) and (b). Curve (b) shows an ideal characteristic of the nonlinear spring, thus the difference (c) is constant. In implementation, SMA is required some marginal force to absorb deviation from the constant.

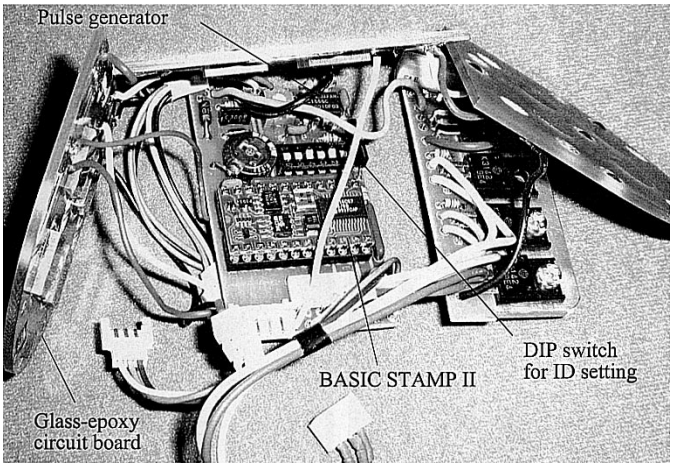

Fig. 7. Control circuits in the passive box.

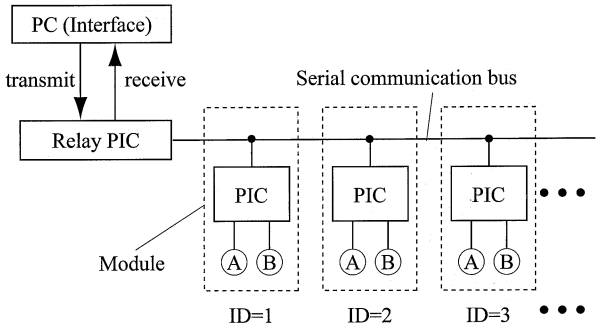

Fig. 8. Schematic of the control system architecture.

communication is 4800-b/s asynchronous serial communication. The host PC and the relay PIC are connected by ordinary bilateral communication, while single-wire communication with token passing is adopted between the relay PIC and modules to reduce the number of electrodes. First, the host PC issues a control command that includes the module's ID number. The relay PIC then broadcasts the command to all the modules on the serial bus. A module sends back a validation signal if its ID coincides and executes the command. After the execution of the command, it sends a completion signal to the host PC via the relay PIC. When no validation is returned from the module, the host PC sends the command again for recovery. There is no restriction on the network topology.

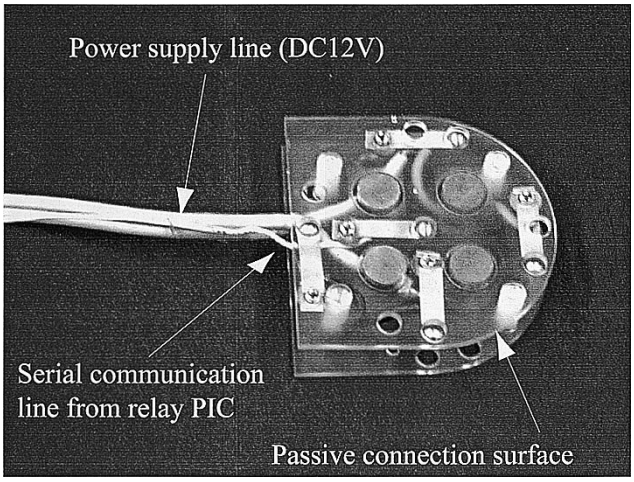

Fig. 9. Photo of the base plate.

TABLE I
SPECIFICATIONS OF M-TRAN

Item	Value
Dimension	66 × 132 × 66mm
Weight	0.44kg
CPU	BasicStamp II
Communication	4,800 bps (asynchronous serial communication)
Power supply	DC 12V
Max. torque of each axis	23 kg cm (rating)
Connecting force	25 N
Total power dissipation	0.35W(12V)

A base plate provides electrical connection between the relay PIC and M-TRAN modules (Fig. 9). The base plate is actually a passive connection surface, which can be attached to an active surface of any module.

Module specifications are summarized in Table I.

III. SOFTWARE

We need several types of software to control the hardware of a self-reconfigurable robot. In this section, we first explain the difficulty of the general self-reconfiguration problem and then describe the motion design interface and locomotion planner for the M-TRAN cluster.

A. Difficulty in Self-Reconfiguration Problem

The most desired software for a SRS is no doubt a general planner that is able to verify if two arbitrary configurations A and B are interchangeable and then to calculate a reconfiguration path between the configurations. However, this kind of general self-reconfiguration algorithm is known to be difficult even for 2-D systems [17]. Such difficulty has its roots deep in the nature of the many DOF searching problems of modular architecture. There exist configurations that are not interchangeable for M-TRAN as shown in the example in Fig. 10. The following discussion focuses on cases where reconfiguration is feasible.

The key issue in designing a self-reconfiguration planner is defining the *metric*, which indicates the difference between two

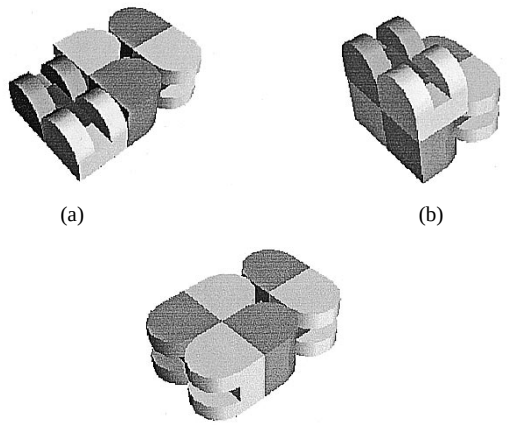

Fig. 10. Reconfigurability: Three modules are on a flat surface of other modules (not drawn), assuming that the modules can attach to this surface, reconfiguration from (a) to (b) is possible while from (a) to (c) is not.

configurations. For most 2-D lattice systems and for isotropic 3-D modules, there is a good correspondence between distance in lattice space and distance evaluated as the number of necessary motion steps. Therefore, the lattice distance can be used as a metric for those systems and it gives planning methods at reasonable cost [1], [3], [7], [18]. This property also facilitates employing graph-search techniques [5], [19] including real-time A * searches [20].

However, other hardware constraints for modules must be taken into account for many self-reconfigurable systems with less symmetry. Typical hardware constraints are as follows.

a) *Connectivity*—All modules must remain connected.
b) *Collision*—Collisions between modules must be avoided.
c) *Torque limit*—One module can only lift one or a few modules.

For M-TRAN, changing the posture of one module is difficult in some cases, as it involves two modules in cooperation and this makes the problem more complicated. Figs. 11 and 12, show that simple lattice distance gives almost no information in the M-TRAN system. In both figures, we deal with a three-module cluster composed of three modules M1 to M3 located on the floor tiled with connecting surfaces. Here, a set of module motions is referred to as a "step" that can be achieved simultaneously under these hardware constraints. Moving a module on the floor onto other modules as in Fig. 11 requires a lengthy reconfiguration path of 15 steps. In contrast, in Fig. 12, the lattice distance between the initial position and the goal is greater than in the previous case, but this is achieved in a single step.

To cope with such difficulty of planning, we have developed two types of software. The first is a motion design interface, which helps a human programmer to design a reconfiguration sequence and motion generation through a powerful graphic interface. The second is a locomotion planner for an M-TRAN cluster, in which the above difficulties are relaxed by introducing some regularity into the structure.

B. Motion Design Interface

A graphic interface significantly helps users to plan reconfiguration sequences in 3-D space. We have developed a motion

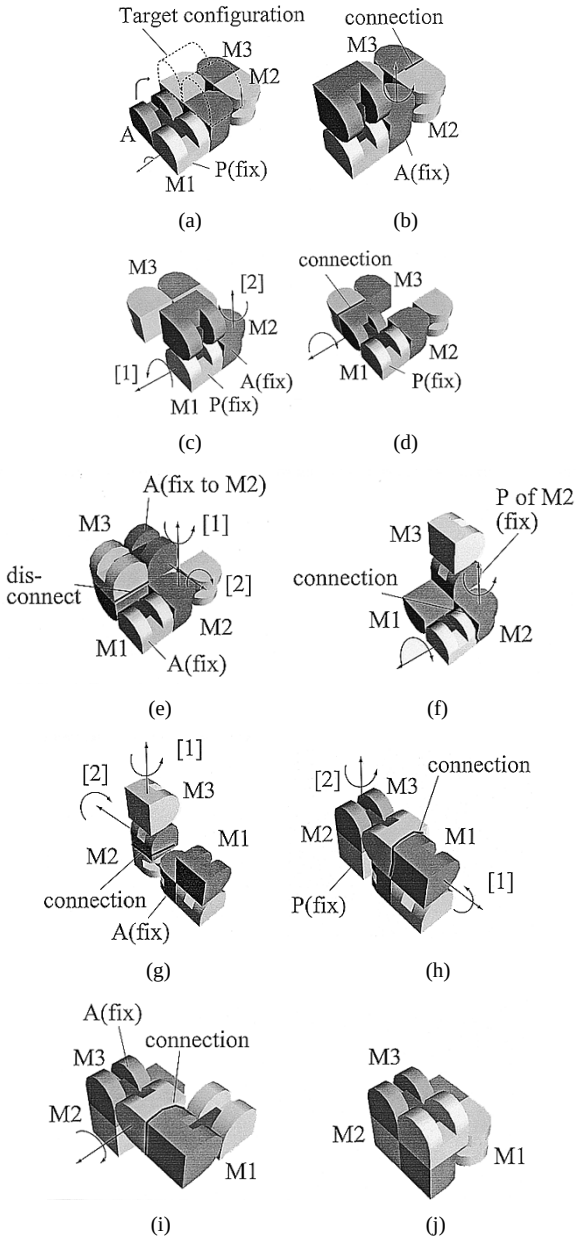

Fig. 11. A planning problem requiring many motions. Starting from the initial state (a), to the final configuration indicated by dotted line. Here, modules and their active and passive parts are indicaed by M_i, *A*, and *P*, respectively. (a) M1 lifts A with P fixed. (b) M2 carries M3 through their connection. (c) M1 connects to M3, then M2 disconnects from M3. (d) M1 lifts M3 with P fixed. (e) M2 connects to M3, then by fixing A to M2 and lifting P, M3 disconnects from M1. (f) M2 carries M1 forward, then M1 lifts A with P fixed. (g) M2 rotates M3 CCW, then M3 lowers P to connect to M1. (h) M1 detaches P from the floor upwards still connecting to M3, then M2 moves A CCW with P fixed. (i) Finally, M3 lowers M1 to the final configuration.

design interface for M-TRAN using the OpenGL Library [13]. Fig. 13 shows windows of the developed interface. The users can design any configuration by indicating the position and orientation of each module by simple mouse operation. The interface allows the users to design a sequence of module motions in a similar interactive way. It checks the connectivity of all modules and alerts the designer if some part of the system is disconnected. Collision between the modules is also checked.

A static mechanics simulator is also incorporated in the interface software. It displays information such as center of gravity

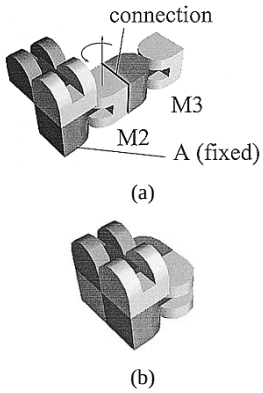

Fig. 12. A planning problem requiring fewer motions, but with larger distance in lattice space than in Fig. 10. From the initial state, M2 carries M3 CCW with A fixed, into the final configuration. (a) Initial state, (b) Final state.

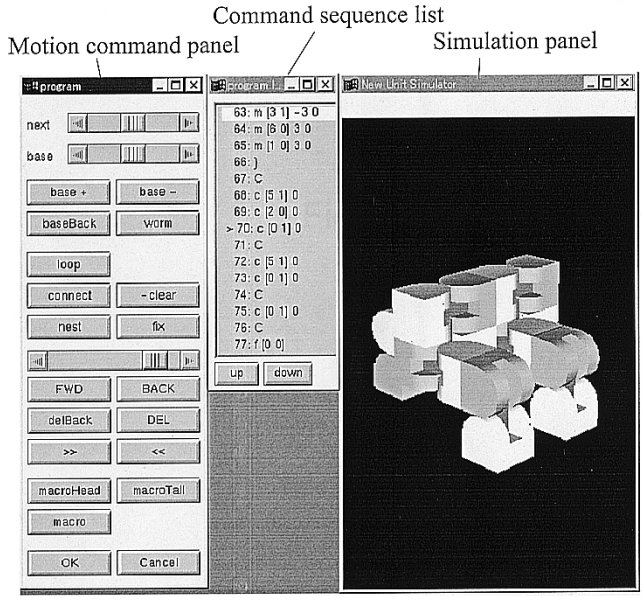

Fig. 13. Snapshot of the graphical user interface.

and contact area with the floor. These are necessary to design walking motion where the c.g. must be contained in the convex hull made by supporting feet.

Another utility of the interface is macro commands. Sometimes the same short sequence of reconfiguration is useful in many situations. Those sequences can be registered as macros in reusable form. Users can edit a list of structured commands involving plain motion commands and macros.

Figs. 14 and 15, show snapshots from the reconfiguration sequence planned by the software. In Fig. 14, a long cluster of M-TRAN crosses an obstacle by repeating the same reconfiguration sub-sequences. In Fig. 15, a more complicated sequence is built by using macro commands. Here, a two-layer cluster (a) is used to generate walking robots. In the first stage, some modules at the end of the cluster are elevated on the cluster, transported to the other end (b) and metamorphosed into an H-shape structure made of nine modules attached to the side of the cluster (c). The modules are then separated from the main body and rolled up to form a double-loop crawler configuration (d). It crawls for a while (e) and then stands up (f) and changes into a quadruped robot (g). This production sequence can be repeated

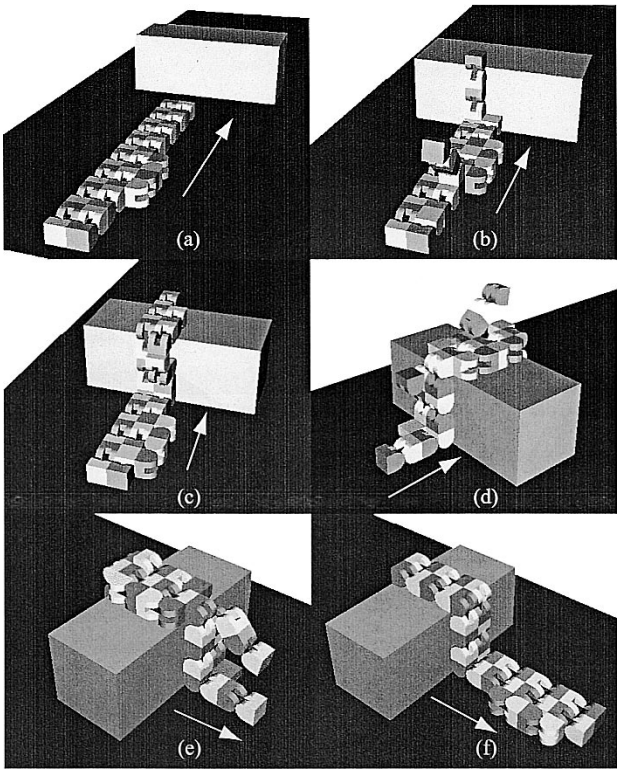

Fig. 14. Modular robot crossing an obstacle through reconfiguration. Arrows indicate the direction of overall motion. (a) Initial state. The modular robot starts to advance in the vertical direction (b), then advances atop the obstacle, adapting its shape (c, d) and finally reaches to the other side of the obstacle (e, f).

as long as the cluster remains. Most of this sequence is programmed by macros.

C. Locomotion Planning

This section describes a planner for locomotion with reconfiguration [14], [15] that enables a serial collection of four-module *blocks* (Fig. 16) to move along a desired 3-D trajectory through self-reconfiguration as illustrated in Fig. 17. Here, we focus on building a feasible planner for locomotion of a particular class of module.

An advantage of this cluster structure is that any serial connection of the blocks maintains the connectivity of the whole cluster. Moreover, regarding a block as one “large module” at a 3-D lattice point simplifies the 3-D planning problem, which can narrow the search space. A couple of modules with different rotation axis directions, called *converters*, are attached to the top of the cluster (Fig. 16). They are used to change the direction of the rotation axis of the modules in the cluster.

We have developed a centralized locomotion planning framework composed of global and local planners (Fig. 17). We will briefly outline these planners (refer to our previous papers [14], [15] for details). The global planner decides the global motion called *flow* for the cluster. The flow is actually a trajectory in 3-D space that the cluster must follow. The cluster motion is realized by sending the tail block toward the head. More precisely, the global planner outputs candidates of path running along the cluster for each module in the block. The planner issues these motion commands for the modules so that the overall connectivity is maintained. The local planner decomposes the global

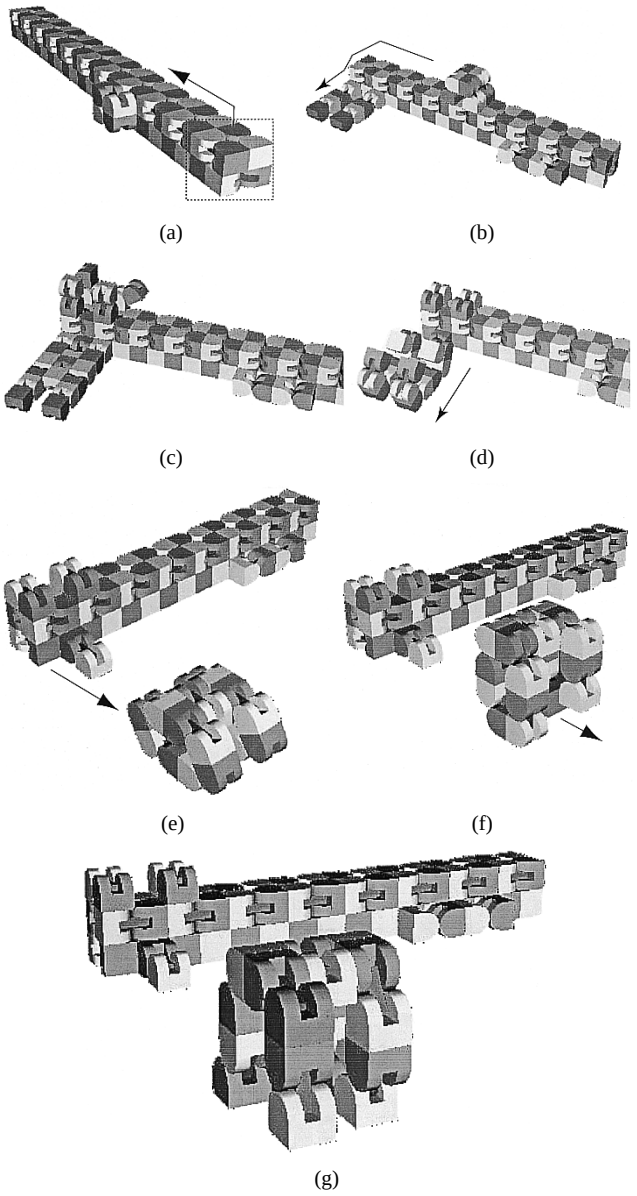

Fig. 15. Reconfiguration from a block structure to a quadruped robot via a crawler.

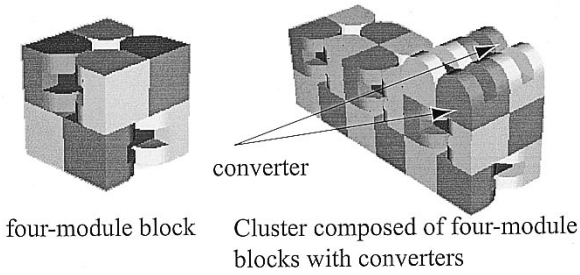

Fig. 16. A cluster composed of four-module blocks with two converter modules.

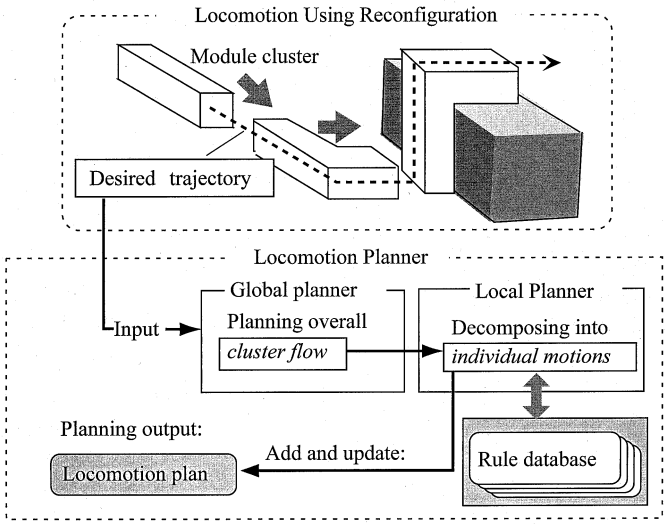

Fig. 17. Locomotion planner architecture.

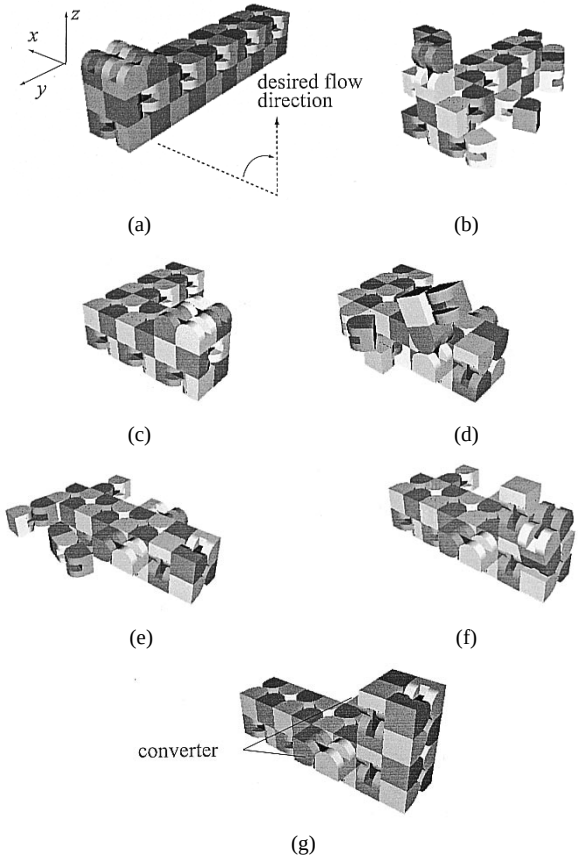

Fig. 18. Generated plan for desired motion by 22 modules. Simultaneous module motions are allowed in a single step under hardware constraints. Starting from the initial configuration (a), the cluster moves by two blocks in $-x$ direction (b, c). The flow is changed in z direction using converter modules (d, e) and advances by two blocks again using converters (f), into the final configuration (g). (a) Desired trajectory. (b) step 25. (c) step 80. (d) step 123. (e) step 146. (f) step 170. (g) step 199 (finished).

motion commands into steps of local module motion based on a rule database. Each rule in the database includes a *motion scheme* that is a local step motion associated with an applicable local connectivity. From a set of matched rules, the local planner selects one rule that gives the forward movement along the path given by the global planner. In this selection, hardware constraints (a) to (c) in Section III-A are taken into account. By repeating this block-based motion, the planner can generate a feasible motion sequence to achieve the desired 3-D motion. The planned sequence is reorganized into concurrent module

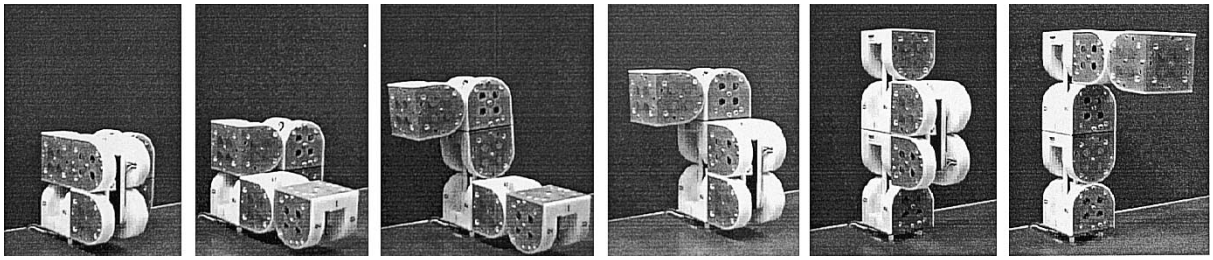

Fig. 19. Self-reconfiguration experiment.

motions, which make the best of the parallelism of the modular robot.

Fig. 18 illustrates a planned motion of a 22-module cluster. In the desired trajectory, the cluster first advances by two blocks in the $-x$ direction. It then changes the flow direction to the z direction and advances by two blocks. Simultaneous module motions that satisfy the hardware constraints are achieved. In this plan, approximately 30 manually coded rules are provided in the database.

IV. Experiments

We produced ten M-TRAN modules and performed several experiments to demonstrate the system's feasibility (video clips are available on the Web.[1] .

A. Basic Motion of Self-Reconfiguration

Fig. 19 shows an elementary reconfiguration experiment with three modules. This experiment successfully realized automatic reconfiguration from the initial to the final configuration. Relative position errors between the connection surfaces were absorbed by the magnetic connection and a reliable electrical connection via the connection surfaces was verified. Fig. 20 shows the experiment of the eight-module cluster flow motion described in Section III-C. It shows that the module cluster travels forward by one block according to the motion sequence made by the motion planner.

B. Locomotion and Transformation

We realized self-reconfiguration between different robotic configurations. The same sequence of the simulation (metamorphosis from H-shape structure to a crawler and quadruped walker in Fig. 15) is examined (Fig. 21). Power consumption without motion in this experiment was 4 to 6 W. Peak power for each motion was 26 W for crawling [Fig. 21(c) and (d)], 53 W for standing (e), 36 W for SMA heating (lasting 5 to 8 s) and 13 and 26 W for walking (g and h). In this experiment, the reconfiguration sequence was preprogrammed in the host PC and sent to the modules. No human intervention was necessary during the experiments.

V. Discussion

This section compares several self-reconfigurable systems in terms of module design, efficiency, structural strength and motion planning.

[1][Online]. Available: http://staff.aist.go.jp/e.yoshida/test/index-e.htm

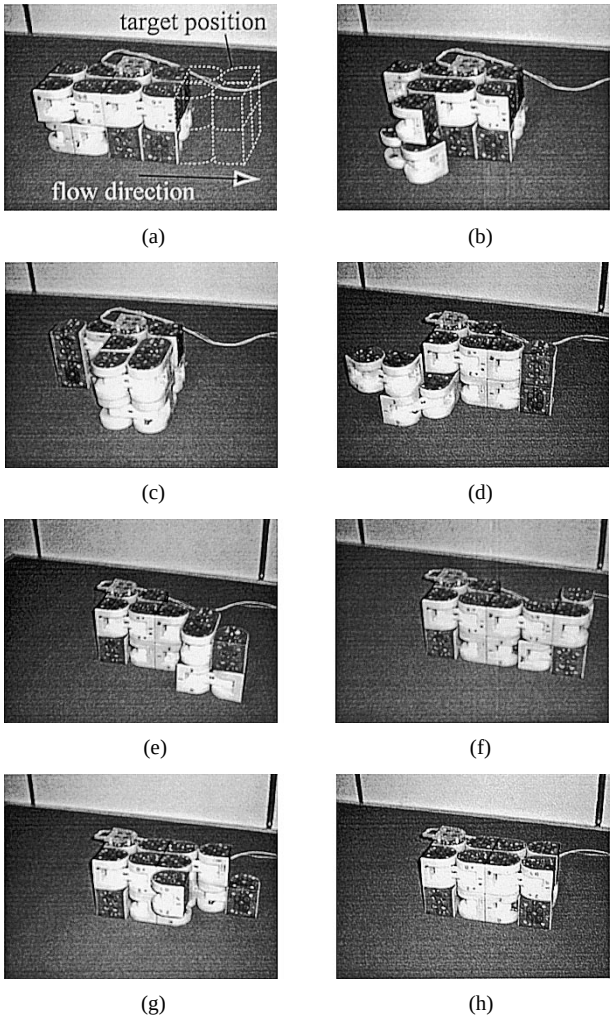

Fig. 20. Experiment of cluster motion of block structure (8 modules). Leftmost four modules are moving toward the rightmost target positions (indicated by white lines) along the "flow." (a) Initial state. (b) Step 4. (c) Step 8. (d) Step 14. (e) Step 17. (f) Step 18. (g) Step 21. (h) Final state.

A. Module Design

The most important issue in a SRS is designing the module. The structure of the module, especially allocation of actuated DOFs and connection surfaces, determines the overall system performance. As mentioned in Section I, many systems have been designed based on space-filling polyhedra, but there is no practical ground for doing this. Design based on cubic lattice seems to be good because the orthogonal axes ensure less interaction between the coordinates. Homogeneity or isotropy of the module is no doubt desired from the planning point of view, but it requires heavy connection mechanisms. We need to balance these contradicting constraints. In M-TRAN, we tried to

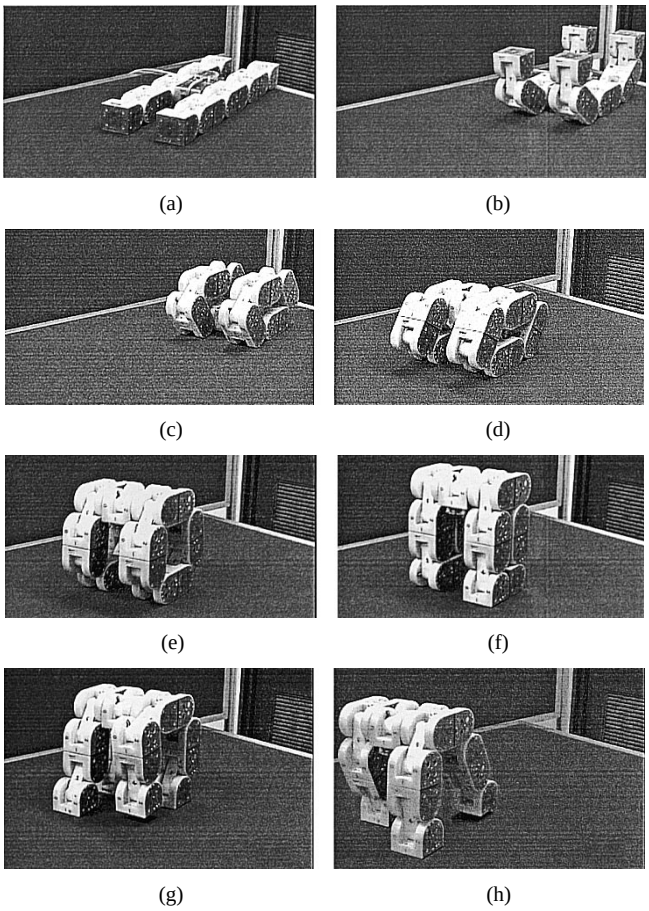

Fig. 21. Locomotion and transformation experiments (nine modules).

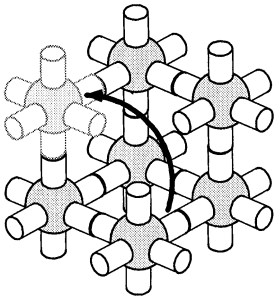

Fig. 22. 3-D SRS and reconfiguration motion.

simplify the module by sacrificing mobility. Basically speaking, M-TRAN's motion is restricted in a plane and multimodule cooperation is necessary to change the operational plane. It results in a feasible hardware platform for the self-reconfigurable robot, but motion planning becomes more difficult.

B. Efficiency

Here, we show that the double-cube construction of M-TRAN is more efficient than single-cube (or cellular) modules. To facilitate the discussion, we compare M-TRAN and the 3-D SRS [1]. In a self-reconfiguration process, an 3-D SRS module changes its position as shown in Fig. 22. Here, only one DOF out of six is used. In contrast, an M-TRAN module moves either by itself or with one neighbor module. To evaluate the efficiency of reconfiguration motion, we estimated how much torque is required to lift other modules.

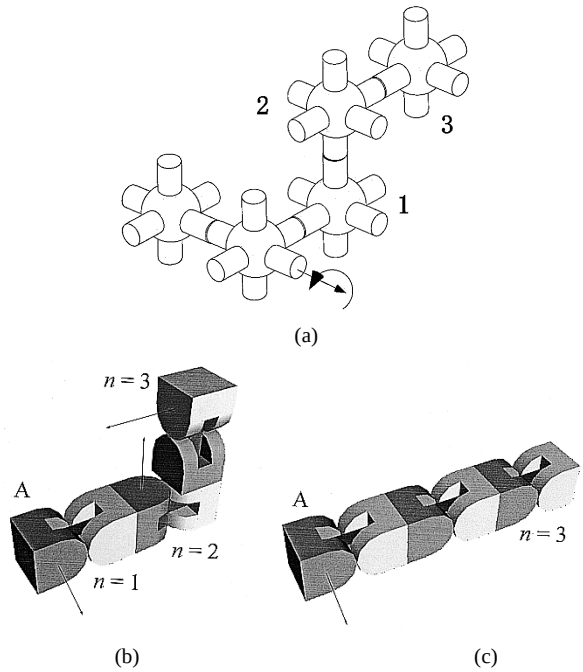

Fig. 23. Serial structures for lifting up: (a) 3-D SRD structure, (b) three-axis structure of M-TRAN, (c) worst case.

TABLE II
TORQUE FOR LIFTING UP (KG-CM)

	n	1	2	3
SRS-3D	(a)	11	21	42
M-TRAN	(b)	1.5	8.7	17
	(c)			22

k_m, k_v: 2.0, M_r: 120g, M_c:150g, M_o: 170g
V_r: 210cm^3, V_c: 68cm^3, V_o: 300cm^3

Suppose M-TRAN is made of three kinds of parts, i.e., rotation parts, connection parts, and the remainder. M-TRAN mass M_m and volume V_m are expressed as

$$M_m = M_r + M_c + M_o,$$
$$V_m = V_r + V_c + V_o,$$

where subscripts, r, c and o indicate rotation, connection and other parts respectively. If 3-D SRS is made of the same components, its mass and volume, M and V, will be

$$M = 3M_r + M_c + k_m M_o,$$
$$V = 3V_r + V_c + k_v V_o,$$

where k_m and k_v are parameters for adjustment. Those parameter values depend on the actual design, but could range from 1 to 3, because rotational parts become triple and connection parts are the same. In the following, both are set to 2.

Consider the situation in Fig. 23 where one actuator A is lifting several modules, where n is the number of lifted modules. In Fig. 23, structures (a) and (b) for $n = 3$ have three rotational axes and (c) is the hardest case of the M-TRAN. The required torque for each n is calculated for measured mass and volume of the M-TRAN hardware.

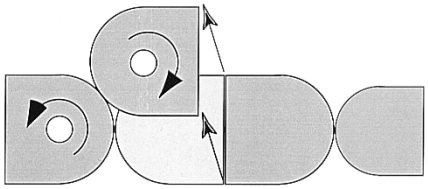

Fig. 24. Collision avoidance of M-TRAN module.

Table II shows that both 3-D SRS and M-TRAN can lift two modules within the actual torque limit (23 kg-cm) of the hardware. (Actual M-TRAN hardware can lift two modules at $n = 3$ in case (b) but not in case (c) because of the loss due to friction mainly caused by gear systems.)

During reconfiguration, motions like case $n = 1$ are always necessary for 3-D SRS, whereas M-TRAN moves sometimes by itself as $n = 1$ and sometimes as $n = 2$, both of which require much less torque. In this sense, M-TRAN has more margins for mechanical design.

C. Collision Avoidance and Connection Mechanism

In lattice-type self-reconfigurable systems, collision between the modules becomes a difficult design constraint. For instance in 3-D SRS and Molecule, the connection mechanisms must be retracted into the module during reconfiguration. Considerable space of the module is consumed for such retraction. In the M-TRAN module, even adjacent modules are connected by maximum area, but collision between the modules can be avoided by a motion such as shown in Fig. 24. Moreover, the M-TRAN connection mechanism while being thin and compact compared with other mechanisms, it provides enough connection force and connection speed, consumes no power once connected and can be quickly detached. It also absorbs some positioning errors when connecting modules.

D. Structural Strength

Modules based on space-filling polyhedra can form a regular lattice structure, but the structure itself contains large cavities between the modules. This means the mechanical strength of the overall structure depends on the strength of the connection mechanisms. M-TRAN can form a solid structure with its casings. The modules contact side-to-side in the cluster and thus have greater mechanical strength. However, there are two surfaces for each box without connection capability. Therefore, appropriate arrangement of modules is necessary to maintain connectivity and to obtain a firm structure.

E. Motion Planning

As mentioned in Section III-A, isotropic modules are easier to reconfigure but the M-TRAN module does not have such a property. It is, however, possible to design a complicated reconfiguration sequence between different configurations by using the interface software (Section III-B). Self-reconfiguration difficulty is also relaxed by limiting the structure to a certain class in M-TRAN (Section III-C). We need to accumulate many design examples and develop a method to combine appropriate parts from the examples to achieve a feasible solution for a given task.

VI. Concluding Remarks and Future Work

In this paper, we presented a novel design for a self-reconfigurable robotic system called M-TRAN. The hardware system and its software for self-reconfiguration were described and the systems feasibility was demonstrated through simulations and experimental results. It is realized by a simple module design and compact implementation of hardware.

There are several important issues remaining in the M-TRAN project. The first one is to improve the hardware system. Currently, we are developing the next version of the M-TRAN module, which is smaller and lighter, but more powerful in computation, communication and sensing. Batteries will be embedded to achieve a self-contained robot. The second issue is software for motion planning. Various learning and search methods must be tried to solve self-reconfiguration problems. The third issue is the control architecture. Our previous systems, fracta and 3-D SRS, are homogenous distributed systems without a central supervisor. In M-TRAN, we adopt central control by a host PC to realize coordinated synchronous motion. This, however, is not suitable for self-repair so we must investigate a distributed control algorithm for M-TRAN. The last issue is more general and may apply to all self-reconfigurable systems, that is, how to find or design the target configuration itself. In other words, we need to develop an algorithm to generate an optimal or near-optimal configuration for the given task or environment.

References

[1] S. Murata, E. Yoshida, H. Kurokawa, K. Tomita, and S. Kokaji, "Self-repairing mechanical systems," *Auton. Robot. 10*, pp. 7–21, 2001.

[2] A. Pamecha, C.-J. Chiang, D. Stein, and G. Chirikjian, "Design and implementation of metamorphic robots," presented at the ASME Design Eng. Tech. Conf. Computers Eng. Conf., Irvine, CA, 1996.

[3] K. Tomita, S. Murata, H. Kurokawa, E. Yoshida, and S. Kokaji, "Self-assembly and self-repair method for a distributed mechanical system," *IEEE Trans. Robot. Automat.*, vol. 15, pp. 1035–1045, Dec. 1999.

[4] D. Rus and M. Vona, "Crystalline robots: Self-reconfiguration with compressible unit modules," *Auton. Robot.*, vol. 10, pp. 107–124, 2001.

[5] C. Ünsal, H. Kiliççöte, and P. K. Khosla, "A modular self-reconfigurable bipartite robotic system: Implementation and motion planning," *Auton. Robot.*, vol. 10, pp. 23–40, 2001.

[6] M. Yim, Y. Zhang, J. Lamping, and E. Mao, "Distributed control for 3-D metamorphosis," *Auton. Robot.*, vol. 10, pp. 41–56, 2001.

[7] K. Kotay and D. Rus, "Locomotion versatility through self-reconfiguration," in *Robot. Auto. Syst.*, vol. 26, 1999, pp. 217–232.

[8] T. Fukuda and S. Nakagawa, "Approach to the dynamically reconfigurable robotic system," *J. Intell. Robot. Syst.*, pp. 55–72, 1988.

[9] M. Yim, D. Duff, and K. Roufas, "PolyBot: A modular reconfigurable robot," presented at the IEEE Int. Conf. Robotics and Automation, 2000.

[10] M. Yim, Y. Zhang, and D. Duff, "Modular robots," *IEEE Spectr.*, pp. 30–34, Feb. 2002.

[11] W.-M. Shen, B. Salemi, and P. Will, "Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots," *IEEE Trans. Robot. Automat.*, pp. 700–712, Oct. 2002.

[12] S. Murata, E. Yoshida, H. Kurokawa, K. Tomita, and S. Kokaji, "Concept of self-reconfigurable modular robotic system," *Artif. Intell. Eng.*, pp. 383–388, 15.

[13] H. Kurokawa, K. Tomita, E. Yoshida, S. Murata, and S. Kokaji, "Motion simulation of a modular robotic system," in *Proc. IEEE Int. Conf. Industrial Electronics, Control and Instrumentation (IECON-2000)*, Oct. 2000, pp. 2473–2478.

[14] E. Yoshida, S. Murata, A. Kamimura, K. Tomita, H. Kurokawa, and S. Kokaji, "A motion planning method for a self-reconfigurable modular robot," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS-2001)*, Oct. 2001, pp. 509–597.

[15] ——, "A self-reconfigurable modular robot: Locomotion planning and experiments," *Int. J. Robot. Res.*, submitted for publication.
[16] S. Hirose, M. Imazato, Y. Kudo, and Y. Umetani, "Internally-balanced magnetic unit," *Adv. Robotic.*, vol. 3, no. 1, pp. 225–242, 1986.
[17] G. Chirikjian, A. Pamecha, and I. Ebert-Uphoff, "Evaluating efficiency of self-reconfiguration in a class of modular robots," *J. Robot. Syst.*, vol. 13, no. 5, pp. 317–338, 1996.
[18] K. D. Kotay and D. Rus, "Scalable parallel algorithm for configuration planning for self-reconfiguring robots," in *Proc. Int. Society Optical Engineers (SPIE)*, vol. 4196, 2000, pp. 377–387.
[19] C.-J. Chiang and G. S. Chirikjian, "Modular robot motion planning using similarity metrics," *Auton. Robot.*, vol. 10, pp. 91–106, 2001.
[20] K. Miyashita and S. Kokaji, "Navigating modular robots in the face of heuristic depressions," in *Proc. Distributed Autonomous Robotic System 5 (DARS)*, 2002, pp. 7–26.

Satoshi Murata (M'01) received the B.E., M.E., and Dr. Eng. degrees in aeronautical engineering from Nagoya University, Nagoya, Japan, in 1984, 1986, and 1997, respectively.

In 1986, he joined the Mechanical Engineering Laboratory, Ministry of International Trade and Industry (MITI), Ibaraki, Japan. Since 2001, he has been an Associate Professor at the Tokyo Institute of Technology, Yokohama, Japan. His research interests include distributed mechanical systems, modular robotics, and graph automata.

Dr. Murata received the IEEE-IE Outstanding Paper Award and the Society of Instrument and Control Engineers (SICE) Outstanding Paper Award, in 1991 and 1996, respectively. He is a member of SICE, the Robotics Society of Japan (RSJ) and the Japan Society of Mechanical Engineering (JSME).

Eiichi Yoshida (S'94–M'96) received the M.E. and Dr.Eng. degree from the Graduate School of Engineering, University of Tokyo, Tokyo, Japan, in 1993 and 1996, respectively.

From 1990 to 1991, he was with the Department of Microtechnique, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. In 1996, he joined the Mechanical Engineering Laboratory, inistry of International Trade and Industry (MITI), Ibaraki, Japan. His research interests include decentralized autonomous systems and modular robotics.

Dr. Yoshida received the Best Paper Award at the 1998 International Symposium on Distributed Autonomous Robotic Systems (DARS '98).

Akiya Kamimura received the M.E. and Dr.Eng. degree from the Graduate School of Engineering, University of Tokyo, Tokyo, Japan, in 1997 and 2000, respectively.

In 2000, he joined the Mechanical Engineering Laboratory, Ministry of International Trade and Industry (MITI), Ibaraai, Japan. His research interests include modular robotics and rapid prototyping systems.

Dr. Kamimura received the Best Paper Award at the 2002 International Symposium on Distributed Autonomous Robotic Systems (DARS '02).

Haruhisa Kurokawa received the B.E. and M.E.degrees in precision machinery engineering and the Dr.Eng. degree in aeronautical and astronautical engineering from the University of Tokyo, Tokyo, Japan, in 1978, 1981, and 1997, respectively.

Since 2001, he has been with Intelligent Systems Institute, National Institute of Advanced Industrial Science and Technology (AIST), Ibaraki, Japan, where he is currently the Head of the Distributed System Design Research Group. His main research interests include control in space, kinematics of mechanisms, distributed autonomous systems, and nonlinear control.

Kohji Tomita (M'99–A'01)received the B.E., M.E., and Ph.D. degrees from the University of Tsukuba, Tsukuba, Japan, in 1988, 1990, and 1997, respectively.

In 1990, he joined the Mechanical Engineering Laboratory, Ministry of International Trade and Industry (MITI), Ibaraki, Japan. From 2000 to 2001, he was a Visiting Assistant Professor at Dartmouth College, Hanover, NH. Since 2001, he has been a Senior Research Scientist at the National Institute of Advanced Industrial Science and Technology (AIST). His research interests include modular robots, distributed software systems, and graph automata.

Shigeru Kokaji (M'00) received the B.E., M.E., and Dr.Eng. degrees in precision machinery engineering from the University of Tokyo, Tokyo, Japan, in 1970, 1972, and 1986, respectively.

Currently, he is the Deputy Director of the Intelligent Systems Institute, National Institute of Advanced Industrial Science and Technology (AIST), Ibaraka Japan. His research interests include distributed control of mechanical/robotic systems.