A Self-Reconfigurable Modular Robot (MTRAN) – Hardware and Motion Generation Software –

Akiya KAMIMURA[†], Eiichi YOSHIDA[†], Satoshi MURATA[‡], Haruhisa KUROKAWA[†], Kohji TOMITA[†] and Shigeru KOKAJI[†]

[†]National Institute of Advanced Industrial Science and Technology (AIST): Namiki 1-2-1, Tsukuba, Ibaraki, 305-8564 Japan, kamimura.a@aist.go.jp
[‡]Tokyo Institute of Technology: 4259 Nagatsuta-cho, Midori-ku, Yokohama, 226-8502 Japan, murata@dis.titech.ac.jp

Abstract. In this paper we present our latest modular robot "MTRAN"(Modular Transformer) and its motion generation algorithm based on a four-module block method. The developed module is composed of two semi-cylindrical parts connected by a link. Each part has three connection surfaces where another module can be connected by magnetic force. Each module has only two degrees of freedom, but a group of modules can not only configure static 2-D or 3-D structures but also generate robotic motions. We realized several robotic motions and transformation by hardware experiments. Block motions made by proposed motion generator are also verified by experiments.

Key Words. Modular Robot, Self-Reconfiguration, Robotic Motion Generation, Motion Planning

1 Introduction

In recent years, the feasibility of reconfigurable robotic systems has been examined through hardware and software experiments [1]–[8]. Self-reconfigurable robots, especially homogeneous ones, can adapt themselves to the external environment by changing their configuration and repair themselves by using spare modules. This type of robot is useful in harsh environments where adaptability and self-maintainability are significant factors.

Reconfigurable modular robots are classified into two types, lattice type [2,4,8] and thread type [6,7]. While the former can transform itself into various static structures like jungle-gym, it is difficult to generate dynamic robotic motions. On the other hand, the latter has snake-like shape that can generate various dynamic motions though it has difficulty in self-reconfiguration.

We have developed a new type of modular robot, called *Modular Transformer* or *MTRAN* [9], which can realize both static structure and dynamic motion. This

has been achieved by simplified design of a module and lightweight and firm connecting mechanism. The paper outlines the hardware and control system of MTRAN and demonstrates experiments of many-unit motions. We succeeded in making the modular robot change its locomotion mode through self-reconfiguration. To our knowledge, these are the world heading results.

As to a motion generation method, there have been studies on distributed [10,11] and centralized [4,8] methods. However, these methods cannot directly be applied to MTRAN due to its restricted degrees of freedom. The latter half of this paper describes a motion generation method to enable a class of MTRAN clusters to move along a desired trajectory in a series of reconfiguration. The motion generator consists of a planner and a scheduler and generates an efficient module motion plan where several modules are driven in parallel.

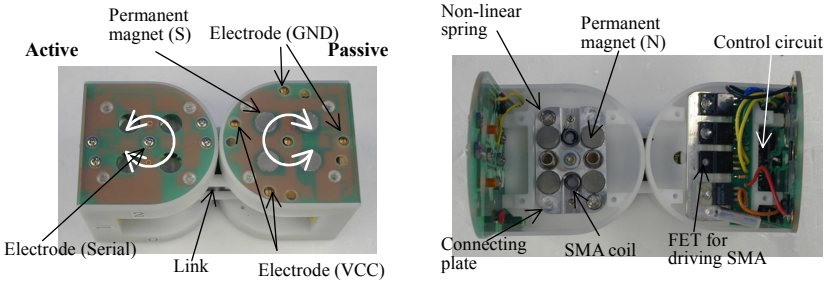


Fig. 1. Photos of appearance and inner structure of the developed module

2 Hardware

We have developed a module hardware that has a simple structure shown in Fig.1. The module is composed of two semi-cylindrical parts connected by a link. Each semi-cylindrical part can rotate about its axis by 180 degrees with a servomotor embedded in the link. Each semi-cylindrical part has three connecting surfaces with four permanent magnets and can connect to other modules' surfaces that have the opposite polarity of the magnets. The two semi-cylindrical parts have surfaces of different polarity. If we call these as an A-type (for 'active') part and P-type (for 'passive', see Fig.1) part, A and P-type parts occupy disjoint positions in 3-D space. A-type subspace and P-type subspace form 3-D checkerboard-like structure. Reconfiguration is achieved by repeating basic operations such as detaching, rotating and reconnecting.

The advantages of this module over our previous modules [1,2,3] are as follows.

1. Both 3-D static structures and dynamic motions are available.
2. Wires for power supply and communication are decreased.

3. Downsizing is realized by adopting the simple structure.
4. The connecting mechanism using permanent magnets is simple, lightweight and reliable, which also has a capability of self-alignment.

Table 1. Specification of the module

Item	Value
Dimension	66x132x66 mm
Weight	0.44 kg
CPU	BASIC STAMP II
Power supply	DC 12V
Maximum torque of each axis	23 kg·cm
Connecting force	25 N
Elapsed time for detachment	5 seconds
Power consumption for detachment	180 J
Electrical resistance of module	1.3 Ω

2.1 Mechanical and Electrical Design

The specification of the module is summarized in Table 1.

The link part includes two sets of geared motors and servo circuits. The maximum torque of each servo is enough to lift up one module as shown in Fig.2.

On all the connection surfaces, there are electrodes for power supply and serial communication. Therefore it is enough to connect wires to a single module to supply power and to communicate with all the modules.

There are two parts, called active and passive part as shown in Fig.1. Inside the active part, there is a connection mechanism that is composed of non-linear springs, SMA (Shape Memory Alloy) coils and four permanent magnets (N poles on the surface), which are fixed on the connecting plate. This mechanism is based on the technique of IBMU (Internally Balanced Magnet Unit [12].) The connecting plate moves automatically to connect to the other modules' surface by attractive force of magnets. It is detached by heating SMA coils by electric current. When detaching, the springs help SMA coils [9].

Inside the passive part, there are control circuits including an onboard microcomputer BASIC STAMP II (BS II, Parallax, Inc.) It controls motor rotation and heating of SMA coils. Control commands corresponding to these operations are issued by the host computer through the serial communication line as shown in Fig.3.

2.2 Control System Architecture

The system consists of host PC, relay BS II (the same one as the onboard microcomputer) and modules, and they are connected as shown in Fig.3. Communication is asynchronous (RS-232C) and based on token passing.

The host PC broadcasts a control command with a module's ID number. Only one module of this ID executes the task and sends back a validation signal to the host.

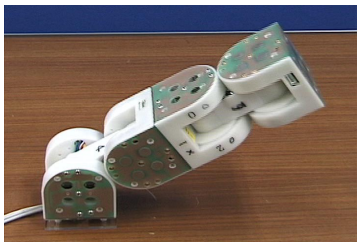


Fig. 2. Lifting up motion with two modules

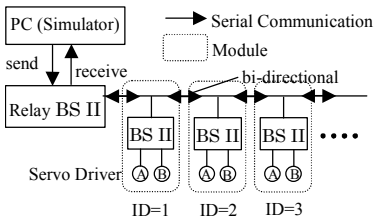


Fig. 3. Schematic of control system architecture

2.3 Motion Planning Simulator / Controller

We developed a simulator for motion planning (Fig.4). The process of motion planning is as follows [13]: First, initial configuration of modules is created by using a configuration editor control panel (not shown). Next, the module motion sequence is programmed manually by using the motion control panel in Fig.4. The generated motions are recorded and displayed in motion language. Collisions between modules and connectivity of the whole configuration are automatically checked. In the current simulator, quasi-static motion is implemented by taking gravity and sliding friction into account. Finally, the sequence is translated into a series of hardware control commands, which are sent to the module hardware via the serial line.

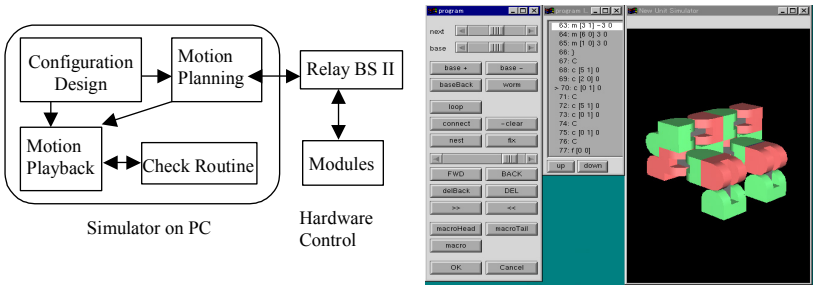


Fig. 4. Diagram of simulator (left) and screen capture (right)

3 Motion Planning Method

The motion generator was developed for a particular class of module clusters (Fig.5). The cluster is a variable-length chain composed of four-module *blocks*

that look like large cubes. In addition, a couple of modules called a converter that have different direction of rotation axes are attached to change the direction of rotation axes of modules in the chain cluster.

The goal of this motion generator is to let the cluster move along a certain given three-dimensional trajectory in the lattice grid (Fig.6). The generator outputs a series of reconfigurations, in each of which one block is transferred to another place.

In our previous paper [14], the motion of a cluster was generated by a two-layered *motion planner*. The upper layer decomposes the planning problem into subproblems solvable by the lower layer. The lower layer is designed to solve simplified planning problems based on a database of rules for each local reconfiguration.

As the generated motion allows only one module to move at a single step we introduce motion parallelism by a *motion scheduler*. It processes the planned sequence into a motion plan including motion steps that can be executed in parallel. This makes use of the concurrent feature of the modular robot and increases the efficiency by reducing the total time required for the plan.

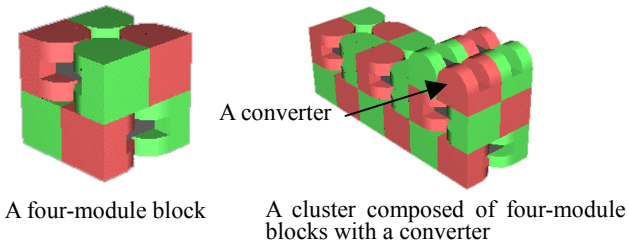


Fig. 5. A four-module block and a cluster

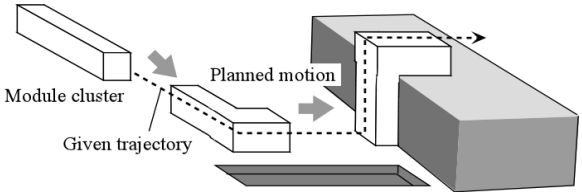


Fig. 6. Planning of cluster motion

3.1 Motion Planner Architecture

This section briefly outlines the motion planner [14]. The upper and lower layers of the motion planner are called the *global flow planner* and the *local motion scheme selector* respectively. As shown in Fig.7, the global flow planner searches possible module paths and motion orders to provide the global cluster movement,

called *flow*, according to the desired trajectory. This is realized as a motion of a block such that the tail block is transferred toward the given heading direction via the side of the cluster. We adopt simple motion schemes sending modules one by one towards the head.

The local motion scheme selector checks if the paths generated by the global planner are valid for each *member* module of the block. If a given path turns out to be valid, the selector chooses the motion plan by adding a set of locally coordinated motion sequences called *motion schemes* from a rule database. Otherwise it tries another possible path. Note that this is a centralized planning method assuming that all the information of modules in the cluster is available.

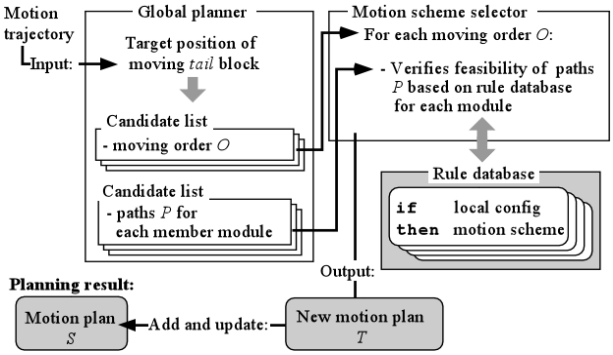


Fig. 7. Motion planner architecture

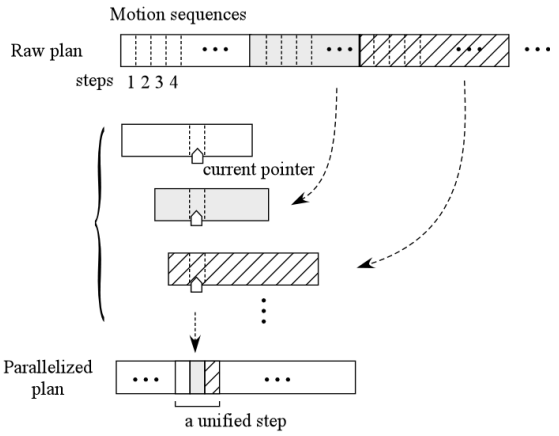


Fig. 8. Parallelization of plan by motion scheduler

3.2 Motion Scheduler

The output of the planner described so far is a sequence of motion schemes to achieve the desired trajectory. However, only one motion scheme is allowed at a

time. The motion scheduler is devised to improve the efficiency through parallel execution of multiple motion schemes.

3.2.1 Parallelizing a motion plan

The output plan by the motion planner is a series of motion sequences each of which corresponds to a single path of a module from tail to head. The motion scheduler tries to parallelize the plan by interleaving a motion sequence with following ones (Fig.8). During this process, collisions and total connectivity are checked. Also, it is checked whether the parallelized motion plan end up with the same configuration as the original plan. Those motion steps executable in parallel are unified into one motion step. By repeating these procedures throughout the original plan, a parallelized motion sequence is derived.

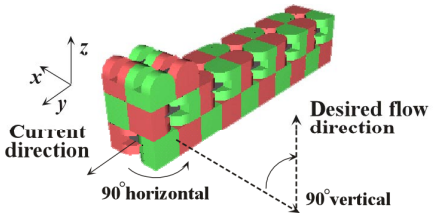


Fig. 9. Desired trajectory of module cluster

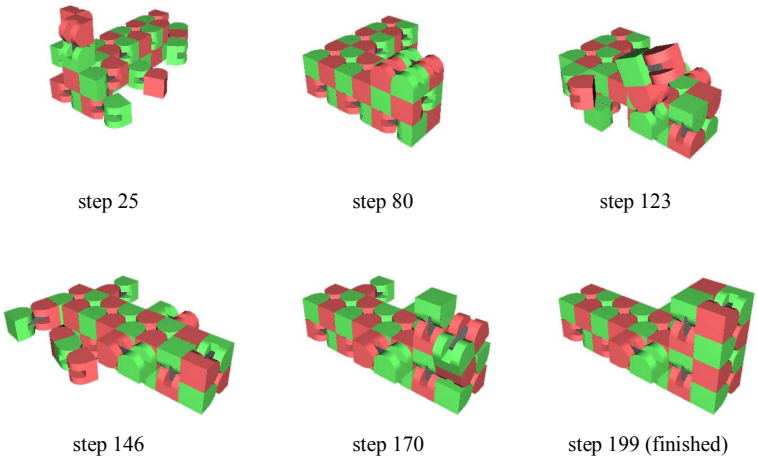


Fig. 10. Generated plan for desired motion in Fig.9

3.2.2 Generated motion

The motion generation framework described so far is applied to a module cluster composed of 22 modules. The desired trajectory includes horizontal and vertical direction changes of cluster flow as shown in Fig.9. Fig.10 shows some snapshots taken from the generated motion.

The raw plan generated by the motion planner takes 354 motion steps, where only one motion scheme is allowed at one step. After rescheduling, the length was reduced down to 199 steps.

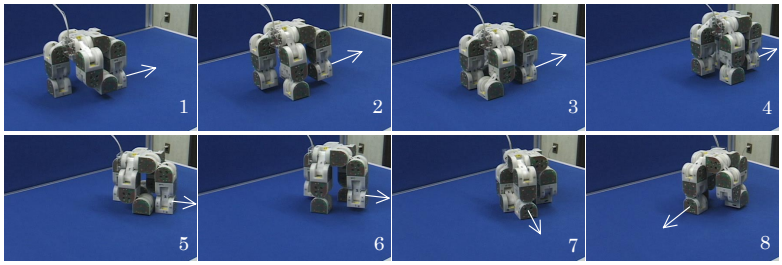
4 Hardware Experiments

To verify the module hardware and the motion planning method, we carried out several experiments.

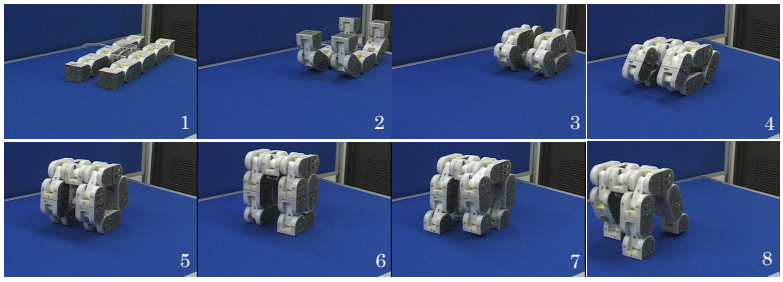
4.1 Experiment of Robotic Motion and Transformation

Fig.11 (a) shows locomotion of quadruped type robot using 8 modules. This robot can walk by using two of the leg parts, and turn about its vertical center axis by folding two leg parts in opposite directions. Arrows in photos indicate robot's moving direction.

Fig.11 (b) shows the transformation and locomotion experiment using 9 modules. This robot is transformed from 2-D structure into a crawler, and then to a quadruped robot by releasing the closed module rings to configure legs.



(a) Locomotion of quadruped type robot (8 modules)



(b) Transformation from 2-D static structure to crawler robot and then to quadruped robot (9 modules)

Fig. 11. Robotic motion and transformation experiments

4.2 Experiment of Block Motion

Fig.12 shows the experiment of eight-module cluster flow motion. The output plan by the generator was slightly modified to avoid the current hardware problem.

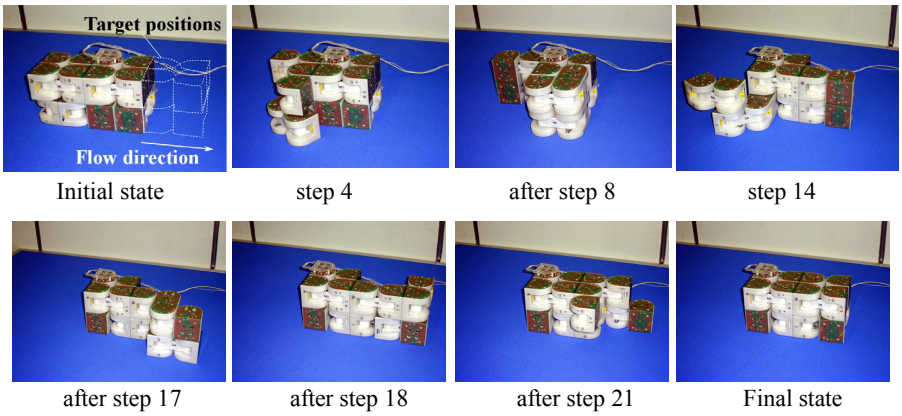


Fig. 12. Experiment of cluster motion of block structure using 8 modules

5 Future Works

The current hardware has following problems to be solved.

1. No sensors are installed for internal and external sensing.
2. Local communication between modules is not realized.
3. Large power is consumed at SMA coils while detaching process.
4. Cables for power supply and serial communication must be attached to one of the modules.
5. Electrodes at the active part are protruded from the surface and they cause a short circuit in some cases.

And also the following software algorithms are required.

1. A general algorithm that schedules each modules' motion for a transformation from one structure to another.
2. An algorithm that generates various module structures suited for locomotion or tasks together with movement sequences of each module.
3. A distributed algorithm that enables each module to move autonomously to realize robotic motions in cooperative manner. In order to realize this, each module must recognize own state, states of neighbor modules and external states using sensor information.

6 Conclusions

In this paper, we described the hardware of the proposed module, the control system architecture and the motion generation algorithm. With those modules, it is possible to build a variety of configurations and to generate transformation among the configurations. The motion generator is composed of planner and scheduler and it improved parallelism of the modular system.

References

1. S. Murata, et al.: "Self-assembling machine, " *Proc. IEEE Int. Conf. on Robotics and Automation*, 441-448, 1994.
2. S. Murata, et al.: "A 3-D self-reconfigurable structure," *Proc. IEEE Int. Conf. on Robotics and Automation*, 432–439, 1998.
3. E. Yoshida, et al.: "Micro self-reconfigurable robotic system using shape memory alloy," *Distributed Autonomous Robotic Systems 2000*, 145-154, 2000.
4. K. Kotay and D. Rus: "Motion synthesis for the self-reconfigurable molecule," *Proc.1998 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 843–851, 1998.
5. P. Will, et al. : "Robot modularity for self-reconfiguration," *Proc. SPIE, Sensor Fusion and Decentralized Control in Robotic Systems II*, 236–245, 1999.
6. A. Casal and M. Yim: "Self-reconfiguration planning for a class of modular robots," *Proc. SPIE, Sensor Fusion and Decentralized Control in Robotic Systems II*, 246–257, 1999.
7. A. Castano, et al.: "Autonomous and self-sufficient CONRO modules for reconfigurable robots," *Distributed Autonomous Robotic Systems 4*, Parker L E, et al. eds., Springer, 155–164.
8. C. Ünsal, et al.: "A modular self-reconfigurable bipartite robotic system: implementation and motion planning," *Autonomous Robots*, 10-1, 23–40, 2001.
9. A. Kamimura, et al.: "Self-reconfigurable Modular Robot – Experiment on reconfiguration and locomotion," *Intelligent Robots and Systems (IROS2001) Proc. 2001 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2001)*, 606–612, 2001.
10. E. Yoshida, et al.: "A distributed method for reconfiguration of 3-D homogeneous structure," *Advanced Robotics*, 13-4, 363–380, 1999.
11. K. Tomita, et al.: "Self-assembly and self-repair method for distributed mechanical system," *IEEE Trans. on Robotics and Automation*, 15-6, 1035–1045, 1999.
12. S. Hirose, et al.: "Internally-Balanced Magnet Unit," *Advanced Robotics*, 1-3, 225-242, 1986.
13. H. Kurokawa, et al.: "Motion Simulation of a Modular Robotic System," *Proc. IECON 2000*, 6, 2000 (in CD-ROM).
14. E. Yoshida, et al.: "A Motion Planning Method for a Self-Reconfigurable Modular Robot," *Intelligent Robots and Systems (IROS2001) Proc. 2001 IEEE/RSJ Int. Conf. On Intelligent Robots and Systems (IROS2001)*, 590–597, 2001.