Self-Reconfigurable Modular Robot M-TRAN: Distributed Control and Communication

Haruhisa Kurokawa, Kohji Tomita, Akiya Kamimura, Shigeru Kokaji, Takashi Hasuo, and Satoshi Murata

***Abstract*—M-TRAN is a self-reconfigurable modular robot: each module has an independent battery, two-degree-of-freedom motion, six-surface-connection capability, and intelligence with inter-module communication. The M-TRAN system can perform flexible and adaptive locomotion in various configurations using coordination control based on a central pattern generator (CPG). Various structures of several modules can perform metamorphosis, such as that between a four-legged robot and a snake-like one. In addition to these self-reconfigurations with synchronous control, M-TRAN structures having regularity can move using parallel distributed control and message exchange via the network bus. Self-reconfiguration using infrared local communication has been attempted to improve the system's scalability.**

***Index Terms*—Central pattern generator, Coordinated motion control, Distributed autonomous system**

I. Introduction

Several modular robotic systems that can form various structures have been proposed. A module functions as an autonomous building block like an artificial cell, which in most cases has independent power, actuation and connection capability, and intelligence with inter-module communication. Most proposed modular robotic systems are classifiable as one of two types: a lattice type [1–5] or a chain type [6–9].

Chain-type and lattice type modular robots differ in their design and research objectives. A chain-type modular robot is suitable for locomotion. Manual connection of modules is sufficient, as has been the case with most systems. Rapid long distance communication facilitates motion coordination, especially for dynamic walking, although local communication is used in some cases because rapid communication is unnecessary for static locomotion such as snake-like, caterpillar-like, or crawler motion. A candidate communication method is a network bus system of a multi-computer system, such as Ethernet, CAN, wireless LAN, and Bluetooth, which functions similarly to a living creature's nervous system [6, 8]. On the other hand, a lattice-type module is advantageous for self-reconfiguration. Automatic connection of modules is indispensable. Rapid long distance communication is not always necessary, depending on the algorithm used.

A research objective of a self-reconfigurable modular robot is to realize self-organization of a distributed autonomous system. It is expected that self-reconfiguration and self-assembly based on self-organization can render the system more flexible, reliable, scalable, and adaptive than those using conventional top-down and model-based design approaches. Our Fractum [1] was aimed at autonomous shape formation based on a scalar field and diffusion dynamics. A similar line was sought with discrete dynamics (algorithms) using modules such as Molecule and Cristalline [10]. SlimeBot [11] uses coupled nonlinear oscillators to produce an adaptive coordinated motion of a module cluster. All of those studies were intended for realization of global functions based on local coordinate-free dynamics.

Some preliminary experiments involving the above systems, which are mostly two-dimensional (2D), have been successful. Difficulties arise in design and hardware realization of a three-dimensional (3D) system. Required actuation degrees of freedom tend to increase and the effect of gravity becomes severer for 3D systems than with 2D ones. Several designs such as ATRON [3] have been proposed to reduce modules' symmetry; various trials have been made to relax constraints and difficulties, e.g., operation in a fluid or in space. However, 3D self-reconfiguration remains a challenge.

The design of our modular robot, Modular TRANsformer (M-TRAN), is intended to provide advantages of both chain type and lattice type systems [12]. Because of this hybrid design, the M-TRAN system can perform both locomotion and self-reconfiguration. For locomotion, we have proposed a general design method of robust locomotion for various configurations, which uses central pattern generators (CPGs) for coordinated motion control and a genetic algorithm (GA) for CPG network optimization. The real-time CPG controller is a robust controller that is adaptive to its environment [13].

On the other hand, self-reconfiguration using the M-TRAN system is less simple because an M-TRAN module has less symmetry than other lattice-type systems. Despite this fact, we have developed various programs for self-reconfiguration design, simulation, and controllers. Subsequently, results of experiments of small-scale metamorphoses (self-reconfiguration) verified the outstanding capabilities of the system [14]. In addition, we have proposed a

Haruhisa Kurokawa, Kohji Tomita, Akiya Kamimura and Shigeru Kokaji are with the Intelligent Systems Research Institute at the National Institute of Advanced Industrial Science and Technology (AIST), Tsukuba, Japan (e-mail: {kurokawa-h, k.tomita, kamimura.a, s.kokaji} @aist.go.jp).

Takashi Hasuo is with the Graduate School of Systems and Information Engineering at the University of Tsukuba, Tsukuba, Japan (e-mail: hasuo@bcl.esys.tsukuba.ac.jp).

Satoshi Murata is with the Interdisciplinary Graduate School of Science and Engineering at Tokyo Institute of Technology, Yokohama, Japan (e-mail: murata@dis.titech.ac.jp).

ROBOCOMM 2007, 14th – 16th Oct 2007, Athens, Greece.

ISBN 978-963-9799-08-0
DOI 10.4108/ICST.ROBOCOMM2007.2119

method using module clusters, so-called meta-modules, for large-scale self-reconfiguration and for distributed control [15, 16]; we also carried out experiments [17]. Because the system's reliability increased and because various software tools were developed, experiments of large-scale parallel distributed self-reconfiguration have become more successful. (http://unit.aist.go.jp/is/dsysd/mtran3/ for movies.)

The control of these experiments is still open-loop, but it is a basis for development of a self-organizing autonomous system. For inter-module communication, a wired network has been used, because circuit installation and programming were much easier than with a local communication method at the time of prototype design. A network bus system with ID numbering, however, is unsuitable for system scalability and parallelism. We have developed local communication among neighbors using infrared devices as a preliminary test; using it, we have started experiments of distributed self-reconfiguration.

Basic design and hardware development of M-TRAN prototypes are described in Section II. In Sections III and IV, experiments of locomotion and self-reconfiguration by M-TRAN II and III are presented. Basic motion procedures, algorithms and communication protocols are detailed. Section V concludes the paper with descriptions of future works.

II. M-TRAN System

A. Basic Design

An M-TRAN module comprises two blocks and a link (Figs. 1 and 2). Both blocks have identical shape, comprising a half-cube and a half-cylinder. Two blocks can rotate independently about their axes by ±90 deg. At any angle, their half-cylindrical parts always have mutual physical contact.

The two blocks differ in gender: opposite gender blocks of two modules can connect mechanically with a counterpart by their three flat surfaces. Although the three surfaces' shapes are not identical, they can all connect symmetrically in four possible orientations. Using this flexibility of connection, various structures that are suitable for locomotion can be constructed as a chain-type modular robot.

As a lattice-type modular robot, the M-TRAN structure can self-reconfigure by controlling the module's position and connection. All blocks align with a cubic lattice when all the modules' angles in a structure are restricted to 0 or ±90 deg, which ensures that open-loop positioning control of each module is sufficient to align a pair of surfaces for reconnection.

B. Mechanical Design

For self-reconfiguration, an M-TRAN module must fit in two cubic blocks so that mechanisms, circuits, and wires are all installed inside the three block parts, as shown in Fig. 1. The passive (female) block contains a power circuit board, a battery and a controller board. The link block contains two geared motors and their controller circuit. The active (male) block contains three mechanisms for surface connection and their controller circuit.

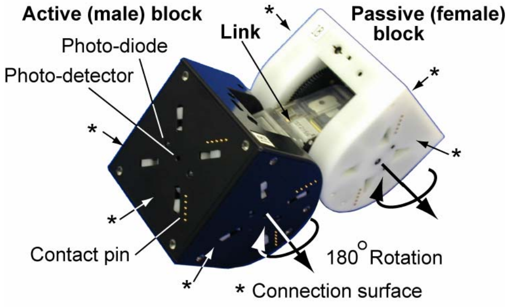


Fig. 1 M-TRAN module

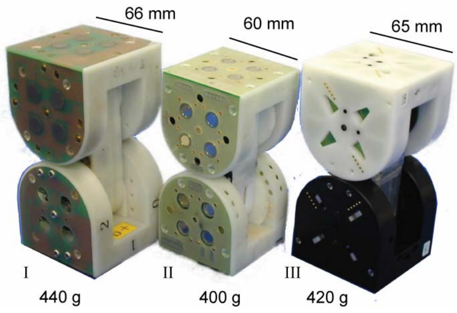


Fig. 2 Three generations of M-TRAN

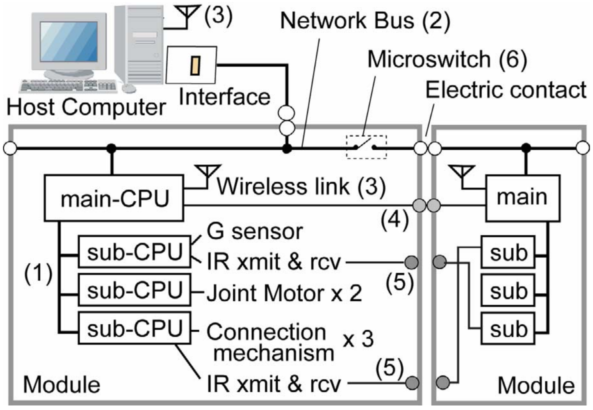


Fig. 3 Computer and communication system of M-TRAN

The connection mechanism was greatly improved by M-TRAN III. Instead of magnetic connection used of the two preceding prototypes, M-TRAN I and II, M-TRAN III uses mechanical connection, which is much faster, more reliable, and less power-consuming [17].

C. Controller and Communication Design

Figure 3 shows controller diagrams of M-TRAN II and III; Table I shows specifications of controllers and communication media. The latest prototype, the M-TRAN III system, has five different communication channels.

The first, not shown in Table I, is a three-wire serial channel (send, receive and ground; 19.2 kbps) among the main and the three sub CPUs inside a single module (Fig. 3(1)). The messages on this channel are commands of up to four bytes to one of the three sub CPUs for position control and connection

TABLE I
Microcontroller and communication medium of M-TRAN Series

	main-CPU	SUB-CPU	Communication
I	Basic Stamp II (Parallax, Inc.)	---	three wire serial with PC
II-1	Neuron Chip (TMPN3120FE 5M; Echelon Corp.)	PIC(PIC16F873; Microchip Technology)	Lonbus (RS485, 39 Kbps) + Wireless modem
II-2	SH II (HD64F7047; Renesas Technology Corp.)	PIC	CAN (1 Mbps) + Wireless modem
III	SH II	H8 (HD64F3687; Renesas Technology Corp.)	CAN (1 Mbps) + Blue tooth modem + IR serial (333 bps)

control, and sensing data. The sub CPUs work as intelligent peripheral devices of the main CPU, some of whose tasks are as follows. The sub CPU in the link controls two joint angles using either PID control, bang-bang control with braking by flywheel diodes, or an open loop control. Another one in the active block drives hooks for surface connection. That in the passive block acquires three-axis acceleration data for gravity direction detection. The sub CPUs in two blocks perform infrared (IR) transmission and detection, either for proximity sensing or for local communication.

For inter-module communication, a network bus, the controller area network (CAN) bus for M-TRAN III, is used (Fig. 3(2)). In addition, each module has a Bluetooth device (Fig. 3(3)). Although the standard Bluetooth covers network communication called Piconet, M-TRAN III only equips one-to-one communication capability. This channel, therefore, is used mainly for command transfer from the host PC, except in experiments of docking by two separate module clusters [18].

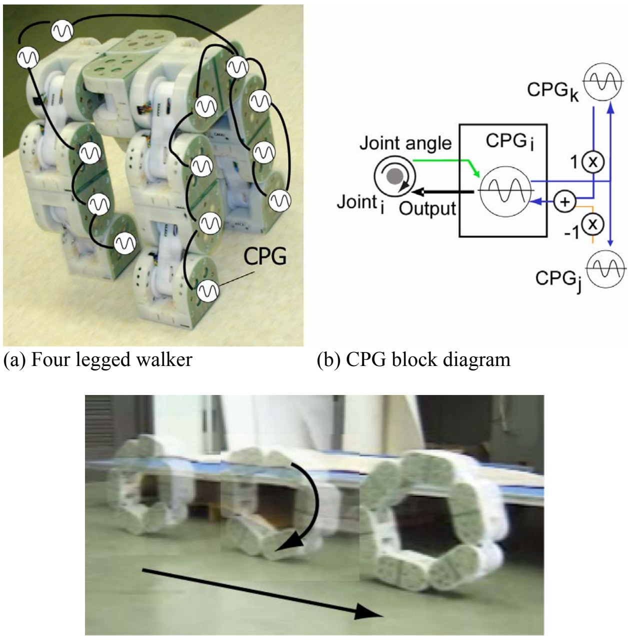
The remaining two communication channels are between two facing surfaces. One is unidirectional and is used for confirmation of connection (Fig. 3(4)). The other uses infrared light (Fig. 3(5)). Both are described below.

III. Locomotion and Metamorphosis

A. Locomotion

Various structures were designed; locomotion experiments were carried out using either feed-forward or feedback control. Feed-forward control using a lookup table is simple but sufficient for locomotion in various configurations such as snake-like or caterpillar-like ones or a crawler because such locomotion is stable and effective, even with some disturbance.

In case of dynamic walking, however, coordination and feedback of the whole body motion are necessary. The controller used for M-TRAN II is based on the central pattern generator (CPG) concept; CPGs are nonlinear neural oscillators observed in living creatures. We used a fourth-order oscillator that directly drives one joint motor. Local PID control is switched off but reading of the joint angle is fed back to the CPG dynamics so that the angle is regulated by a larger control loop (Fig. 4(b)). Two state variables of each CPG are input to other CPGs with weights of either 0, 1, or -1. Proper selection of the network weights realizes coordinated motion suitable for walking.



(a) Four legged walker (b) CPG block diagram

(c) Rolling wheel

Fig. 4 Locomotion using CPG control

In the experiments, each module's main CPU calculates dynamics of two CPGs; two state variables are shared via network communication called "remote frame" function of CAN. Because the maximum number of modules in a configuration for locomotion is 20 and because the CPU (SH-II) has sufficient message slots (8-byte buffers called mailboxes), 8-byte information (two 4-byte state variables in our case) of each module is stored in each specific slot and is periodically broadcast to all other modules by a background process (remote frame transfer). Consequently, a user program of each module can access any other modules' state variables without considering message exchanging via the CAN bus.

This network mechanism and CPG locomotion control were successful in experiments by M-TRAN II, such as four-legged walking adaptive to variation of ground friction and inclination, and a rolling wheel regulated by global entrainment in Fig. 4(c) [13].

B. Metamorphosis

Self-reconfiguration of M-TRAN I was controlled using a host PC via a serial communication line [12]. For self-reconfiguration of M-TRAN II, distributed but globally synchronous control was used [14].

A homogeneous system was sought for M-TRAN III, although self-reconfiguration of a few modules, such as those depicted in Fig. 5, was still made using centralized coordination control. At the startup, all modules have identical programs and data except for their identification numbers (IDs). The program uses fixed IDs, which are specified by each configuration. The actual IDs were different when the structure was assembled with arbitrary modules in arbitrary orientations. In the startup process, all modules identify the total configuration and change their ID numbers accordingly. Then one of the modules conducts a self-reconfiguration procedure as a master and others follow it as slaves [17].

The self-reconfiguration procedure shown in Fig. 5 is almost identical to that produced by M-TRAN II [14]. Because the connection mechanism has improved and because communication has become more reliable, as described below, the whole process has become much faster (8 min with M-TRAN II and 2 min with M-TRAN III) and much reliable. In addition, a reversal metamorphosis from a linear form to a four-legged robot has become possible.

IV. Parallel Self-Reconfiguration

A. Experiments

Parallel control is advantageous when the number of modules is large. We designed suitable self-reconfiguration procedures for parallel distributed control with three types of regular structures [15, 16]. The first type, type I, is a linear sequence of 4*n* modules, as portrayed in Fig. 6. The whole structure can move straight with pair-wise motions and switching of connections. A pair of modules on one side moves and steps forward by one module length. During this step motion, it is necessary to change connections among six surface pairs. Then another pair on the other side starts motion. Each pair synchronizes with its neighbors to avoid collision and separation of the whole structure.

Several different pair-wise motions were designed and tested: two are shown in Fig. 6. We made experiments using from 4 (minimal number) to 20 modules, all with the same programs running in all modules.

Type III in [16] is a 2D regular structure made of four module cross-shape building blocks (Fig. 7). Pairs of modules on one end ascend independently onto the backbone structure surface, travel along the surface, and finally descend to become part of the structure again. Other pairs follow them at a safe distance. The same program in all modules makes the rectangle arrangement of building blocks in any size move straight. The maximum rectangle in the experiments was of three lines by two block length; consequently, it comprises 24 modules. The whole structure moved about 400 mm in 16 min; connections between surfaces were changed more than 300 times. Consequently, the reliability of the total M-TRAN III system was verified. The following are details of basic procedures and algorithms used for the experiments.

B. Reliable Connection

The joint angle control of M-TRAN is a local bang-bang and/or PID control using angle sensors in modules. Because no sensor measures the alignment between two modules' surfaces,

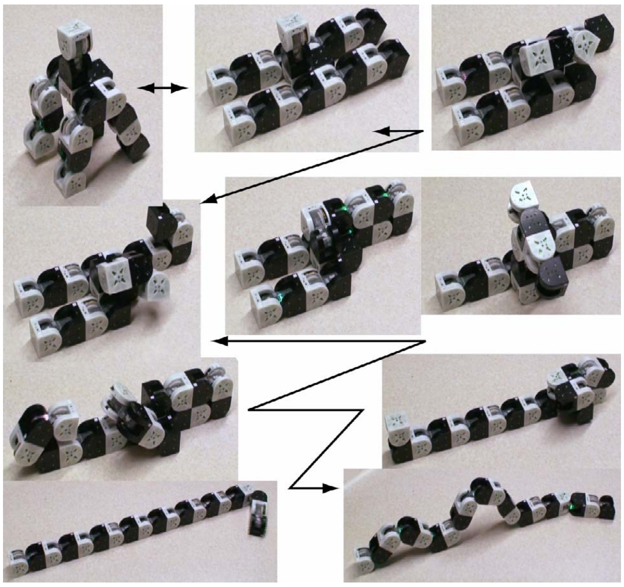
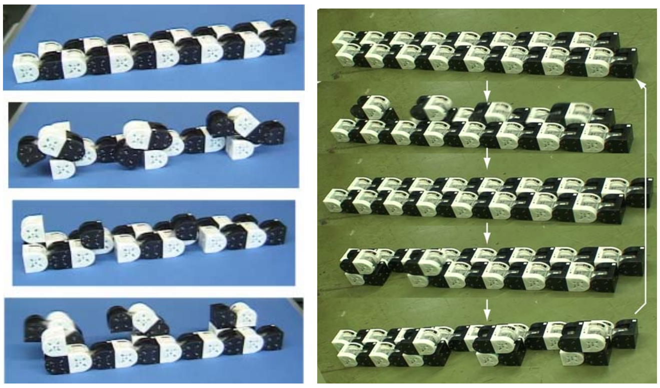


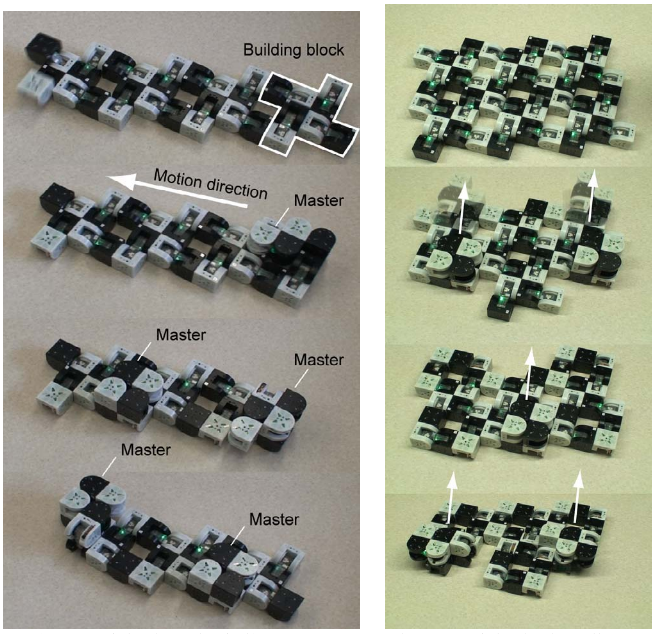
Fig. 5 Metamorphosis using centralized control



(a) Eight modules (b) 16 modules

Fig. 6 Linear structure

A pair-wise motion resembles rolling in (a) and an inch-worm motion in (b).



(a) 16 modules in a single line (b) 24 modules in three lines

Fig. 7 Self-reconfiguration of a mesh structure

feedback control cannot be made to align surfaces before their connection. Successful connection depends on the balance between open-loop positioning accuracy, geometric and mechanical precision of modules, and the misalignment margin of the connection mechanism. Misalignment depends on the configuration and posture: it is greater with a long serial chain configuration or overhanging posture. Self-reconfiguration sequences were therefore designed to reduce misalignment.

For confirmation and for retrial in cases of failure, completion of surface connections is checked as follows. Of the two modules to be connected, the male block starts the process: its connection hooks come out of its surface and stop at their respective limits. This process proceeds and finishes even if there is no female surface or if a connection is unsuccessful. After that, the male block sends a message containing its ID number via a pair of electric contact pins on the surface (Fig. 1 and Fig. 3(4)) and awaits an answer from the counterpart via the CAN bus. Because the contact pins are connected only when the connection is successful, both modules can confirm a successful connection using this message exchange and share information such as the neighbor's ID number and orientation. In case of failure, the connection hooks are retracted and the same process is repeated.

C. *Motion Sequence*

Both ascending and descending motions are not simple: they consist of more than 10 steps of joint angle positioning with reconnections. When a module on one end receives a message for wakeup, it becomes a master and starts an ascent procedure. As depicted in Fig. 8, the master and its specific neighbor (partner) move up with the help of their two neighbors (helper). A misalignment problem, which was serious in similar experiments in [17], was solved by a better design of this local self-reconfiguration.

Once a module pair is on the surface, the pair (a *walker*) travels by a simple procedure conducted by the same master. This motion requires the assistance of other modules. For each walker's motion, two ends of the walker connect with modules in the backbone structure. Such modules must make their connection surfaces face upward before the walker reaches them. This is made as in Fig. 9, by which two modules' surfaces are prepared without separation of the whole structure and without collision.

D. *Signal Transfer for Collision Avoidance and Wakeup*

Collision avoidance of multiple walkers is achieved using signal transfer along the backbone. As a walker travels forward, a message is sent to a module of the backbone structure connected to the walker. The message is then propagated to specific neighbors and reaches other walkers. Using a simple counting process, two succeeding walkers can maintain a safe distance. In addition, when the message reaches the end of the backbone structure, it serves as a wakeup signal. Currently, the first master module must be specified and awakened by an external command to start the whole process.

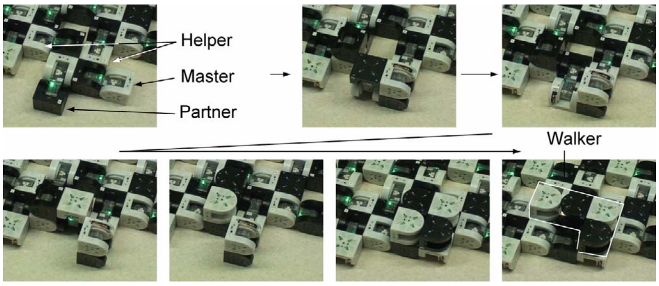


Fig. 8 Ascending procedure

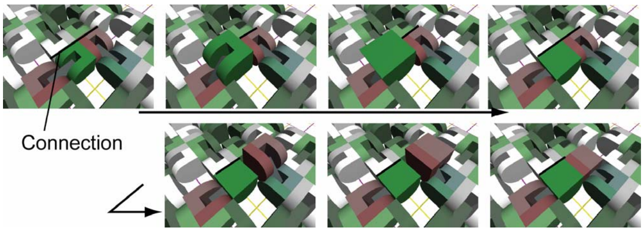


Fig. 9 Motion of backbone structure

E. *Bus communication Protocol*

The CAN bus traffic is not heavy for locomotion and centralized self-reconfiguration because messages are either commands from a single master or responses to them. For parallel self-reconfiguration, however, several master modules function independently. For that reason, message collision tends to occur. For this problem, the CAN bus and its implementation in the main CPU (SH-II) have a built-in mechanism for message arbitration and for reliable communication in a multi-master system: Carrier Sense Multiple Access with Collision Detection and Arbitration on Message Priority (CSMA/CD+AMP).

This mechanism, however, is insufficient for reliable communication of the M-TRAN III. Six mechanical switches are installed in a module to isolate the contact pins electrically when the surface is not connected: one for each of two CAN-bus lines on each of the three connection surfaces of a male block (Fig. 3(6)). An unintended consequence is that no switch is sufficiently reliable; it temporarily switches to OFF, even when the surface is connected. This malfunction occurs when joint motors are moving. Such a failure sometimes causes a temporary cut of bus lines and communication errors. This state lasts from milliseconds to a second, and the sender module becomes isolated from the network with the mechanism of CAN. This *bus-off* state is not recovered automatically, though the mechanical failure is temporary and is recovered after all motors cease motions.

We first introduced periodic null-message transfer similar to that of Heart-Beat in the Device Net Protocol to enable automatic recovery from the bus-off state and for failure detection. This, however, repeatedly caused the same problem.

Reliable one-to-one communication was finally achieved using a mixture of commonly used communication protocols: handshaking with an acknowledgement message (ensured by avoiding broadcast message transfer), retrial on message loss, forced reset of the CAN interface from a bus-off state, etc.

During experiments portrayed in Figs. 6 and 7, several retrials of message transfer were observed. This makes up about 4–5 s of maximum delay with a 800 ms retrial period, which roughly coincides with the time constant of large angle positioning.

F. Management of Neighbors' IDs

All modules have their unique IDs in the network bus and one-to-one communication relies on them. The network has no information related to the actual distance between modules and no difference pertains to communications depending on the distance. Because a master module and its neighbor modules make local coordinated motions for self-reconfiguration, each module needs IDs of other modules, mostly of its direct neighbors.

All experiments described above are made using arbitrary modules with the same program. Neighbors' IDs in each module are acquired and managed as follows. At startup, all modules start an initialization process similar to the process described in Sections III-B and IV-B, and acquire neighbors' IDs. During self-reconfiguration, modules exchange their neighbors' IDs accordingly after they change locations or connections.

G. Local Communication

As is often claimed, bus communication is inappropriate for system scalability. ID numbering might limit the maximum number of modules. In our case, the maximum is set as 60, i.e. 6 bits, because we only have 50 M-TRAN III modules. Bus traffic is also dependent on the number of modules. It can therefore be a problem for a larger system. An alternative is local communication between each of two neighbor modules, which works in parallel and is independent of the system's size. An infrared (IR) serial communication is an appropriate candidate [1, 3, 7, 9] because it is non-contact; devices for standard protocols such as IRDA are available. Its drawback is that many devices for all connections, e.g. six devices for M-TRAN, but bus communication requires only a single device. For that reason, M-TRAN III uses CAN and does not use an IRDA device. Both IR transmitters and detectors are implemented in M-TRAN III, but they are intended for proximity sensing. Nonetheless, serial communication is possible using these IR devices. Although we have made programs for them, they are as slow as 333 bps.

Figure 10 shows a preliminary experiment in which a module moves along a flat surface made by other modules. Because the walker module can send a message directly to its neighbor without knowing the neighbor's ID, a simple program in the walker module executes the process. Moreover, the backbone modules are not connected at all in this setup; this experiment cannot be done using CAN bus communication, as described above.

Using a similar program, parallel self-reconfiguration similar to those in Fig. 6(b) was also made using IR communication alone. The experiment using eight modules was slower than the original one (not slower than 50%), but this slow IR communication will ultimately be beneficial in a very large system because of its parallelism and because of the delay of CAN communication described in Section IV-E.

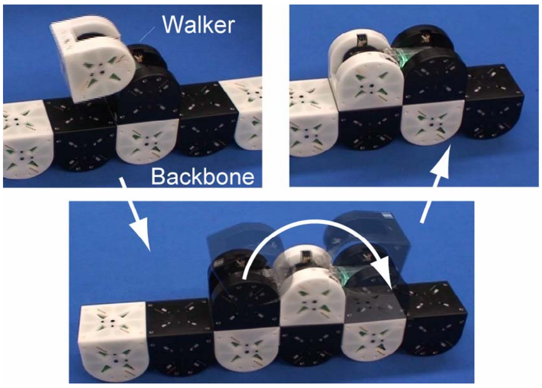


Fig. 10 Preliminary experiment using IR communication

Self-reconfiguration in Fig. 7 has not been conducted using IR communication. It would require additional programs for message propagation along multiple modules.

V. Conclusion

We developed a self-reconfigurable modular robot, M-TRAN, and carried out various experiments of locomotion and self-reconfiguration using distributed control and message exchange by network.

Near future works include various experiments of self-reconfiguration using CAN communication or IR communication. The rectangular structure can move along a fixed direction using local self-reconfigurations. We will try experiments by which such a structure changes its motion direction or changes its shape among various two-dimensional shapes. In such reconfigurations, maintenance of total connectivity and separation avoidance is important. Such works will engender autonomous shape-shifting by introducing a sensing environment and distributed decision-making processes based on local dynamics and local interaction.

In this study, IR communication is used only between direct neighbors, but it can function between distant modules. Another future work is self-assembly of multiple separate module clusters.

References

[1] S. Murata, E. Yoshida, H. Kurokawa, K. Tomita, and S. Kokaji, “Self-repairing mechanical system,” Autonomous Robots, vol. 10, pp. 7-21, 2001.

[2] D. Rus, Z. Butler, K. Kotay, and M. Vona, “Self-reconfiguring robots,” Commun. ACM, vol. 45, no. 3, pp. 39-45, 2002.

[3] M. W. Jorgensen, E. H. Ostergaard and H. H. Lund, “Modular ATRON: Modules for a self-reconfigurable robot,” Proc. IROS 2004, pp. 2068-2073, 2004.

[4] C. Gueganno, and D. Duhaut, “Distributed robotic: a language approach,” Distributed Autonomous Robotic Systems (DARS 7), pp. 12-14, 2006.

[5] V. Zykov, S. Mytilinaios, M. Desnoyer, and H. Lipson, “Evolved and designed self-reproducing modular robotics,” IEEE Trans. Robotics, vol. 23. no. 2, pp. 308-319, 2007.

[6] M. Yim, D. Duff, and K. Roufas, “Polybot: A modular reconfigurable robot,” Proc. ICRA 2000, pp. 514-520.

[7] A. Castano, A. Behar and P.M. Will, “The Conro modules for reconfigurable robot,” IEEE/ASME Trans. Mech., vol. 7, no. 4, pp. 403-409, 2002.

[8] D. Marbach, and A. J. Ijspeert, “Online optimization of modular robot locomotion,” Proc. IEEE Int. Conf. Mechatronics and Automation (ICMA 2005), pp. 248-253, 2005.

[9] W. M. Shen, M. Krivokon, H. Chiu, J. Everist, M. Rubenstein, and J. Venkatesh, “Multimode locomotion via SuperBot robots,” Proc. ICRA 2006, pp. 2552-2557, 2006.

[10] Z. Butler, K. Kotay, D. Rus, and K. Tomita, “Generic decentralized control for lattice-based self-reconfigurable robots,” Intl. J. Robotics Research, vol. 23, no. 9, pp. 919-937, 2004.

[11] M. Shimizu, A. Ishiguro, and T. Kawakatsu, “Slimebot: A modular robot that exploits emergent phenomena,” Proc. ICRA 2005, pp. 2982-2987.

[12] S. Murata, E. Yoshida, A. Kamimura, H. Kurokawa, K. Tomita, and S. Kokaji, “M-TRAN: Self-reconfigurable modular robotic system,” IEEE/ASME Trans. Mech., vol. 7, no. 4, pp. 431-441, 2002.

[13] A. Kamimura, H. Kurokawa, E. Yoshida, S. Murata, K. Tomita, and S. Kokaji, “Automatic locomotion design and experiments for a modular robotic system,” IEEE/ASME Trans. Mech, vol. 10, no. 3, pp. 314-325, 2005.

[14] H. Kurokawa, A. Kamimura, E. Yoshida, K. Tomita, S. Murata, and S. Kokaji, “M-TRAN II: Metamorphosis from a four-legged walker to a caterpillar,” Proc. IROS 2003, pp. 2452-2459.

[15] H. Kurokawa, E. Yoshida, A. Kamimura, K. Tomita, S. Murata, and S. Kokaji, “Self-reconfigurable M-TRAN structures and their walker generation,” Robotics and Autonomous Systems, vol. 54, no. 2, pp. 142-149, 2006.

[16] E. H. Ostergaard, K. Tomita, and H. Kurokawa, “Distributed metamorphosis of regular M-TRAN structures,” Proc. Distributed Autonomous Robotic Systems, pp. 161-170, 2004, (DARS 6, Springer, in press).

[17] H. Kurokawa, K. Tomita, A. Kamimura, S. Murata, Y. Terada, and S. Kokaji, “Distributed metamorphosis control of a modular robotic system M-TRAN,” Distributed Autonomous Robotic Systems (DARS) 7, Springer, pp. 115-124, 2006.

[18] S. Murata, K. Kakomura, and H. Kurokawa, “Docking Experiments of a modular robot by visual feedback,” Proc. IROS 2006, pp. 625-630, 2006.