Yuzuru Terada
Satoshi Murata

Interdisciplinary Graduate School of Science
and Engineering, Tokyo Institute of Technology
string@mrt.dis.titech.ac.jp
murata@dis.titech.ac.jp

Automatic Modular Assembly System and its Distributed Control

Abstract

Construction is difficult to automate because of its complexity. Introducing modularity into both structural components and a means of assembly solves the problem by simplifying the construction task. Based on this idea, we propose a novel concept of a fully automated construction system called the Automatic Modular Assembly System (AMAS). In this paper, we discuss the hardware system and distributed control method of AMAS. This system uses passive building blocks called "structure modules" and an assembler robot that is specialized to handle them. This "modular" concept drastically simplifies structural complexity. We have built a prototype model to evaluate its automatic construction capability. Then we introduce a distributed autonomous control for AMAS, which uses a gradient field to indicate the directions to the assembler robots. The gradient field is generated on the structure modules. To improve the efficiency, we introduce collision avoidance rules such as module relay and local negotiation via a blackboard. We also evaluate the overall performance of the distributed control with simulations.

KEY WORDS—cellular and modular robots, distributed robot systems, mechanism design, path planning for multiple mobile robot systems, autonomous agents, robotics in construction

1. Introduction

The use of industrial robots has become increasingly common in the modern world (IFR 2005). The advantages of factory automation (FA) in the manufacturing industry include the safety of the worker, the efficiency of production and the equable quality of the products. Industrial robots could also be applicable to construction sites because construction involves dangerous (e.g. work performed in high places) and urgent work (e.g. construction of shelter), and, as a matter of course, has efficiency requirements.

There have been some cases where robots have been used practically in construction (Mitsunaga 2002). Some examples of such a challenge are unmanned construction in extreme environments such as a landslide prevention dam (Chayama et al. 2004), space structures (Whittaker et al. 2000) or work performed in high places (Ikeda et al. 2003). While remote controlled construction is the easiest to avoid casualties in these scenes, many problems remain in its practical application. Construction work that is remote controlled by human workers is usually very inefficient owing to the communication delay and poor human interfaces. Full automation cannot be realized by such a method. Some autonomous construction robots have been developed, however they can only be applied to simple tasks to help human workers (Hasegawa 1990).

There are some difficulties peculiar to construction. Environmental change is unavoidable for the tasks performed in the open air. Construction robots need to move frequently to the work area because the materials are too large to transfer to the work line. Sometimes the robots must work in small spaces such as the corner of a room. Moreover, construction needs many types of the material and requires each material to be handled differently.

To solve these problems, we propose a system that simplifies the construction processes by introducing modularity into both structural components and the means of assembly. The system consists of only two kinds of components: cubic structure modules as building components and specially designed assembler robots to manipulate the modules. This system, which is called the Automatic Modular Assembly System (AMAS; see Terada and Murata (2004, 2006) and Terada et al. (2004)), drastically reduces the complexity of the construction task.

Modular robotics (self-reconfigurable robotics) is a research field closely related to AMAS. A modular robot is made of many autonomous modules and is capable of reconfiguring its own shape to adapt to the environments (Fukuda and Nakagawa 1988; Chirikjian 1994; Kotay et al. 1998; Rus and Vona 2001; Murata et al. 2002; Shen et al. 2002; Yim et al. 2002;

Stoy et al. 2003; Jorgensen et al. 2004; Lund et al. 2005; Zykov et al. 2005; Murata et al. 2007). In modular robotics, the modules are usually assumed to be homogeneous and each module has the mobility to change the local configuration. As a control system, the self-reconfiguration algorithm is the central issue in modular robotics. Most modular robot systems are homogeneous, but there are a few heterogeneous systems. An example is I-Cubes developed by Unsal et al. (2001). This system is bipartite, composed of manipulator modules and passive cube modules. The manipulators and the cubes are essentially indivisible and thus the same number of manipulators as cubes is necessary; moreover, it is not possible to remove the manipulators from the assembled structures. AMAS is also a heterogeneous system made of assemblers and cubes, but there is no limitation on the numbers of assemblers/cubes. The assembler robots are only used temporally and they withdraw after the construction.

From the viewpoint of a self-reconfiguration algorithm, the algorithm used in AMAS differs from those used in modular robotics. For instance, for the shape-shifting problem of modular robotics, we need to find a self-reconfiguration path (or rules to generate the path) between shape A and shape B. All of the modules are assumed to be identical and equipped with autonomy with some mobility. However, in AMAS there are two different kinds of modules: active robots and passive cubes. The assembler robots have two different destinations (construction site and module supply area) depending on their states (loaded or unloaded). We need to control a group of assembler robots heading in opposite directions on the same structure. Collision avoidance is the central issue here.

Homogeneous cubic modules are often assumed in studies on algorithms. These cubic modules are usually assumed to have isotropic mobility (i.e. they can move in all x, y, z directions as long as the site is vacant). For instance, Butler et al. (2004) proposed an algorithm to generate group locomotion on random terrains. Kotay (2004) showed its ability to self-repair the shape of the structure based on the same model. Stoy and co-workers (Stoy et al. 2003; Stoy and Nagpal 2004; Stoy 2006) have proposed a powerful self-reconfigurable algorithm based on the isotropic mobility model. The core of the algorithm is a gradient field generated by using a CAD model of the target shape. Using this algorithm, they simulated the self-assembly of highly complicated three-dimensional shapes. Our method presented in this paper is inspired by this work. We have made some important changes to apply the principle to a real construction task, as well as showing a concrete design of a hardware system.

Werfel et al. (2006, 2007) proposed a construction algorithm called "swarm construction". In this method, assembler robots move around the structure and place the modules according to the rule set given to each of the assemblers. In AMAS, the assemblers have almost no autonomy; instead, they are led by the gradient field generated on the modules. The gradient field dynamically changes its shape along with the current construction status, and the effectiveness of the gradient field control is quantitatively examined in many aspects. In a sense, we utilize more computation capabilities on the structure.

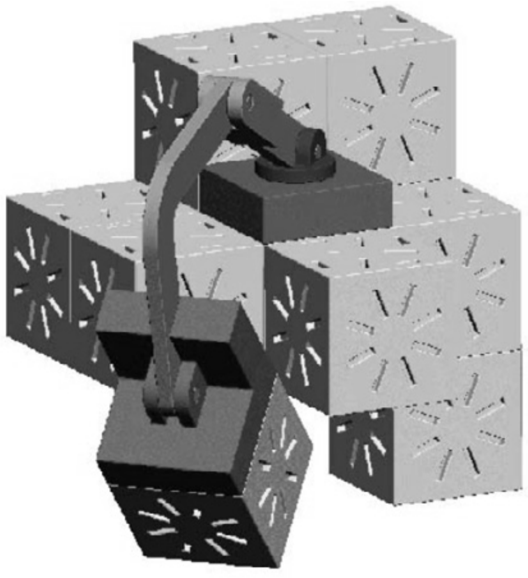


Fig. 1. Concept of AMAS.

2. System Design based on Modularity

2.1. Required Functions

In this paper, we describe the hardware design of AMAS and its distributed control method of the assembler robots (Figure 1). There are two functions that are essential for the automated assembly of a modular structure: module transportation and connection. Transportation requires grasping, carrying, localizing and routing. Connection involves precise positioning of a module and fixing the module onto other modules. We must plan and schedule the assembly process, allocate an appropriate amount of the task to each assembler robot and provide materials for the given overall shape of the structure.

2.2. Modular Design

All of the module dimensions should be normalized in our design. The whole structure is regarded as a kind of regular grid system. Thus the assembler robot's position and orientation can be normalized by the grid system. This is advantageous because a complicated measuring process that is usually necessary for construction can be omitted through the use of this grid, resulting in high efficiency and low cost.

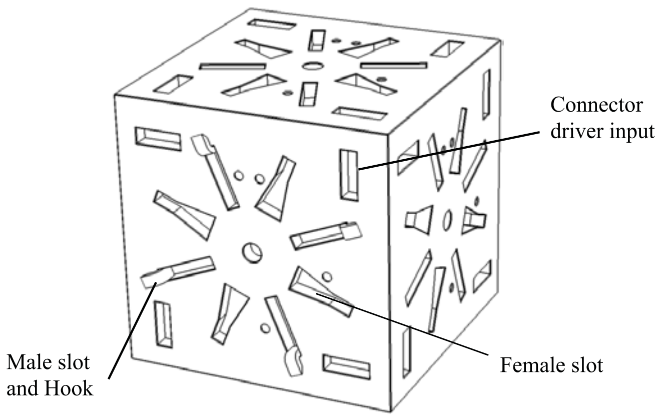


Fig. 2. Structure module.

2.3. Structure Module

The proposed design of a structure module in AMAS (Figure 2) has two features. First, every structure module is a regular hexahedron, used as a component of the modular structure. Second, the mechanical connector used to fix the module is implemented on each surface of the module. The connector, which is driven by an assembler robot as described below, are genderless and rotation-symmetric to give it complete modularity. Built-in power transmission lines are connected automatically when the modules are assembled. Assembler robots use this power network. Structure modules have no actuators but each of them has a microprocessor and sensors for information processing.

2.4. Connection Mechanism

Several properties are necessary for the automatic connection mechanism (Nilsson 2002): (1) geometrical properties such as symmetry and compactness/thinness; (2) mechanical robustness including latching ability; (3) power to hold the connection; (4) an energy/information transmission channel; and (5) maintenance and production.

We designed a hooking mechanism (Extension 1) to simplify the overall mechanism. The hook mechanism can fasten and release the connection by one axis. This design satisfies Nilsson's requirements.

(1) *Geometrical properties*. Eight radial slots are cut on each connection surface (Figure 2). Four of them contain hooks; the other four are just slots. To realize genderless connection, hooks and slots are placed alternately. This rotational symmetry allows four directions of connection. The assembler robot drives the hook via a connector driver transmission.

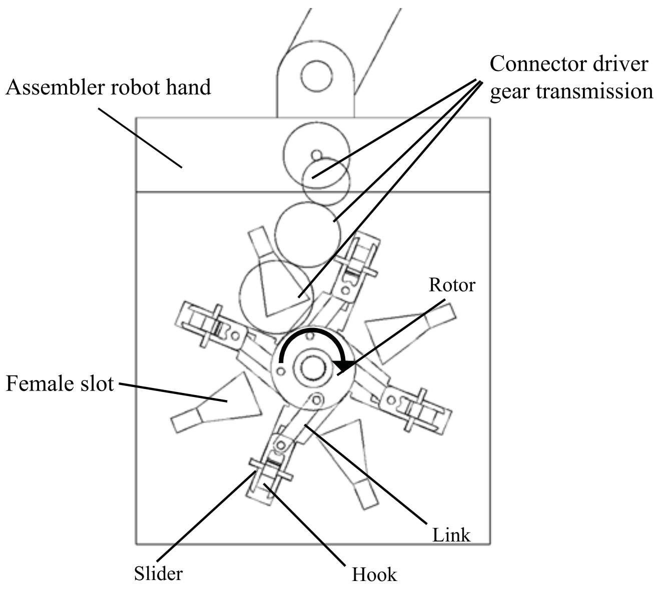


Fig. 3. Connector mechanism.

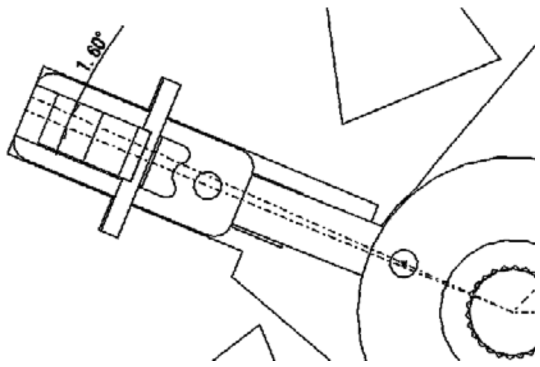


Fig. 4. Latching mechanism.

(2) *Mechanical robustness*. Figure 3 shows a plane view of the transmission mechanism, which transmits power from the assembler robot to the hook sliders. The central rotor is driven by the robot through transmission gears. The rotor rotation is changed to a sliding motion of the sliders. The rotation angle of the rotor is limited mechanically at the point where the rotor slightly (1.6°) goes beyond its maximum stretch (Figure 4). This effectively prevents pullback of the hooks by an external force (latching).

(3) *Power to hold the connection*. This connector does not require power to hold the connection (cf. electromagnets). Figure 5 shows how to move the hook of the connection mechanism. The slider raises the hook by a

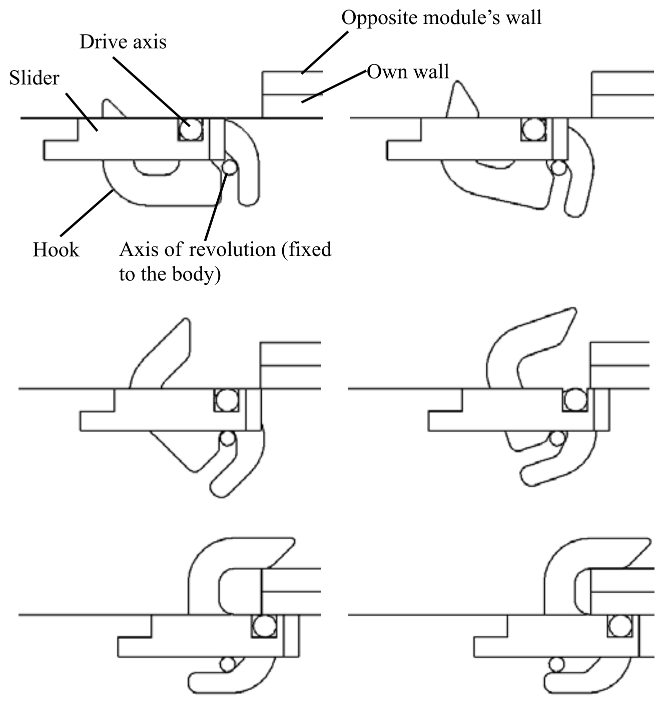


Fig. 5. Hook slide mechanism.

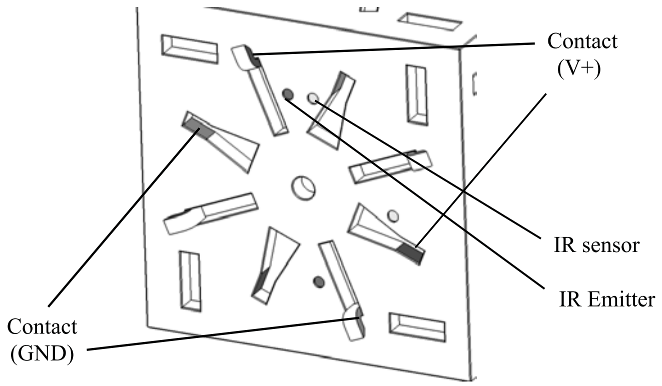


Fig. 6. Connector layout.

fixed axis of revolution and travels further until the concave part of the hook bites its own and the connecting module's wall. (The axis of revolution in the figure is fixed to a body of the structure module). Connection strength depends solely on material strength of the hook and the wall because the tensile force is received by the parallel surface of the hook.

(4) *An energy/information transmission channel*. Figure 6 shows contacts for the power transmission and infrared (IR) communication device. The communication device can also be used as a proximity sensor.

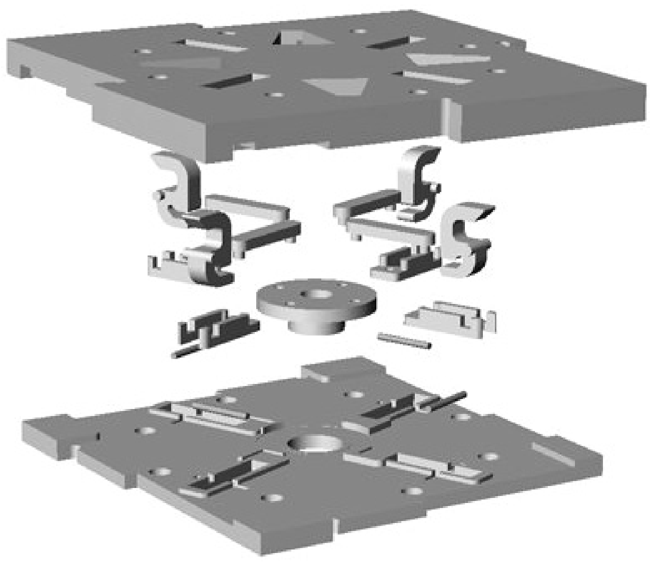


Fig. 7. Exploded view.

(5) *Maintenance and production*. The connection mechanisms are composed of two boards (main shell and backboard) and small parts which are simply sandwiched between them (see Figure 7). We can easily assemble and maintain this mechanism by removing the backboard.

2.5. Assembler Robot

Another important component of AMAS is the assembler robot. The assembler robot can walk on the modules by using connectors on their hands and carrying a module with its hand (L-shaped part). As any modular structure made up of these modules can be described on a cubic grid, a finite set of motion patterns is sufficient to build any shape. We took advantage of this to minimize the configuration of the assembler robot. Only four degrees of freedom are enough for locomotion and adding a new module on any surface of another. We can reduce the cost per robot by virtue of this simplified hardware design. This enables us to use more robots at the same cost and, as a total system, we can realize fast, low-cost and scalable construction.

The assembler robot (Figure 8) moves on the structure using an inchworm motion by repeating connecting–disconnecting actions (Figure 9). Rotation is also possible (Figure 10). The connectors and links are sufficiently strong to hold and support the robot's entire body. Therefore, it can climb a vertical wall and hang onto a ceiling. The assembler robot can construct structures by combining basic assembly actions.

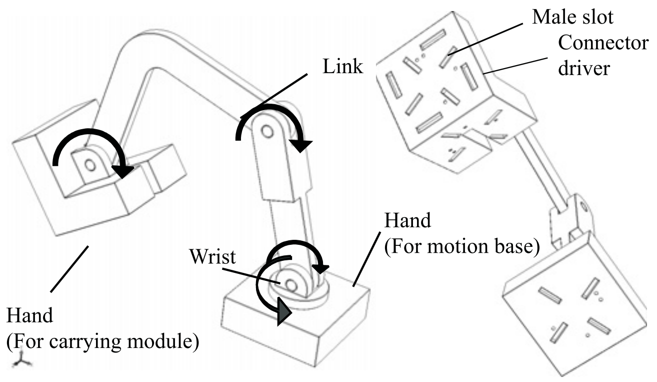


Fig. 8. Assembler robot.

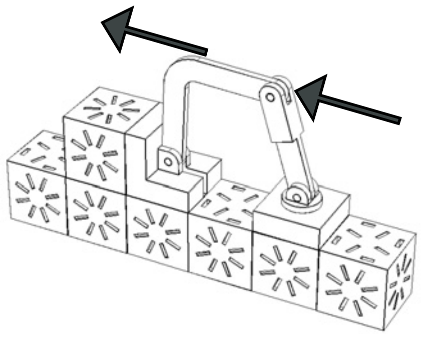


Fig. 9. Inchworm motion.

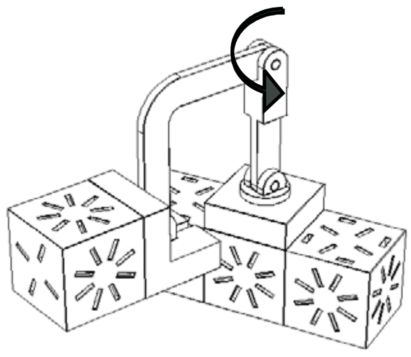


Fig. 10. Rotation.

2.6. Assembly Process

Three types of assembly action are necessary to construct an arbitrary form of the modular structure.

First, the assembler robot adds a module on the end of the plane where the robot is situated (Figure 11(a)). This action extends the modular structure horizontally (level placing). The robot can construct a very large plane by combining this motion and rotation. Second, the assembler robot places a module on the same plane upon which the robot is situated (Figure 11(b)). This action extends the modular structure upwards (over placing). The robot can move onto the vertical wall it has just made and can extend the vertical surface by the level placing action again. Finally, the assembler robot can put a module beneath the plane (Figure 11(c)). This action extends the modular structure downwards (under placing). Similarly, the robot can move onto the vertical wall (Figure 12) it has just made and can extend the vertical surface by the level placing.

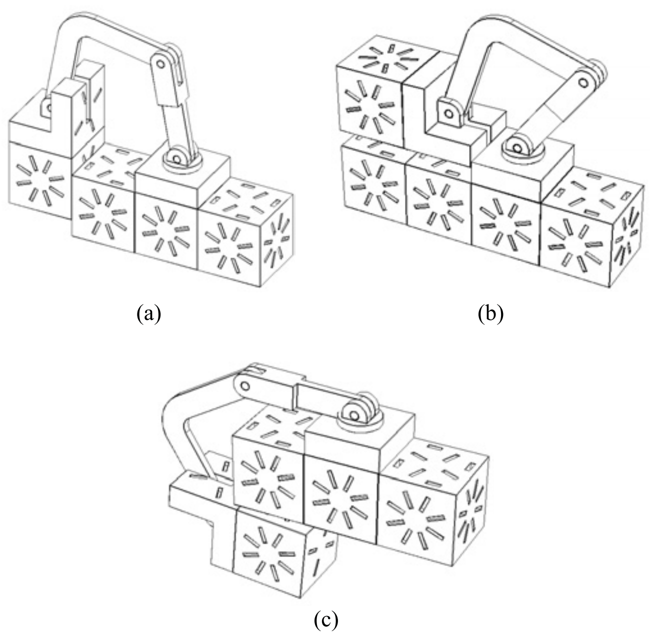


Fig. 11. Assembly actions.

3. Experiments using AMAS Hardware

We have produced a prototype system based on the design concept described in the previous sections. Figures 13–15 show the structure module, the connection mechanism (the backside of the connection surface) and the assembler robot, all of which are produced by CNC Machine Tools.

In this prototype, the module length is 70 mm and the length of the link of the assembler robot is 110 mm. Both the module and the robot are made from polyoxymethylene (POM) and acrylonitrile butadiene styrene (ABS). We used a Hitec digital servomotor HS-5125MG for the connecter driver and HS-5245MG for the link driver. The weight of the module is 185 g and that of the robot is 280 g.

This prototype system is intended to show the primary functions necessary for automated module assembly: carrying and connecting modules. Our experimental system is the minimum necessary to verify the simplest construction task. The following modifications from the original design were introduced.

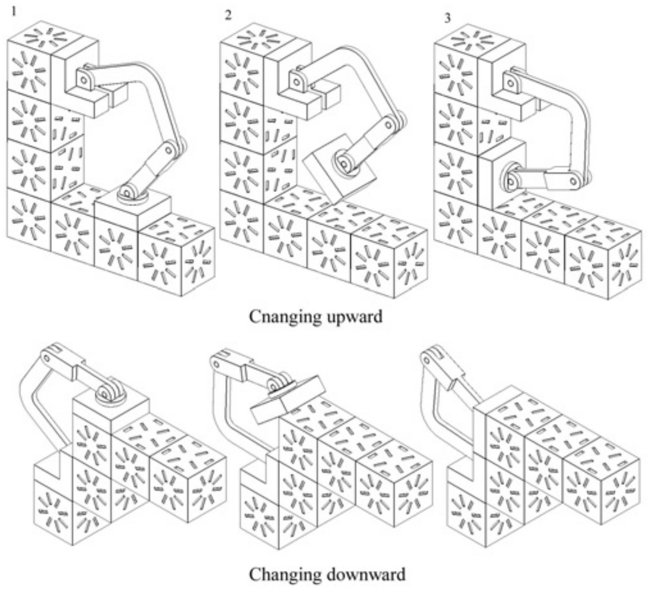


Fig.12. Changing base plate.

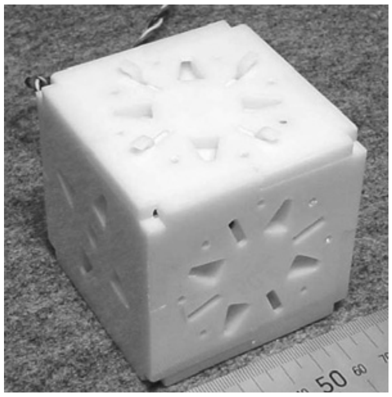


Fig. 13. Structure module.

First, the original assembler robot has two connection surfaces on the module carrying hand, but the produced robot has only one. Second, transmission of the connector driver is omitted. Instead, we install a small servo in the structure module. Third, the rotational axis of the assembler is omitted.

Based on a preprogrammed motion plan, the main PC provides motion commands such as joint angles of the assembler robot and an ON/OFF command to the connection mechanism. Commands are sent via a USB interface to a relay board, which controls all servomotors in the system. There is no positional feedback to minimize the control system except for the internal feedback of the servomotors. The error-correction ability of the connection mechanism is sufficient for this assembly task.

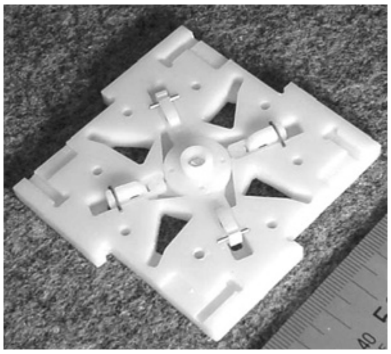


Fig. 14. Connector unit.

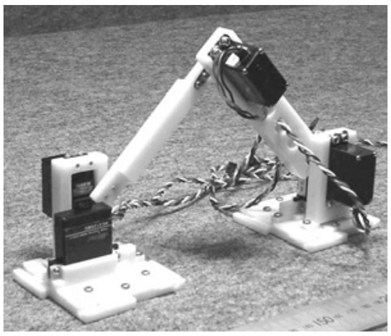


Fig. 15. Assembler robot.

The test sequence used to evaluate the basic functions was performed as follows. In the initial condition, the assembler robot is connected to one end of a structure made of four passive modules through its base hand. It holds a module with the other hand (Step 1-1 in Figure 16).

Then the assembler robot places the structural module at the assembly position in Step 3-1 and attaches it at the end of the modular structure in Step 3-2. In Step 3-1, the structural module deviated slightly from the correct position. However, it is corrected to the right position by the connection mechanism (see Extension 2).

The assembler robot revealed no problems except that in Steps 1 and 2, the link stiffness was insufficient to support twisting moment by the payload. To rectify this, we placed a guide made of thin plastic film at the side of the modular structure. This problem could be solved by redesigning the link component.

4. Distributed Control of AMAS

We have shown the hardware design of the AMAS in the previous sections. Now we discuss the control method.

AMAS is a multi-robot system. Assembly planning of a multi-robot system is difficult in general because the planning

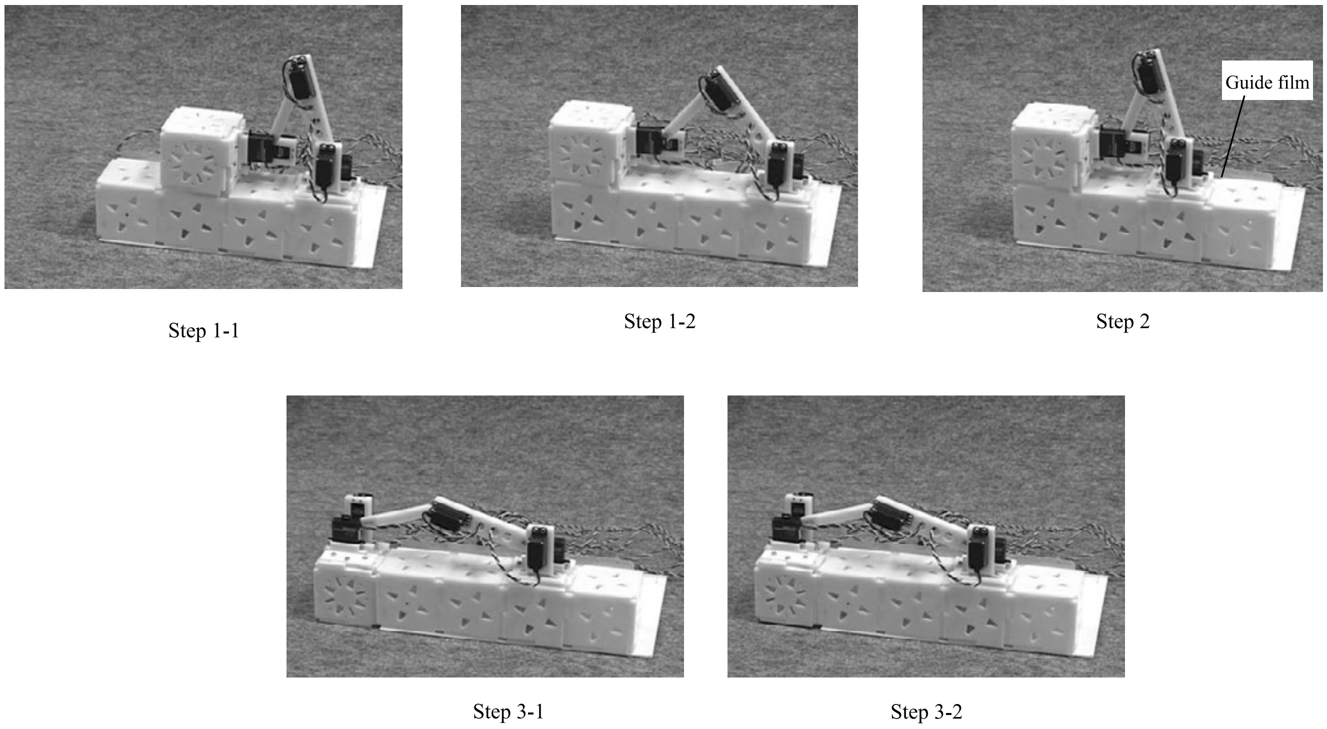


Fig. 16. Assembly task completed by the prototype hardware. Step 1: the assembler stretches its arm out to push the module forward to the next position. Step 2: the module is connected to the structure and the base hand is released and the arm contracts. This moves the assembler forward (inchworm motion) by one module. Step 3-1: the assembler reconnects the base hand to the structure and releases the module from the structure. It lifts the module and places it at the other end of the structure. Step 3-2: the assembler connects the module.

includes a cooperation of robots such as planning motion paths without interference.

Motion planning of multiple mobile robots can be classified into two categories in terms of planner: a central controller and distributed controllers (Arai and Ota 1992; Todt et al. 2000; Farinelli et al. 2004). The centralized control for multi-robot systems is a hard problem to solve in real-time because the frequency of interference among the robots increases exponentially with the density of the robots. A more feasible control method for online multi-robot systems is distributed control. In the distributed control, all of the robots have some degree of intelligence and they work in cooperation with each other by local sensing and communication (Yoshimura et al. 1996). It is processed in real-time and robots can react to unexpected disturbances autonomously. In the distributed control system, mobile robots should have simple architectures and have limited information and intelligence. However, this may cause the trapping in local minima.

In AMAS, structure modules are supplied at a certain area called the supply area throughout construction (Figure 17). The assembler robots pick up the modules at this area and carry them to the construction site (called the growth front). They shuttle back and forth between the module supply area and the construction site. This means that robots meet each other frequently during construction and they should avoid collisions. Another point is that the surface on which the assembler robots operate dynamically expands during the construction process. This means that the construction site changes from one moment to the next. Therefore, our problem is path planning (Ota et al. 1994) and task allocation (Dahl et al. 2003) of the multi-

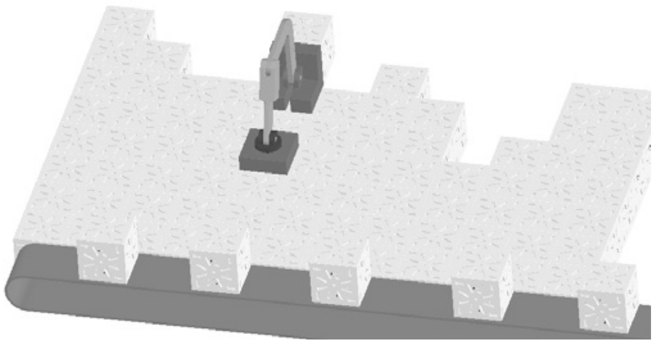


Fig. 17. Task environment.

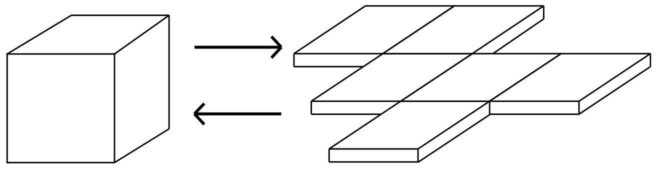


Fig. 18. Shell structure.

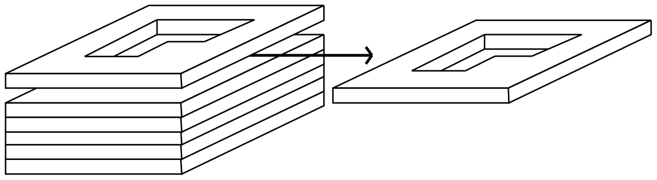


Fig. 19. Layered structure.

robot system in the changing environment. These are the reasons why we adopt the distributed control method.

4.1. Assembly Task: Planar Structure

In this paper, we restrict ourselves to a two-dimensional construction problem as shown in Figure 17. However, we believe that an algorithm that is capable of assembling any given two-dimensional structure can easily be extended to classes of three-dimensional construction.

We give two examples of such three-dimensional classes. The first is a cubic shell structure made of six planes. Similarly, any shell structure made of rectangular panels can be spread out onto a two-dimensional structure (Figure 18). A two-dimensional construction method appended with corner rules is capable of building such a class. The second example is a solid object with an arbitrary shape. It can be regarded as a pile of two-dimensional slices. If each slice has a connected shape, this kind of structure can be built using layer-by-layer construction (Figure 19).

Figure 17 shows the typical task environment we consider in this paper. A belt conveyor at the bottom represents the supply area, which gives the structure modules to the assembler robots. We assume that the number of modules on the conveyor is sufficient so that the assembler robot can pick up a module at any time. After the robot picks up a module, it carries the module towards the growth front where the carried module will be placed and connected to the panel, before it returns to the supply area. The robot shuttles back and forth between the supply area and the growth front until the modules occupy all of the vacant grid points in the panel.

Here, we assume that both the robots and the modules are equipped with microprocessors and some contact sensors. These processors can exchange digital information when they are adjacent (between modules or between a module and a robot). We also assume that an absolute coordinate system is defined on the lattice and each module or robot can identify their coordinates by communication. The desired shape of the panel is given *a priori* to both the robot and the structure modules. The modules can tell whether they are at the growth front, or inside of the shape, and whether their neighbor point is occupied.

4.2. Algorithm based on a Gradient Field

In order to optimize the complex, coordinated motion among the assembler robots on the changing field, we introduce a distributed algorithm based on a gradient field (Khatib 1986; Borenstein and Koren 1989; Stoy 2002). In our system the structure modules generate the gradient field by using neighbor-to-neighbor communication. Namely, the planar lattice of the modules functions as an intelligent field, which is capable of simulating computational model such as a partial differential equation. In AMAS, the proposed method is very simple; the gradient indicates the direction of the assembler robots whenever it is heading to the construction site or returning to the supply area. This method is capable of dynamic planning where the surface on which the assembler robots operate dynamically expands during the construction process. Incorporating some additional rules for the assembler robots, a kind of self-organized flow is generated on the construction field, and this enables collision-free and time-effective assembly.

Durna et al. (2000) showed another approach to control a distributed swarm of robots using the Holon network. The algorithm is designed for homogeneous modular robot systems and they cannot be directly applied to AMAS because of its heterogeneity.

4.3. Gradient Field

We assume that both the structure modules and assembler robots are only able to communicate neighbor-to-neighbor. Based on such local communication, robots have to acquire the location of the growth front and module supply area. In our method, the gradient field is generated by a bucket-brigade process of the positional information[1]. The robots can read the value of the gradient and recognize which direction they should go to reach the destination. To represent the gradient field, we can use continuous values or discrete values.

1. In three-dimensional structures, a structure module can have a maximum of six neighbors. If the value of the gradient field is expressed in 8-byte floating point notation, then the total communication required to update the field is $8 \times 6 \times 2$ (bilateral communication) = 96 bytes = 768 bits per cycle. We assume that the modules communicate via an IR communication channel. For instance, if we adopt a commonly used standard such as IrDA 1.0 (115.2 Kbps), the gradient field can be updated more than 150 times a second. This means that the gradient field is updated almost instantaneously.

4.4. Gradient Field using Continuous Values

A gradient field using continuous values is defined by the following diffusion equation:

$$\frac{\partial P}{\partial t} = D\nabla^2 P \quad (1)$$

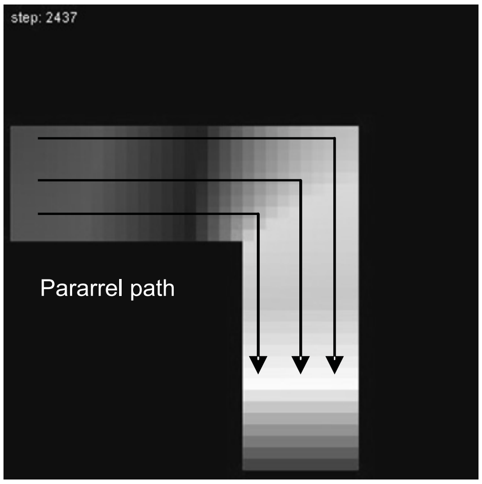
where P is the neighbor's potential and D is the diffusion coefficient. There are three kinds of boundary conditions for this system. On the growth front, the potential value is fixed at a constant high value. On the supply area, it is fixed at a constant low value. They act as source and drain. The other kind of boundary appears at the edge of the panel, where the desired shape is already assembled. On the completed edge, we assume no leakage of potential. The path generated by this field is not the shortest, but it is not crossed on the way. This is effective method to reduce the collision rate of the robots (Figure 20(a)).

4.5. Gradient Field using Discrete Values

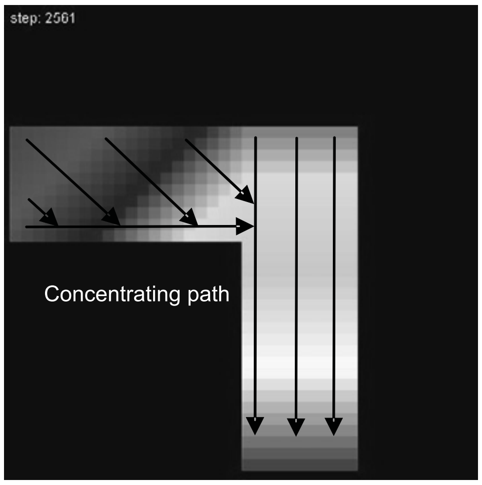
To generate the field using discrete value, the following equation is used:

$$P_{next} = \min(P_1, P_2, P_3, P_4) + 1 \quad (2)$$

The other modules read the value of four neighbors and the smallest plus one is set as its value. This value presents the Manhattan distance from the destination module, therefore representing the shortest path. Here, the destination area should have the fixed value zero. All of the other modules repeat this calculation that propagates the potential field (bucket-brigade process). A gradient field using discrete values indicates only one destination area. However, our system has two destination areas: the growth front and the module supply area. This necessitates two fields for each destination (Figure 20(b)).



(a)



(b)

Fig. 20. (a) Continuous field and (b) discrete field.

4.6. Additional Rules for Collision Avoidance

A robot carrying a module should move to the growth front lead by the gradient direction, while a robot carrying no module should move to the module supply area. As mentioned before, the robot can see only the nearest four neighbor modules. Collision may occur when more than one robot tries to move onto the same module. In such a case, the robots communicate to exchange randomly generated tokens. The robot that has the largest token wins the priority to move. Other robots cannot move until the collision is resolved. This is the main cause of decreased efficiency of AMAS. We use two additional methods to reduce such dead time.

4.6.1 Module relay

This method reduces the time loss caused by collisions. When two robots collide with each other, the one carrying the module hands it off to the other. This effectively improves the efficiency and prevents deadlock (Drogoul and Ferber 1992).

4.6.2 Planning (local negotiation via blackboard)

This method prevents collisions using only local communication. This is called a blackboard system. When a robot plans to move to the next module, the robot writes down the plan on the module. Other robots can read the plan before they try to move to the module. If the module is already engaged, they give up the motion.

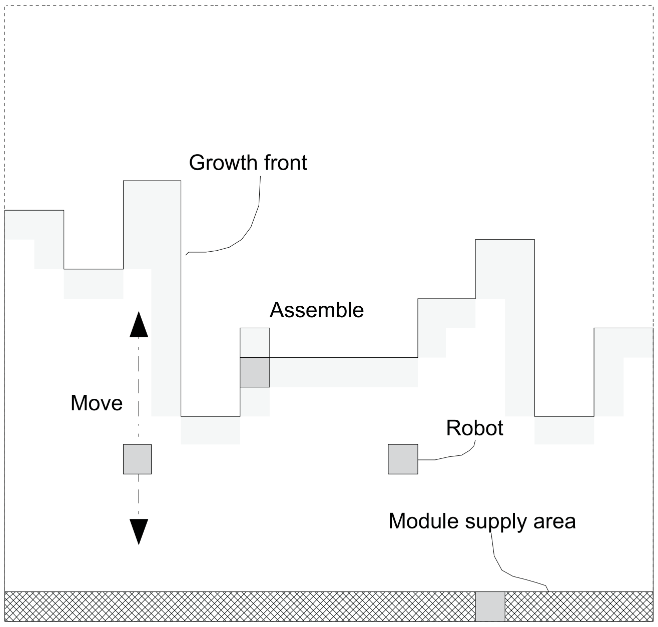


Fig. 21. Simulation environment.

5. Simulation

5.1. Simulator for AMAS

We built a simulator to evaluate the efficiency of the algorithm. Figure 21 shows the simulation environment. The same size square represents one robot or one module for simplicity. The bottom row represents the module supply area. The line represents the growth front.

In the simulation, each module or robot has its own absolute location as a pair of integers. The robots move onto the adjacent module located either above, below, to the right or to the left according to the gradient field. We assume discrete time series and that the internal state of each module and robot is synchronously updated in a step. We assume that moving one module length, picking up a module, assembling the module, relaying the module and the penalty for collision all require one time step each. The simulation process is given in the following pseudo code:

```
1: while (structure is not complete) do
2:     for ($i = 1$ to $N$) do
3:         robot[$j$] plan for this step
4:     for ($k = 1$ to $N$) do
5:         robot[$k$] do the planned action
6:     for ($l = 1$ to $U$) do
7:         for ($m = 1$ to $M$) do
8:             module[$m$] update the gradient
```

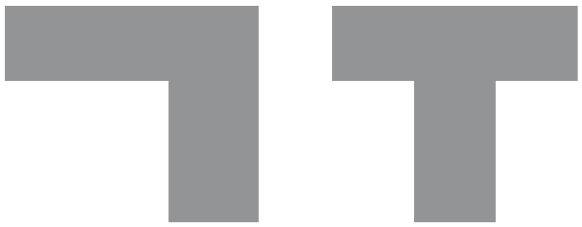


Fig. 22. Benchmark structures.

N = number of robots
j = random number which is not appeared in this loop yet
U = ratio of communication to robot motion
M = number of modules

When the simulation starts, only the module supply area exists and all of the assembler robots are located there.

We use two target structures as benchmarks. They are an L-shaped structure and a T-shaped structure (Figure 22). The lower bound of the assembly step can be calculated by the sum of the Manhattan distance of all of the modules.

5.2. Simulation Results

The total steps required to complete each of the two structures are evaluated by the simulation. The total number of steps is calculated as the number of steps required to complete the structure multiplied by the number of robots. It is the total amount of working time steps. The number of robots is varied from one to ten. All combinations of target structure, number of robots, types of gradient fields and types of collision avoidance methods are examined. The plots below are an average of ten trials and the error bars show the maximum and minimum of the trials. Lower bound of the assembly step is shown as a control[2].

Figure 23 shows the result of the simulations (see also Extensions 3 and 4). First we compare the results of the assembly using two kinds of gradient field. When assembling the L-shaped structure, the efficiency suffers as the numbers of robots are increased because of collisions. Paths at the corner become the bottleneck in the L-shaped structure. The discrete field makes only the shortest paths, and thus all of the robots paths are concentrated in the corner (Figure 20(b)). The continuous field generates paths with more clearance (Figure 20 (a)). This is the reason why the continuous field is better

2. The lower bound is calculated as the Manhattan distance between the module supply area and the place where each module is assembled. This distance is multiplied by two (because the robot should return to the supply area) and summated for all of the modules in the complete construction. Note that we subtract (robot number) × (the distance from module supply area to the farthest module), because the construction process is completed when the robot assembles its last module.

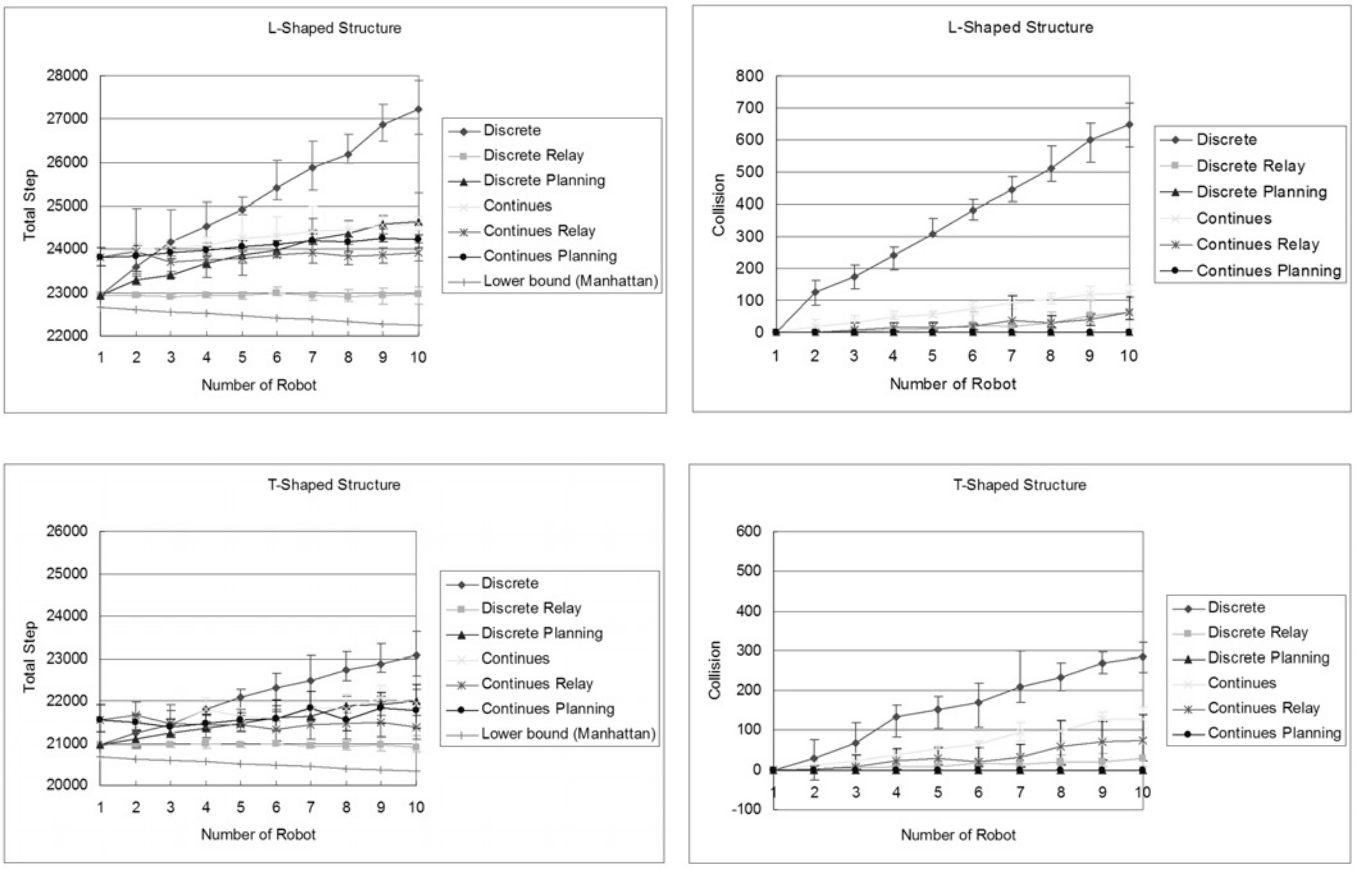


Fig. 23. Simulation results.

than the discrete field at high robot densities. The result of assembling a T-shaped structure is quite similar to that of assembling an L-shaped structure, but the number of collisions is almost 50% less, because the T-shaped structure has two corners so the apparent density of robots per corner is reduced by one-half.

Next, we have evaluated the effect of the collision avoiding rules. In both target structures, the rules effectively reduced the number of collisions. An interesting point is that the number of collisions becomes zero when using the blackboard planning but the performance becomes less efficient than using the module relay rule. Using blackboard planning, robots should avoid other robots that are in their direction of travel. In contrast, module relay does not require avoidance if the robots meet; one simply relays a module to the other. The two methods differ in that module relay prevents a chain of collisions while the blackboard planning does not. Figure 24 (and Extension 5) shows the L-shaped structure assembled by the combination of a discrete field and module relay. In this case, robots are spontaneously grouped into two sets: the first group carrying modules to the corner and the second group assembling the modules received from the first. By dividing the labor, the frequency of collisions is decreased because the robots do not need to go through the corner where the collisions are most likely to occur. As a consequence, the best performance is given by the combination of a discrete field and module relay.

5.3. Evaluating Robustness

To evaluate the robustness of this system, we conducted two different simulations. First, we removed some modules from the structure. This simulates the defective connection of the modules or removal of the modules. Second, we assumed a breakdown of communication ability in some modules. We used 28 robots to assemble a square structure made by 28 × 28 modules in these simulations. Under the normal conditions, this assembly task was completed within 730 steps. We apply disturbance (removal of the modules or communication breakdown) at the 360th step.

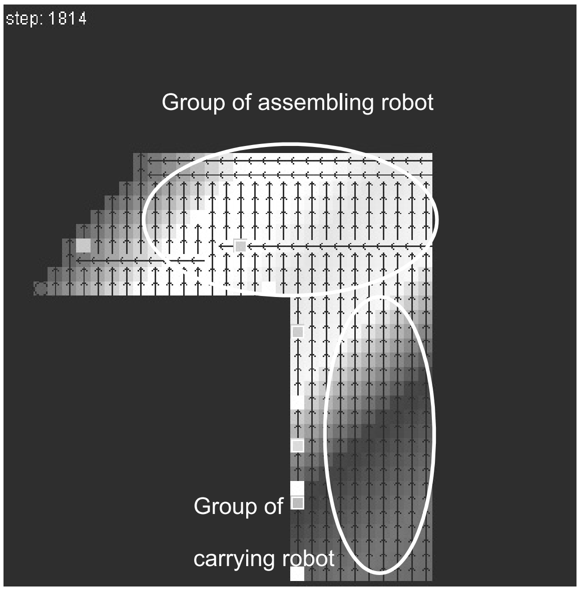


Fig. 24. Division of labor.

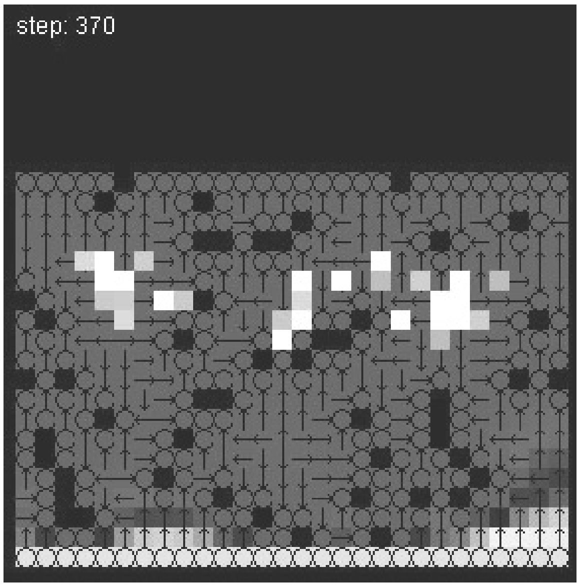


Fig. 25. Robust assembly with module removal.

5.4. Robustness for Module Removal

In this simulation, modules are removed by some external disturbances. The modules can detect the disappearance of their neighbors because they constantly communicate with each other to update the gradient field. When a module detects its neighbor's absence, its status is changed to "growth front", which attracts the assembler robots towards the removal site.

In the simulation, we observe that robots surrounded by such growth-front modules lose their way. In Figure 25 (and Extension 6), there are three clusters of robots in this situation. They cannot identify the direction of the module supply area, because all of the modules around them have the same value and make no gradient. Consequently, robots have to move randomly until they reach the normal gradient field[3]. This kind of wandering behavior of robots results in low total construction efficiency. However, we also observe that the assembly is not deadlocked in these regions as long as there is at least one robot that can escape from the trap, because the gradient field is restored when the robot returns to the site with a module to fill the hole.

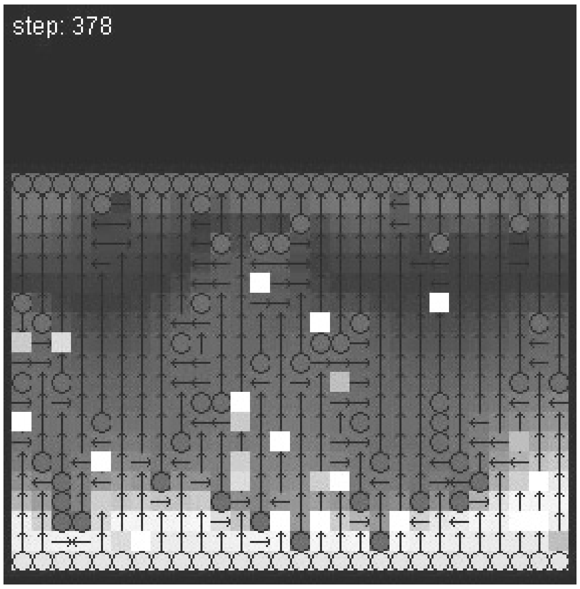


Fig. 26. Robust assembly with communication breakdown.

5.5. Robustness for Module Breakdown

In this simulation, communication among some modules is assumed to be severed. When modules detect communication breakdown, they cannot exchange potential values to update the gradient field (Figure 26 and Extension 7). Also, they cannot communicate with the assembler robots. This kind of error is more serious than the module removal, because the robot cannot restore the broken gradient field.

A quantitative comparison of robustness between these two types of disturbances is given in Figure 27. Communication breakdown always requires more time steps than module removal. This plot shows that when the number of dis-

3. Every module holds the values of the gradient field its neighbors have. The discretized direction (NÅCS, E, W) calculated by these values is displayed to the robot. When these values do not differ (i.e. difference between the maximum and minimum values is less than threshold), the module displays "no effective gradient" to the robot and the robot randomly decides its direction.

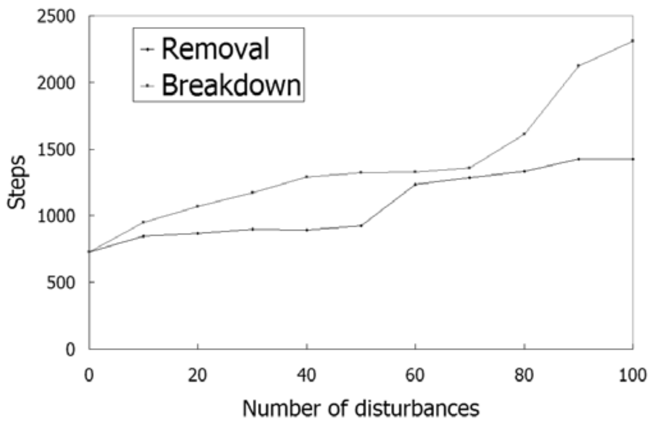


Fig. 27. Evaluating robustness.

turbances does not exceed some threshold (50 for module removal (4% error), 70 for communication breakdown (9% error)), efficiency is not much affected by the disturbances.

6. Flexible Construction with AMAS

6.1. Disassembly and Reassembly

In this section, we discuss a method to build temporary structures such as a scaffold to build over-hanged structures. Temporary structures should be assembled and disassembled quickly at low cost. Our structure modules are suited to this purpose, because the connector of the modules is designed without any permanent mounting mechanism. Disconnected modules are reusable and thus it is advantageous to reduce the waste and the material cost.

6.2. Algorithm of Disassembly

To disassemble the structure, the robots should be directed to the disassembly area and remove the modules to this area. This is naturally realized by the gradient field; just assigning the "module supply area" property to the modules in the disassembly area will do.

The disassembly proceeds as follows. When a module is assigned "to be removed" status, its adjoining modules become the "module supply area" (Figure 28). This is because a robot cannot remove a module it sits on. These modules attract the assembler robots by the gradient field, and eventually they are carried away to the assembly area.

In the disassembly process, the order of removal is important. If the modules in the disassembly area are removed randomly, it may break the connectivity of the whole structure.

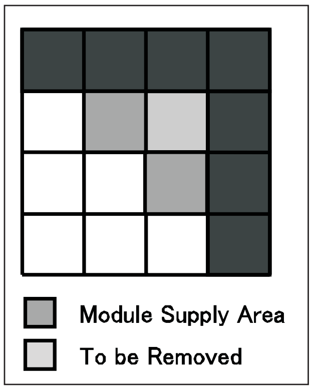


Fig. 28. Removal of a module.

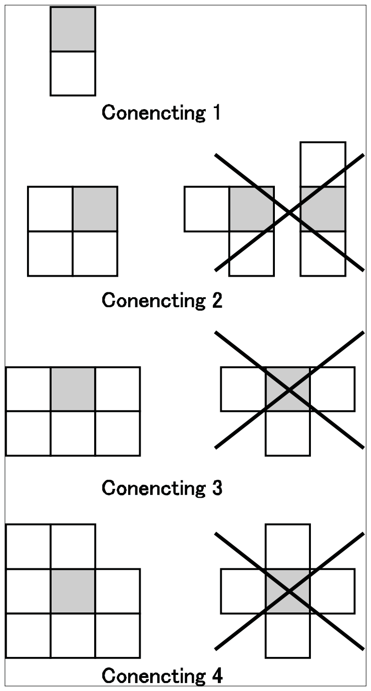


Fig. 29. Criteria of module removal.

6.3. Requirement to Keep Connection

In order to guarantee the connectivity during the disassembly, we introduce the connection number C defined on each module. It is the number of modules directly connected to a particular module. The criteria of module removal is as follows (see also Figure 29):

- $C = 1$: the module is removable;
- $C = 2, 3, 4$: the module is removable only when all of its neighbors will stay connected when the module is removed.

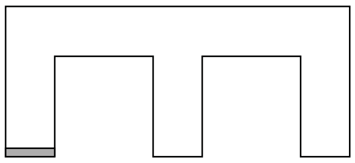


Fig. 30. Bridge.

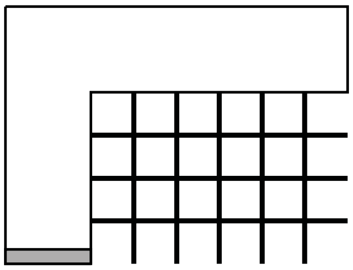


Fig. 31. Scaffold structure.

6.4. Bridge Construction using Disassembly and Reassembly

Arches and domes require temporary scaffolds during their construction. After the completion of these structures the scaffold must be removed. We give a demonstration of bridge construction (Figure 30) that utilizes the disassembly and reassembly capability of AMAS. We usually use a scaffold that supports the over-hanging part of the structure (Figure 31). In AMAS, we can do without such a scaffold, instead elevating the whole structure by removing modules beneath the beam and reassembling them on the bridge surface.

In this demonstration, we use several (three) intermediate target shapes as shown in Figure 32. When the assembler robots have completed the target shape of a stage, the target shape is switched to the next target shape. (This switching can be done by a centralized controller or by interaction between the robots by which they confirm the completion of the stage.)

To evaluate the effectiveness of the disassembly–reassembly capability, we conducted a simulation study. The target structure is a bridge of size 35 × 15 modules (Figure 33). Five assembler robots were deployed and they were controlled by the discrete gradient field with module relay. We have conducted ten trials and evaluated the average number of steps needed to complete the bridge. Normal assembly in one stage took 3,304 steps. (We allowed an over-hanging shape during the construction in this simulation, but it would be problematic in the real implementation.) In contrast, it took 3,010 steps utilizing the disassembly and reassembly procedure (see also Extension 8). A 10% reduction in time is achieved by reuse of the modules.

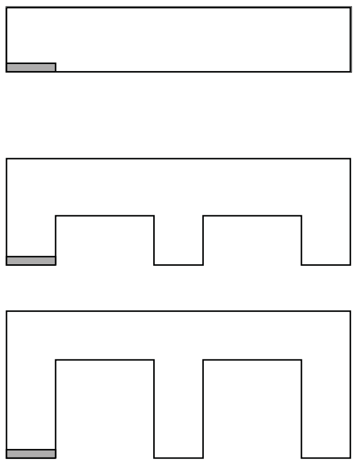


Fig. 32. Bridge assembly without a scaffold.

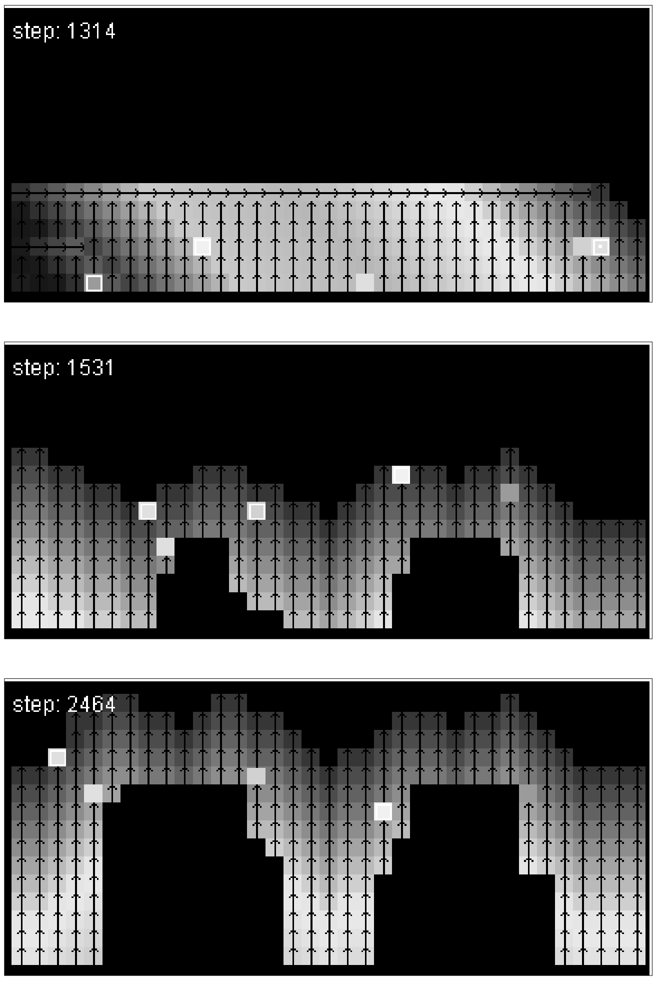


Fig. 33. Simulation of staged assembly.

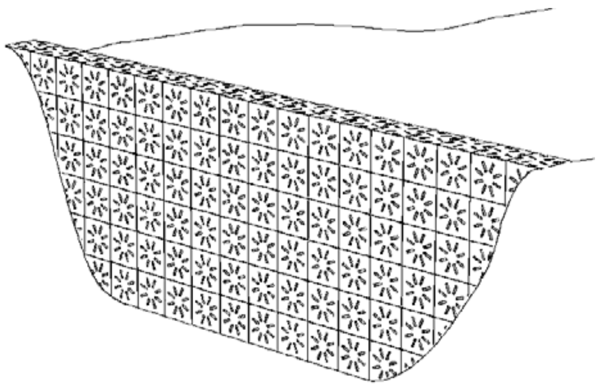


Fig. 34. Landslide barrier.

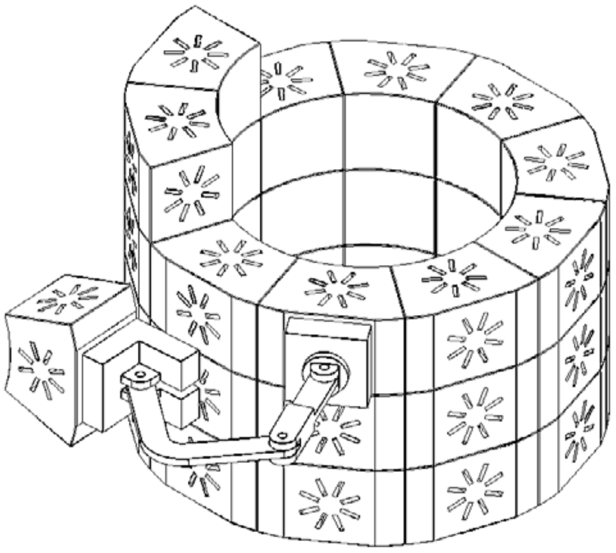


Fig. 35. Caisson.

7. Discussion and Application Images

Construction work of various structures in hazardous areas is a typical application of the proposed system. An automated structural assembly system that allows round-the-clock operation without exposing human workers to danger is highly desirable in such environments. The modular building can be used as a substitute for a road or bridge, or for a human shelter. Figure 34 shows this system being used as a landslide barrier.

Many other applications of this system are possible. For instance, the structural module can be used as a container; the modules can produce a warehouse system with high room efficiency and flexibility. We can modify the regular cube shape of the structural module in many ways. The slightly curved module shown in Figure 35 allows production of a cylindrical structure used as a caisson for bridge or pier construction. Extensible modules can be used to build a large space structure (Figure 36), where assembly is done before expanding the modules.

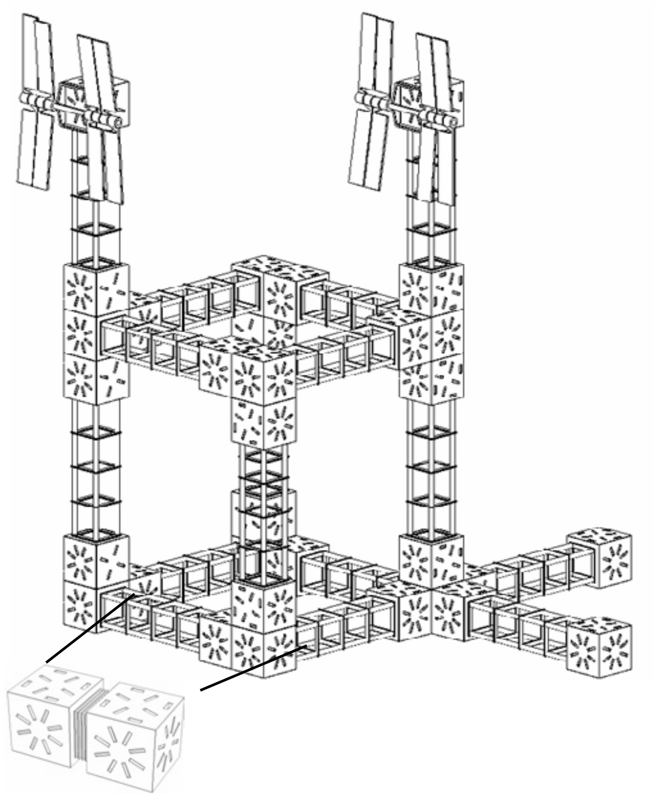


Fig. 36. Space structure.

8. Conclusions

We have proposed the Automatic Modular Assembly System (AMAS) in this paper. The system consists of cubic structure modules and assembler robots compatible with the modules. The assembler robots transport and assemble the modules to build a large-scale structure without any human intervention.

The coordination problems among multiple assembler robots have been considered and a distributed algorithm based on the gradient field has been proposed. Using the algorithm, completely decentralized control of the robot group is realized with the gradient field generated on the modular structure by inter-module communication. It is a kind of self-organizing system in the sense that robots build the structure and the structure controls the robots. Simulations of two-dimensional construction by a group of assembler robot have been conducted. Performances of the two kinds of gradient and two kinds of collision-avoidance methods are evaluated in terms of the efficiency for various densities of the robots.

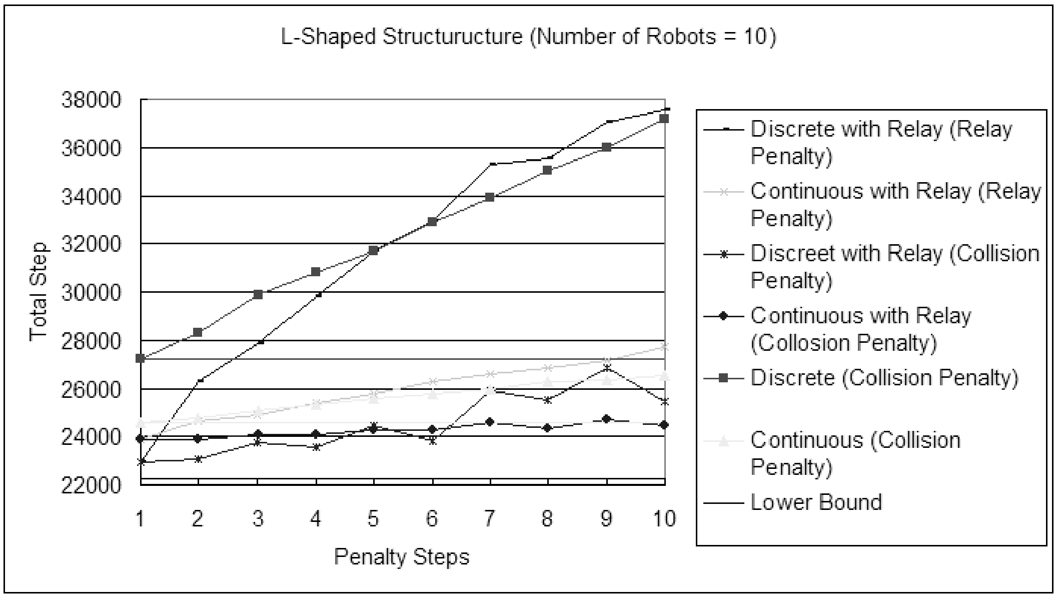


Fig. B.1. Influences of operation efficiency.

Appendix A: Possible Deadlock Situations in AMAS

Deadlock is a crucial problem in the distributed algorithm based on local and parallel information processing. Detecting the deadlock and escaping from it is an important issue in the distributed algorithm. Although AMAS is a distributed multi-robot system, it does not suffer from this kind of problem, because its algorithm is based on the gradient field, which does not have any local minima. However, we have to consider two types of deadlock caused by local conflicts among the robots:

- conflict in the robots' paths;
- conflict at the assembly (disassembly) site.

Hereafter, we consider the above conflicts by simulation and discuss method to avoid them in real implementation.

A.1. Conflict in the Robots' Paths

There are two types of this kind of deadlock: (1) conflict between two adjacent robots and (2) conflict between robots in the vicinity of each other (not adjacent).

(1) When two adjacent robots want to move to the same site, they can avoid collision by the priority assignment procedure stated in Section 4.4.

(2) Collision may happen among robots in the vicinity of each other. The same algorithm can again solve this by extending the range of inter-robot communication. In order to guarantee that there is no mechanical crash, communication ability with the robots within the vicinity of radius two is necessary.

Practically, collisions between a loaded robot (with a module) and an unloaded robot (without the module) are most commonly observed in the simulation. This type of collision can be solved by the module relay procedure.

A.2. Conflict at the Assembly (Disassembly) Site

This type of conflict occurs when two robots want to assemble (or remove) the same module. The priority algorithm can solve this situation (Section 4.4).

Appendix B: Influences of Operation Efficiency

The assembler robots perform different types of operation in the construction task. They have to move onto the module, turn to the left or right or 180°, picking up a module, placing a module and so on. In the simulation, we assume that all of these different operations require the same duration. Here, the feasibility of this simplification is discussed.

We can evaluate the effect of duration change for each operation. As an example, the L-shaped target shape built by ten robots is used in the following simulation. The parameters are the duration for the module relay and the penalty for collision. The result is shown in Figure B.1.

First, we compared the total steps required to complete the construction with and without the module relay. As the duration for the module relay increased, total step increases sharply for both the discrete and continuous gradient fields. These plots cross where penalty step is around two. Therefore, the module relay can be considered more efficient when it requires less than two time steps.

Next, we increased collision penalty steps and again compared the total steps required to complete the construction with or without the module relay. In the entire region, with the module relay is better than without. Also, if the penalty is equal to or less than four, the discrete field is better, while the continuous field is better for a penalty over five. In this region, the low rate of collision, which is the feature of the continuous field, is more effective than the effect of the module relay.

Appendix C: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org.

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Motion of the connection mechanism
2	Video	Assembly task using the prototype hardware
3	Video	Result of the simulation assembling an L-shaped structure a using continuous field
4	Video	Result of the simulation assembling a T-shaped structure using a continuous field
5	Video	Result of the simulation assembling an L-shaped structure using a discrete field with module relay
6	Video	Robust assembly with module removal
7	Video	Robust assembly with communication breakdown
8	Video	Assembling a bridge using disassemble and reassemble

Acknowledgement

This work is supported by Grant-in-aid for Scientific Research (B), # 16300051.

References

Arai, T. and Ota, J. (1992). Planning of multiple mobile robots. *Journal of Robotics Society of Japan*, **10**(4): 444–449.

Butler, Z., Kotay, K., Rus, D. and Tomita, K. (2004). Generic decentralized locomotion control for lattice-based self-reconfigurable robots. *International Journal of Robotics Research*, **23**(9): 919–938.

Borenstein, J. and Koren, Y. (1989). Real-time obstacle avoidance for fast mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics*, **19**(5): 1179–1187.

Chayama, K., Fujioka, H., Okano, M. and Mori, T. (2004). Implementation result of unmanned construction of a sabo dam using steel slit structure at Unzen Fugen-Dake. *Proceedings of the 10th Symposium on Construction Robotics in Japan*, pp. 123–142.

Chirikjian, G. (1994). Kinematics of a metamorphic robotic system. *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 449–455.

Dahl, T. S., Mataric, M. J. and Sukhatme, G. S. (2003). Multi-robot task-allocation through vacancy chains. *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, 2293–2298.

Drogoul, A. and Ferber, J. (1992). From Tom Thumb to the dockers: some experiments with foraging robots. *Proceedings of the 2nd International Conference on Simulation of Adaptive Behavior*, pp. 451–459.

Durna, M., Erkmen, A. M. and Erkmen, I. (2000). Self-localization of a holon in the reconfiguration task space of a robotic colony. *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pp. 1748–1754.

Farinelli, A., Iocchi, L. and Nardi, D. (2004). Multirobot systems: a classification focused on coordination. *IEEE Transactions on Systems, Man, and Cybernetics Part B: Cybernetics*, **34**(5): 2015–2028.

Fukuda, T. and Nakagawa, S. (1988). Dynamically reconfigurable robotic system. *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 3, pp. 1581–1568.

Hasegawa, Y. (1990). Present status and future themes of construction robot research and development. *Journal of Robotics Society of Japan*, **8**(2): 66–70.

Ikeda, Y., Shiokawa, T., Kawakami, H. and Miyajima, H. (2003). Development of an automated building construction system (part 3)—improvement of system and application of super high-rise hotel. *Technical Lab. Report No. 66*, Obayashi Corp.

International Federation of Robotics (IFR) (2005). *World Robotics 2005*.

Jorgensen, M., Ostergaard, E. and Lund, H. (2004) Modular Atron: modules for a self-reconfigurable robot. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2068–2073.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotic Research*, **5**(1): 90–98.

Kotay, K., Rus, D., Vona, M. and McGray, C. (1998). The self-reconfiguring robotic molecule. Proceedings of the IEEE International Conference on Robotics and Automation, pp. 424–431.

Kotay, K. and Rus, D. (2004). Generic distributed assembly and repair algorithms for self-reconfiguring robots. *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, Sendai, Japan.

Lund, H. H., Beck, R. and Dalgaard, L. (2005). ATRON hardware modules for self-reconfigurable robotics. *Proceedings of 10th International Symposium on Artificial Life and Robotics (AROB'10)*.

Mitsunaga, J. (2002). Construction machinery/robot business. *Journal of Robotics Society of Japan*, **20**(7), 696–700.

Murata, S., Yoshida, E., Kamimura, A., Kurokawa, H., Romita, K. and Kokaji, S. (2002). M-TRAN: self-reconfigurable modular robotic system. *IEEE/ASME Transactions on Mechatronics*, **7**(4): 431–441.

Nilsson, M. (2002). Connectors for self-reconfiguring robots. *IEEE/ASME Transactions on Mechatronics*, **7**(4): 473–474.

Ota, J., Arai, T. and Yokogawa, Y. (1994). Distributed strategy-making method in multiple mobile robot system. *Proceedings of the 2nd International Symposium on Distributed Autonomous Robotic Systems (DARS02)*, pp. 103–106.

Rus, D. and Vona, M. (2001). Crystalline robots: self-reconfiguration with compressible unit modules. *Autonomous Systems*, **10**(1): 107–124.

Shen, W., Selemi, B. and Will, P. (2002). Hormone-inspired adaptive communication and distributed control for CONRO self-reconfigurable robots. *IEEE Transactions on Robotics and Automation*, **18**: 700–712.

Stoy (2002).

Stoy, K. (2006). Using cellular automata and gradients to control self-reconfiguration. *Robotics and Autonomous Systems*, **54**: 135–141.

Stoy, K., Shen, W. and WillStoy, P. (2003). A simple approach to the control of locomotion in self-reconfigurable robots. *Robotics and Autonomous Systems*, **44**(3–4): 191–199.

Stoy, K. and Nagpal, R. (2004). Self-repair through acale independent self-reconfiguration. *Proceedings of IEEE/RSJ International Conference on Robots and Systems (IROS)*, Sendai, Japan, 30 September–2 October, pp. 2062–2067.

Todt, E., Raush, G. and Sukez, R. (2000). Analysis and classification of multiple robot coordination methods. *Proceedings of the 2000 IEEE, International Conference on Robotics & Automation*, pp. 3158–3163.

Terada, Y., Kurokawa, H. and Murata, S. (2004). Concept of automatic assembly system for a large modular structure. *Proceedings of the 8th International Conference on Intelligent Autonomous Systems (IAS-8)*, pp. 739–745.

Terada, Y. and Murata, S. (2006a). Modular structure assembly using gradient field. *Proceedings of 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2006)*, pp. 604–611.

Terada, Y. and Murata, S. (2006b). Modular structure assembly using blackboard path planning system. *Proceedings of 22nd International Symposium on Automation and Robotics in Construction (ISARC 2006)*, pp. 852–857.

Unsal, C., Kiliccote, H. and Khosla, P. K. (2001). A modular self-reconfigurable bipartite robotic system: implementation and motion planning. *Autonomous Robots*, **10**(1): pp. 23–40.

Werfel, J. and Nagpal, R. (2006a). Three-dimensional directed construction. *Proceedings of the Workshop on Self-Reconfigurable Modular Robots, at Robotics: Science and Systems II*, Philadelphia, PA.

Werfel, J. and Nagpal, R. (2006b). Extended stigmergy in collective construction. *IEEE Intelligent Systems*, **21**(2): 20–28.

Whittaker, W. L., Urmson, C., Staritz, P., Kennedy, B. and Ambrose, R. (2000). Robotics for assembly, inspection, and maintenance of space macrofacilities. *Proceedings of AIAA Space 2000*, Long Beach, CA, paper AIAA-2000-5288.

Yim, M., Zhang, Y., Roufas, K., Duff, D. and Eldershaw, C. (2002). Connecting and disconnecting for chain self-reconfiguration with Polybot. *IEEE/ASME Transactions on Mechatronics*, **7**(4): 442–451.

Yoshimura, Y., Ota, J., Inoue, K., Kurabayashi, D. and Arai, T. (1996). Iterative transportation planning of multiple objects by cooperative mobile robots. *Distributed Autonomous Robotic Systems*, **2**: 171–182.

Zykov, V., Mytilinaios, E., Adams, B. and Lipson, H. (2005). Self-reproducing machines. *Nature*, **435**(7038): 163–164.