Planning Behaviors of Modular Robots with Coherent Structure using Randomized Method

Eiichi Yoshida[1], Haruhisa Kurokawa[1], Akiya Kamimura[1], Satoshi Murata[2], Kohji Tomita[1], and Shigeru Kokaji[1]

[1] Intelligent Systems Institute, National Institute of Advanced Industrial Science and Technology (AIST), 1-2 Namiki, Tsukuba, Ibaraki 305-8564 Japan
{e.yoshida, kurokawa-h, kamimura.a, k.tomita, s.kokaji}@aist.go.jp

[2] Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology, 4259 Nagatsuta-cho, Midori-ku, Yokohama, Kanagawa 226-8502 Japan
murata@titech.ac.jp

Summary. A behavior planning method is presented for reconfigurable modular robots with coherent structure using a randomized planning. Coherent structure is introduced to cope with difficulty in planning of many degrees of freedom, in terms of control system and robot configuration. This is realized by a phase synchronization mechanism together with symmetric robot configuration, which enables the robot to generate various coherent dynamic motions. The parameters of control systems are explored using a randomized planning method called rapidly exploring random trees (RRTs). The RRT planner has an advantage of simple implementation as well as possibility of integrating differential constraints. The dynamic robot motion is thus planned and preliminary simulation results are shown to demonstrate the proposed planning scheme can generate appropriate behaviors according to environments.

1 Introduction

Self-reconfigurable modular robots are recently investigated intensively starting from earlier work [1, 2], since their flexibility, versatility, and fault-tolerance are considered to be suitable for wide range of tasks [3, 4, 5]. They are especially expected to be used as robots that can operate in unknown or unstructured environments, sometimes hostile to humans, through their adaptability. Those applications include a robot that tries to find survivors in corrupted buildings, planetary exploring vehicles, or inspection robots in nuclear plants. Many hardware systems have been developed as well as software. As to the hardware, many types of three-dimensional (3-D) self-reconfigurable robots have been developed and recently their reliability and self-containedness made a remarkable progress [6].

In this paper we focus on a behavior planning of such modular robots, to decide how those robots with many degrees of freedom act according to the environments. There are mainly two contexts of research on the software of self-reconfigurable robots. One is discrete reconfiguration planning that gives how the robot should make a sequence of configuration changes so that it can transform itself from one configuration to another [7]–[12]. The other is continuous

aspect about how to generate dynamic behavior to allow the robot make useful and meaningful motions for the application [6, 13].

Although the reconfiguration planning has been addressed intensively, dynamic behavior planning remains less exploited except some work on CPG network [6]. In view of unifying the above two research contexts, we propose a method for planning behaviors of a modular robot with coherent structure. Here the "behaviors" corresponds to various different dynamic motions. With "coherent" structures in robot configuration and control system such as synchronized motions or structure symmetry, various behaviors of the robot with many degrees of freedom (DOFs) can be controlled with reduced numbers of parameters.

As control structure, we adopt a simple phase synchronization mechanism where such parameters as phase difference determine the robots' behavior. However, search space is still very large even if those parameters are not numerous. We use a randomized planning method called rapidly-exploring random tree (RRT) [15] to explore this search space for the behavior.

In the next section, a simple phase synchronization mechanism is introduced, and then the planning with randomized method is presented in Section 3. Some simulation results using modular robot M-TRAN [14] are shown in Section 4 before concluding the paper.

2 Using Phase Synchronization for Dynamic Motion

2.1 Simple Phase Synchronization Mechanism

In order a robot with many DOFs such as modular robot to generate useful motions, coherency is necessary for its control system as well as its configuration. There are several ways to realize coherent control system. One of most popular methods is using a neural network that generates oscillatory signals, like central pattern generator (CPG) [16, 17].

In this paper, to give a clear perspective of the problem, we adopt a simple phase synchronization mechanism. We begin with a simple illustrative example composed of two connected elements 1, 2. The value θ_i is a phase variable that describes the internal state of each element. In this example, both elements try to have constant velocity ω so that θ_1 of element 1 is always greater than θ_2 of element 2 by phase difference ϕ. The differential equation can be written as

$$\begin{aligned} \dot{\theta_1} &= \omega - k(\theta_1 - \theta_2 - \phi) \\ \dot{\theta_2} &= \omega - k(\theta_2 - \theta_1 + \phi) \end{aligned} \tag{1}$$

where k is a gain coefficient. By solving this equation, we have

$$\begin{aligned} \theta_1(t) &= \omega t + \frac{1}{2}(\theta_1^0 + \theta_2^0 + \phi) + e^{-2kt}(\theta_1^0 - \theta_2^0 - \phi) \\ \theta_2(t) &= \omega t + \frac{1}{2}(\theta_1^0 + \theta_2^0 - \phi) + e^{-2kt}(-\theta_1^0 + \theta_2^0 + \phi) \end{aligned} \tag{2}$$

where θ_1^0, θ_2^0 are the initial values at t=0. The difference converges to ϕ.

This mechanism can be extended to the case of n elements:

$$\dot{\theta_i} = \omega - k\sum_{j=0}^{n}(\theta_i - \theta_j - \phi_{ij}) \tag{3}$$

If there are not loops with the element connection, the phase difference between element i and j converges to the given value ϕ_{ij} [18]. In this paper we only address the cases without loops

for simplicity, although some methods are proposed to cope with cases of existence of loops [19]. This mechanism has also an advantage that distributed implementation into modular robots.

We will apply this simple mechanism to generate various dynamic motions of modular robots. Since the output of (3) is linearly increasing along the time, we use a sinusoidal function to generate an oscillatory movement like

$$a_i(t) = \beta_i + \alpha_i \sin \theta_i(t) \tag{4}$$

where α_i and β_i is the amplification and offset of oscillation.

By assigning this synchronization element to each actuator, the modular robot is expected to generate diverse coordinated actuator outputs for dynamic motion. However, it is difficult for the robot to generate motions if the control system does not have a good accordance to the robot structure.

For this reason, another coherency is introduced in the robot structure, namely symmetrical configurations in our case. Even if the control system has coherency, a robot that has some irregular structure can hardly be controlled. Looking at the nature, animals indeed have symmetric form to make efficient motions.

Fortunately, self-reconfigurable modular robots can have a variety of configurations. Symmetric configurations can be used to apply the phase synchronization control mechanism to realize the dynamic motions such as gait patterns. Moreover, not only changing the dynamic motion, but they can choose different configurations according to the application, sometimes a four-legged robot or a snake-like robot. This is one of the major advantages of modular robots as mentioned earlier.

2.2 Self-Reconfigurable Modular Robot M-TRAN

To fully exploit these advantages, we adopt a symmetric structure that can realize different dynamic motions as well as configuration using a modular robot platform M-TRAN.

M-TRAN (Modular TRANSformer) has been developed in AIST that can realize both self-reconfiguration and dynamic motions in three dimensions. The module has a simple bipartite structure. Each part rotates about an parallel axis by geared motors and has three magnetic connecting faces as shown in Fig. 1. Each module is a self-contained with embedded a controller circuit board and a battery. Figure 2 shows the newest hardware model "M-TRAN II." For details on hardware, please see other references [6, 14].

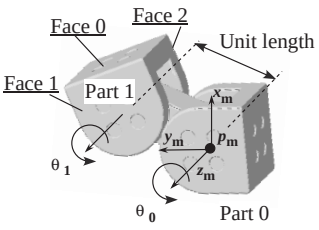


Fig. 1. A module of M-TRAN.

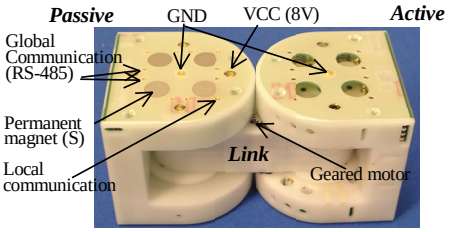


Fig. 2. A hardware module of M-TRAN II.

2.3 Applying the Synchronization to M-TRAN Modular Robot

In this paper, we deal with a configuration composed of nine modules as shown in Fig. 3. By assigning the phase synchronization mechanism to each module's actuator appropriately, generated oscillatory output enables the robot to make efficient locomotion.

Owing to the symmetry of the robot structure, all the parameters do not have to be controlled individually. Instead, the same parameters can be used repeatedly to symmetrically corresponding actuators and synchronization connections. In the case of configuration in Fig. 3, the parameters can be considerably reduced since there are two symmetry axes as shown Fig. 4. In this figure, circles and dotted lines denote oscillatory elements and connection for synchronization respectively. The arrow is defined the direction in such a way that ϕ is the phase difference from outgoing element to incoming one.

Exception of the symmetrical parameter assignment exists regarding Module 1 that is in the center of the structure. Its amplification and offset of the oscillation α_i, β_i and phase difference ϕ_5 are need to be controlled separately. The value ϕ_4 to Module 1 is also applied in different direction.

Based on this parameter assignment, the modular robot with this configuration can realize different structures for locomotion. We assume that the actuators are controlled by velocity.

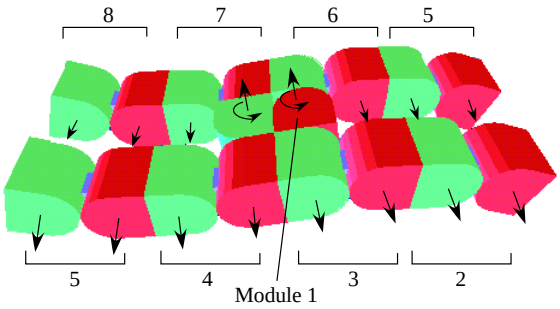


Fig. 3. A Coherent configuration of M-TRAN modules

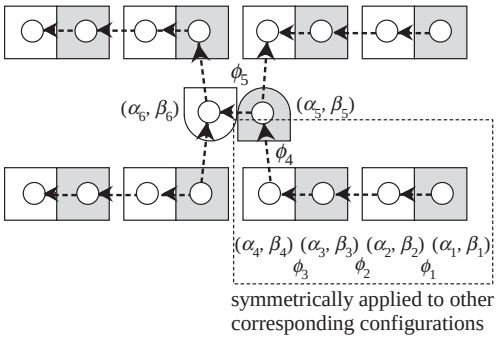


Fig. 4. Assignment of phase synchronization parameters to symmetric robot structure

One is a flat configuration whose corresponding motion snake-like wave motion (Fig. 5). Another motion is four-leg configuration that requires certain gait pattern to move, using parameters illustrated in Fig. 6. Rotational motion can be realized by setting non-zero values to α_5, α_6 for Module 1. In general, the opposite direction of motion can be generated by reversing ϕ_i values.

It is noteworthy that a single synchronization scheme can realize those very different locomotion modes are by changing parameters and synchronizing connection. In the next section, we will describe how those behaviors can be planned using a randomized method.

3 Behavior Planning using a Randomized Method

There are several ways to derive behaviors determined a number of parameters. A heuristic planning method for static motion is proposed to enable a legged robot to move rough terrain [20]. Støy et al. proposed a control model for chain-type modular robots using coupled oscillators [22]. As a more general scheme, gradient method has been proposed to optimize those phase synchronization mechanism [21].

Kamimura et. al use a genetic algorithm (GA) to obtain CPG parameters for locomotion [6]. This pattern generation method is more dedicated to real-time control and adaptation according to external stimuli, where many parameters of CPG model should be regulated. However, in the planning phase, a simplified control model is more preferable than direct usage of rather complex model of CPG since it is important to reduce the number of parameters to explore.

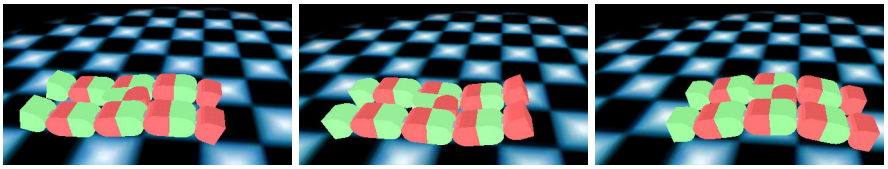


Fig. 5. A snake-like wave motion: with parameters $\alpha_1 \sim \alpha_4 = 10^\circ$, $\alpha_5, \alpha_6 = 0^\circ$, $\beta_1 \sim \beta_4 = 0^\circ$, $\beta_5 = -90^\circ$, $\beta_6 = 90^\circ$, $\phi_1 \sim \phi_3, \phi_5 = 30^\circ$, $\phi_4 = 0^\circ$, $\omega = 180^\circ$

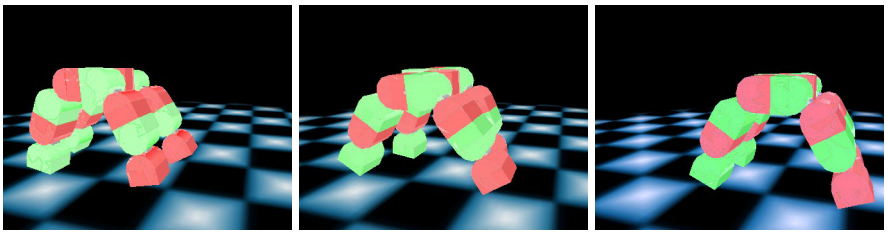


Fig. 6. Legged locomotion: with parameters $\alpha_1 \sim \alpha_4 = 5^\circ$, $\alpha_5, \alpha_6 = 0^\circ$, $\beta_1 = 0^\circ, \beta_2 = 10^\circ, \beta_3 = 20^\circ, \beta_4 = 40^\circ$, $\beta_5 = -90^\circ$, $\beta_6 = 90^\circ$, $\phi_1, \phi_3, \phi_4 = 90^\circ$, $\phi_2, \phi_5 = -90^\circ$, $\omega = 180^\circ$

In this paper, focusing on planning dynamic behavior using a simple control model based on coherent structure, a randomized planning method called rapidly-exploring random trees (RRTs) is introduced. Using this method incremental and reactive behavior planning can be implemented in a simple manner in terms of both planning and control mechanism. The CPG pattern generator [6] can then be used for real-time adaptive control to execute the planned motion. To apply this method, we assume that the robot has knowledge about local environment and its goal through external sensor capacity.

3.1 Rapidly-Exploring Random Trees (RRTs)

RRTs have been proposed by LaValle as a randomized motion planning method [15]. The idea is to incrementally construct search trees that attempt to rapidly and uniformly explore the state space. It has been proven that this method is probabilistic complete, namely desired path will be found eventually as the number of vertex becomes infinity. Since it has such advantages as simplicity of implementation for exploring many DOFs and possibility of including differential constraints, we adopt this method for dynamic behavior planning.

The basic algorithm is shown in Fig. 7 to explore configuration q. The tree $\mathcal{T}$ rapidly explores through biased search of large unexplored region of the state space. At each step, after generating a random configuration node q_{rand}, the function EXTEND($\mathcal{T}$, q_{rand}) is called to extend the tree. As illustrated in Fig. 7(b), this function first selects a node q_{near} nearest to q based on given metric, and then generates a new node q_{new} that advances to q and adds it to the tree $\mathcal{T}$. Several RRT-based motion planners have been proposed according to the problem. For example, RRT-Connect [23] is a bidirectional planner that explores RRTs from initial and goal position in environments, possibly with obstacles. The search can be accelerated using bidirectional planning.

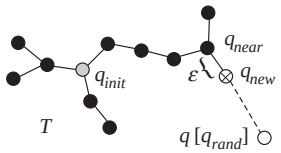
3.2 Applying RRTs to Modular Robot's Behavior Planning

Now RRT is applied to modular robot's behavior planning based on phase synchronization mechanism. As mentioned in 2.3, synchronization mechanism can be described using parameters ϕ_{ij}, ω, α and β for the configuration shown in Fig. 3.

In our case, the robot generates motions according those parameters, then it causes oscillatory motion that brings to the robot to a different position. This is a differential constraint where the relationship between control input and resulting configuration must be specified, since simple interpolation between configurations does not apply. In the algorithm Fig. 7, the configuration q should be replaced by the state x that describes robot's current state [24]. Also,

BUILD_RRT(q_{init})
1 $\mathcal{T}$.*init*(q_{init})
2 **for** k=1 to K **do**
3 $q_{rand} \leftarrow$ RAND_CONFIG( );
4 EXTEND($\mathcal{T}$, q_{rand});
5 Return $\mathcal{T}$;

(a) Building a RRT



(b) function EXTEND($\mathcal{T}$, q)

Fig. 7. Basic Algorithm of RRT

NEW_CONFIG(q, q_{near}, q_{new}) is replaced by GEN_STATE(x, x_{near}, x_{new}, Δt, *Inputs*) that generates next state x_{new} advancing to given state x from its nearest neighbor x_{near} in the tree, by selecting control input from the possible set *Inputs*.

In our example, the state x includes the position and orientation of the whole robot represented by a reference frame fixed in the Module 1 as well as each module's the output angles, position, orientation, and velocity. To compute the distance between the states x, we adopt the distance between these representative positions and orientations as the metric.

The RRT planner explores in the space of those synchronization parameters as the above *Inputs*. Then the next state x is computed as a result of dynamic motion described by the control system based on the phase synchronization mechanism. Usually, those inputs are selected in such a way that it determines directly the state of robot, like position or velocity of each actuator. However, direct search of those input results in meaningless motions in most cases. In contrast, by applying RRT to those "indirect" parameters through coherency of control system and robot structure, we expect our goal of dynamic behavior planning can be achieved. On the other hand, this causes a disadvantage of heavy computation of next state in GEN_STATE(). Currently we must calculate the next state using a dynamics simulator that is computationally expensive. To accelerate planning, this needs to be substituted simple and fast solver for dynamics and collision detection.

4 Simulation Results

Based on the behavior planning scheme presented in the previous sections, we have conducted several preliminary simulations. This section shows its results.

In the simulations, the state of the robots are described using its representative position x, y on the plane and orientation Θ of Module 1. Given its goal state, the modular robot tries to find a sequence of synchronization parameters. In this simulation, the input parameters are selected from following sets shown in Table 1. Although small numbers of values are used to reduce the search space, this simple combination turned out to be sufficient to generate various motions.

The simulation is implemented using Vortex dynamics simulator [25] and MSL library [26] for RRT planner. Collision detection is implemented in both libraries. According to each input, the next state (x, y, Θ) after time Δt = 2 (sec) are calculated by the dynamics simulator. We use relatively large Δt to allow the robot to make oscillatory motion for a certain period. Then RRT planners explore this state space to reach the goal. Here, goal-biased RRT planner is used as the planner.

Table 1. Input parameter sets for phase synchronization

amplification	$\alpha_1 \sim \alpha_4$	$5, 10, 20$
	α_5, α_6	$0, 5$
offset	$\{\beta_1, \beta_2 \beta_3, \beta_4\ \}$	{0,0,0,0}, {0,-5,-5,-5}, {0,10,20,40}
	$\{\beta_5, \beta_6\}$	{-90, 90}, {-80, 80}
phase	$\phi_1 \sim \phi_3, \phi_5$	$\pm 30, \pm 60, \pm 90$
difference	ϕ_4	0, ± 20
	$\{\ \phi_1, \phi_2, \phi_3, \phi_4, \phi_5\}$	$\{\ \mp 90, \pm 90, \mp 90, \mp 90, \pm 90\}$
angular velocity	ω	140, 180, 220

In the initial state, the robot is at (0, 0, 0) and goal state is (18, 0, 0) on a plane, where the unit length in Fig. 1. Two simulations are conducted where there are obstacles with different shape at different positions.

The simulation results are shown in Figs. 8 and 9. In Fig. 8, the robot moves around the obstacle by changing its moving direction to reach the goal position. In the next simulation, first the robot uses the four-leg locomotion to advance in free space. It could go around the obstacle, but goal-biased RRT planner found the motion to go under the obstacle with snake-like locomotion before finally restoring the four-leg locomotion. Those preliminary results demonstrated the effectiveness of the proposed method.

Figure 10 shows the explored tree in the state space projected to x-y plane in the simulations, where the thick lines are the path from the initial position to the goal. Note that the states are actually more smoothly connected than it appears because they are plotted only at every Δt.

The computation time took several minutes in both cases using Pentium M processor with 1.4GHz. In future development, improvements will be addressed using alternative fast solver for dynamics and collision detection.

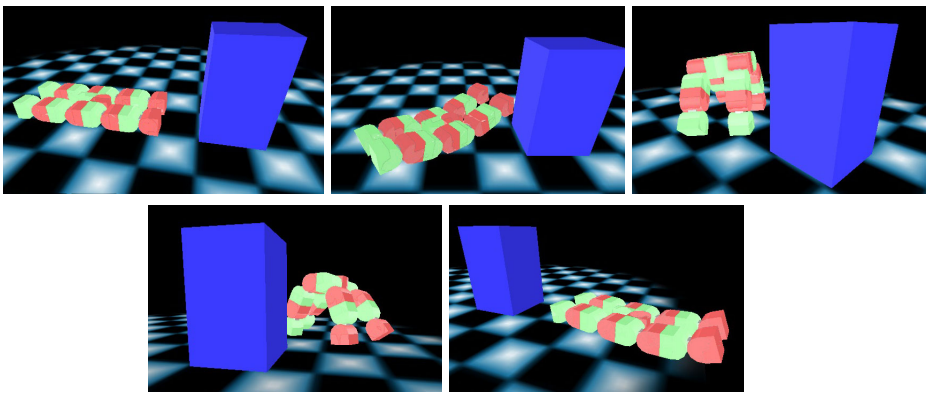


Fig. 8. Simulation results (1): avoiding obstacle in front

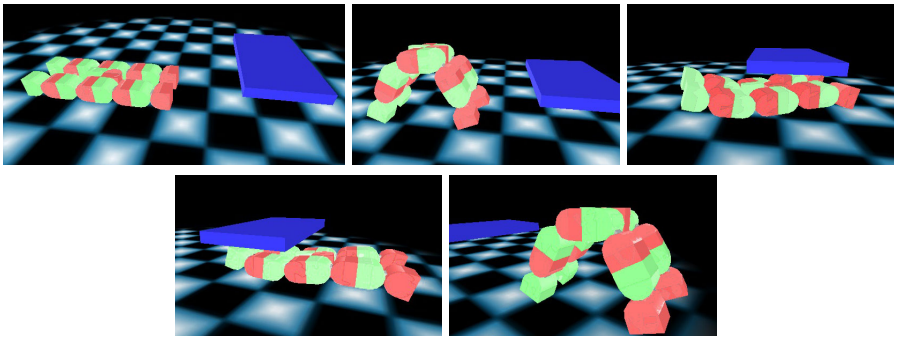
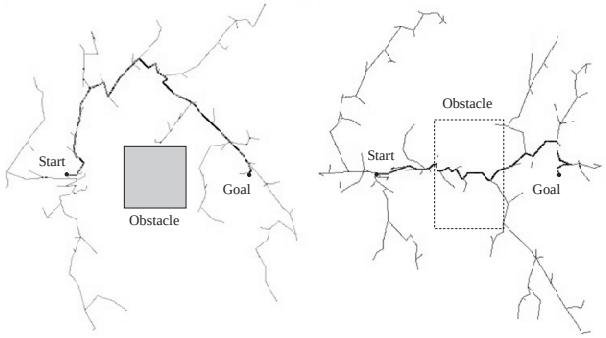


Fig. 9. Simulation results (2): going under the obstacle



simulation (1) simulation (2)

Fig. 10. Explored RRT projected on x-y plane for simulations

5 Conclusions and Future Work

In this paper we presented a behavior planning method for modular robot with coherent structure using a randomized planning method. Coherency is discussed in two aspects, in control system and robot configuration. We have introduced a simple phase synchronization mechanism for the control system and symmetry for the robot configuration. This coherency can reduce the control parameters of various dynamic motion of modular robot. A randomized method called rapidly-exploring random tree (RRT) was adopted to plan the robot's dynamic behavior. This method allows to plan a dynamic system with differential constraints based on simple implementation. The proposed method was implemented for a coherent structure of a modular robot platform M-TRAN. The preliminary simulation results showed the feasibility of the proposed method.

Future work includes such issues as integrating self-reconfiguration process, selection of coherent structures and control parameters, and more efficient implementation. Since the RRT planning scheme can include both discrete planning and differential constraints, the first issue is important in the next stage of development. This is related to the second issue, as several coherent structures can be possible according to the application. Concerning the control parameter, the control input sets are currently defined empirically. To improve the applicability of the method, improvement toward automatic acquisition of those inputs, through learning or evolutionary computation like in [6], will be addressed in the future development. Efficient implementation is also to be addressed so that to reduce the planning time. Likewise, self-learning of dynamic property through interaction with environments is also a challenging issue for implementation in real robots.

References

1. T. Fukuda and S. Nakagawa (1998) Approach to the dynamically reconfigurable robotic system, *Journal of Intelligent and Robot Systems*, **1**, 55/72.
2. S. Murata, et al (1994) Self-assembling machine, *Proc. 1994 IEEE Int. Conf. on Robotics and Automation*, 441/448.
3. *Autonomous Robots* (2001) *Special issue on self-reconfigurable robots*, **10**-1.

4. *IEEE/ASME Trans. on Mechatronics* (2002) *Special issue on reconfigurable robots*, **7**-4.
5. *Science* (2003) Shape Shifters Thread a Daunting Path Toward Reality, 301, 754/756.
6. A. Kamimura, et al (2003) Automatic Locomotion Pattern Generation for Modular Robots, *Proc. 2003 IEEE Int. Conf. on Robotics and Automation*, 2003.
7. K. Kotay and D. Rus (1998) "Motion Synthesis for the Self-Reconfigurable Molecule," *Proc. 1998 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 843/851.
8. E. Yoshida, et al (1999) A Distributed Method for Reconfiguration of 3-D homogeneous structure, *Advanced Robotics*, **13**-4, 363/380.
9. C. Ünsal, et al.: "A modular self-reconfigurable bipartite robotic system: Implementation and Motion Planning," *Autonomous Robots*, **10**-1, 23/40.
10. M. Yim et al (2001) "Distributed Control for 3D Metamorphosis," Autonomous Robots **10**-1, 41/56.
11. D. Rus and M. Vona (2001) Crystalline Robots: Self-reconfiguration with Compressible Unit Modules, *Autonomous Robots*, **10**-1, 107/124.
12. E. Yoshida, et al (2002) A Self-Reconfigurable Modular Robot: Reconfiguration Planning and Experiments, *Int. J. Robotics Research*, **21**-10, 903/916.
13. Y. Zhang et al (2003) Scalable and Reconfigurable Configurations and Locomotion Gaits for Chain-type Modular Reconfigurable Robots *Proc. 2003 IEEE Int. Conf. on Computational Intelligence in Robotics and Automation (CIRA2003)*.
14. S. Murata, et al (2002) M-TRAN: Self-reconfigurable Modular Robotic System, *IEEE/ASME Transactions on Mechatronics*, **7**-4, 431/441.
15. S. LaValle and J. Kuffner (2001) Rapidly-Exploring Random Trees: Progress and Prospects, InB. R. Donald, K. M. Lynch, and D. Rus, eds., Algorithmic and Computational Robotics: New Directions, 293/308, A K Peters, Wellesley.
16. G. Taga (1995) A model of the Neuro-Musculo-Skeletal System for Human Locomotion II / Real-Time Adaptability under Various Constraints," *Biolog. Cybern.*, **73**, 113/121.
17. H. Kimura, et al (1999) Realization of dynamic walking and running of the quadruped using neural oscillator, Autonomous Robots, **7**-3, 247/258.
18. Hiroaki Yamaguchi, et al (2001) A distributed control scheme for multiple robotic vehicles to make group formations, Robotics and Autonomous Systems, **36**, 125/147.
19. S. Kokaji, et al (1996) Clock synchronization algorithm for a distributed Autonomous System, *J. Robotics and Mechatronics*, **8**-5, 317/328.
20. C. Eldershaw and M. Yim (2001) Motion planning of legged vehicles in an unstructured environment, *Proc. 2001 Int. Conf. on Robotics and Automation*, 3383/3389.
21. H. Yuasa and M. Ito (1990) Coordination of Many Oscillators and Generation of Locomotory Patterns, *Biol. Cybern.*, **63**, 177/184
22. H. Støy et al (2002) Using Role Based Control to Produce Locomotion in Chain-Type Self-Reconfigurable Robots *IEEE Trans. on Mechatronics*,
23. J. Kuffner and S. LaValle (2000) RRT-Connect: an dfficient approach to single-query path planning, *Proc. 2000 IEEE Int. Conf. on Roboics and Automation*, 995/1001.
24. S. LaValle and J. Kuffner (1999) Randomized kinodynamic planning, *Proc. 1999 IEEE Int. Conf. on Roboics and Automation* 473/479.
25. http://www.cm-labs.com
26. http://msl.cs.uiuc.edu/msl