Available online at www.sciencedirect.com
SCIENCE DIRECT
Robotics and Autonomous Systems 54 (2006) 193–204
Robotics and Autonomous Systems
www.elsevier.com/locate/robot

# Multi-robot cooperation-based mobile printer system

Kang-Hee Lee*, Jong-Hwan Kim

*Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology, Guseong-dong, Yuseong-gu, Daejon, 305-701, Republic of Korea*

Received 24 March 2005; received in revised form 3 November 2005; accepted 24 November 2005
Available online 24 January 2006

**Abstract**

This paper proposes a mobile printer system (MPS) based on multi-robot cooperation. The system consists of multiple mobile robots, a wireless LAN system, a graphic user interface (GUI), and a host computer. The GUI comprises a user input section, a task allocation optimization section, and a control and communication section. Its operation is as follows: a user draws a picture on an input window of the GUI, and then the host computer commands client printer-robots to reproduce the same on a paper in a finite time.

To control multiple robots during this process, two kinds of multi-robot control architectures along with a collision-free arbitration configuration are proposed. One is a decentralized control architecture, which employs subsumption architecture based on behavior-based robotics. The robots continue to seek the nearest line and to draw it repeatedly until all lines are drawn. This architecture needs no pre-planning and is fault-tolerant. Another is a centralized control architecture, which employs an evolutionary algorithm (EA). The host computer optimizes the task (finding time-optimal path) allocation for each robot by using an evolutionary algorithm and sends the optimized job sequence to the robots. To minimize the elapsed time in drawing all the lines, an evolutionary algorithm with a representation of an individual suitable for the MPS is employed for the task optimization. This architecture can minimize the elapsed time effectively and offers the option of distance-optimality in addition to time-optimality.

The proposed architectures are simulated and real experiments with three omni-directional robots are carried out to demonstrate the effectiveness and the applicability of the proposed mobile printer system.


*Keywords:* Mobile printer; Multi-robot cooperation system; Multi-agent system; Evolutionary robotics; Behavior-based robotics

## 1. Introduction

Due to miniaturization and decentralization in robotics, modern robots show performance superior to previous giant robots in spite of their small size. Further, multiple cooperative robots can perform more difficult operations requiring much complexity and time more efficiently than a single robot. A system like this that is composed of multiple robots is called a "multi-robot system" or "multi-robot cooperation system". A multi-robot cooperation system offers the advantage of acceleration of the performance of a given task due to parallel processing and coordination among robots. Also, it is robust against failure due to the presence of multiple robots. If only one robot is employed to carry out a given task and that robot breaks down, the assigned task cannot be finished properly. In a multi-robot cooperation system, however, the remaining robots can substitute for the faulty robot to complete the given task, because the system has scalability characteristics based on modularity.

Thus, this paper intends to convert the static concept of the conventional printer into the dynamic concept of a mobile printer system (MPS) using a multi-robot cooperation system. The MPS can print several kinds of lines and pictures without being restricted to the size of the printing region and type of printed surface. It also has control architectures to optimize the task-allocation for each robot using evolutionary computation and behavior-based robotics. As this concept has not been proposed previously, it is convenient to classify related studies into the following topics: first, tasks of painting or working by a robot on a canvas, ground, or floor; second, evolutionary optimization for task-allocation in a multi-robot system; and

* Corresponding author. Tel.: +82 42 869 3448; fax: +82 42 869 8877.
*E-mail addresses:* khlee@rit.kaist.ac.kr (K.-H. Lee), johkim@rit.kaist.ac.kr (J.-H. Kim).

0921-8890/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2005.11.005

third, behavior-based robotics, which needs no optimization for task-allocation in a multi-robot system.

With regard to tasks of painting or working by a robot on a canvas, ground, or floor, Hertling [1] and many researchers [2–4] studied robots engaged in spray painting of an arbitrary surface, such as automobiles, furniture, and appliances. For example, Penin [5] developed a system performing automatic task and path planning for a spraying robot to manufacture a cell of pre-fabricated glass reinforced cement for the construction industry. On the other hand, the following studies deal with tasks of painting or tasks on the ground or floor using a mobile robot. Kotani [6] proposed a lane mark drawing robot for detecting, following, repainting and painting traffic marks on an asphalt road using image processing. Jung [7], a study which provided motivation for this paper, designed an omni-directional robot, Omni-Kity I. Its effectiveness was verified experimentally; mounting a pen on the robot's body, the given lines were drawn according to the trajectory of Omni-Kity I.

The second related topic, evolutionary optimization for task-allocation in multi-robot systems, has been studied as partitioning problems or grouping problems which require partitioning $n$ objects into $k$ categories [8] without inclusion of robots. This class contains many well-known problems, such as bin packing problems (assigning items to bins), graph coloring problems (assigning nodes of a graph to specific colors), etc. To solve these problems by evolutionary computation, very important factors must be considered such as the chromosome format representing all objects as a permutation list, special operators which can be applied to it, a decoder making the decisions on the allocations, and the evaluation fitness or cost function to evaluate these [9]. For example, Von Laszewski [10] proposed group-number encoding, as well as structural crossover and structural mutation. Jones [11] adopted different representations and operators using the group-number encoding in their evolution program. Although evolutionary optimization for task-allocation has been studied as a kind of grouping problem in this manner, in reality, there have been few studies on improvement of operation efficiency by applying this grouping problem to a multi-robot system as in the present work. Instead, most research has focused on evolving formation wherein a group of multiple robots continues to perform a task effectively [12,13]. Liu [14] proposed a new format of a chromosome and its fitness function to evolve behaviors for a group of distributed autonomous robots in order to cooperatively push an object toward a goal location. Sheng [15] modeled a robot motion planner problem on a partitioning problem using a decomposition-based approach in manufacturing processes such as spray coating and spray painting using a single robotic manipulator.

The third related topic, behavior-based robots, does not require pre-planning of robot behaviors, because the robots are reactive only to external stimuli without interruption of abstract expression or past events. The subsumption architecture used in this paper was first developed by Rodney Brooks, who introduced a coordination process used between layered behaviors within the architecture [16] and generally has two main considerations such as priority problems and communication protocols. Complex actions subsume simpler behaviors and a priority hierarchy fixes the topology. The lower levels in the architecture have no awareness of the higher levels, which provides the basis for the incremental design. Priority-based arbitration via inhibition and suppression is used as a coordination method and the response encoding follows the rule-based method [17]. Multi-robot cooperation systems using this subsumption architecture have the strengths of hardware retargetability (subsumption can compile down directly onto programmable-array logic circuitry), support for parallelism (each behavioral layer can run independently and asynchronously) and niche targetability (custom behaviors can be created for specific task environment pairs). However, they have drawbacks related to runtime flexibility (because of the architecture's hard-wired aspects) and support for modularity (behaviors cannot always be prioritized). Considering these characteristics, this paper seeks to perform the mobile printing task fast and robustly without pre-planning by combining the strengths of subsumption-based robots and a multi-robot system.

This paper proposes a new system using multi-robot cooperation, referred to as a mobile printer system (MPS), along with its control architecture. The MPS prints letters or pictures input through a GUI by the user. The size of the printing region is not restricted because of the multiplicity of mobile printer robots. The MPS proposed in this paper consists of three omni-directional robots [19], a wireless LAN system, a GUI, and a host computer. In order to achieve the objective stated earlier, a plurality of mobile printer robots has an in-built printer function, e.g. a pen is mounted on the rear of each robot. The host computer includes the GUI function through which a user can input letters or pictures. It optimizes the amounts of tasks allocated to each robot through an evolutionary algorithm (EA) and also arbitrates a collision-avoidance scheme among robots through wireless communication.

Its operation is as follows: a user draws a picture on an input window of the GUI, and then the host computer commands client printer-robots to reproduce the same on paper in a finite time. To control multiple robots during this process, two kinds of multi-robot control architectures along with a collision-free arbitration configuration are proposed: (i) a decentralized control architecture that employs subsumption architecture based on behavior-based robotics. The robots continue to seek the nearest line and to draw it repeatedly until all lines are drawn. This architecture needs no pre-planning and is fault-tolerant. (ii) A centralized control architecture that employs an evolutionary algorithm (EA). The host computer optimizes the task (finding time-optimal path) allocation for each robot by using an evolutionary algorithm and sends the optimized job sequence to the robots. To minimize the elapsed time in drawing all the lines, an evolutionary algorithm with a representation of an individual suitable for the MPS is employed for the task optimization. This architecture can minimize the elapsed time effectively and provides the option of distance-optimality as well as time-optimality. To demonstrate the effectiveness and the applicability of the proposed mobile printer system, simulations and real experiments are carried out.

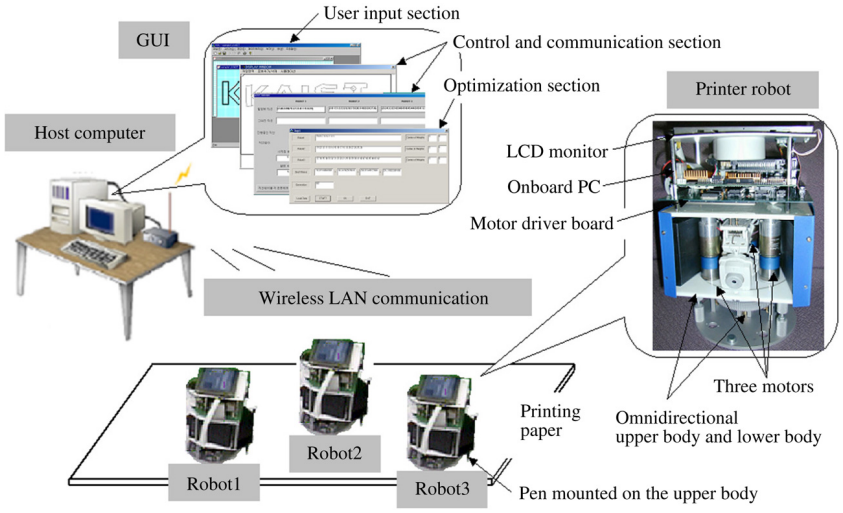



Fig. 1. Structure of the mobile printer system.

The remainder of this paper is organized as follows. In Section 2, the mobile printer system (MPS) is presented. Section 3 proposes two control architectures for the MPS and an arbitration architecture for collision avoidance among the robots. Sections 4 and 5 describe the simulation and the experimental results with the MPS. Finally, concluding remarks follow in Section 6.

## 2. Mobile printer system (MPS) using multi-robots

The main objective of the MPS design using multi-robots is to print user's drawings without being restricted by the size of the printing region or the printed surface type. The static concept of the conventional printer can be converted into the dynamic concept of a mobile printer system in this fashion. The apparatus shown in Fig. 1 comprises a plurality of printer-robots and a host computer that controls their operation. The host computer optimizes the amount of tasks such that they are equally assigned to each robot by using the evolutionary algorithm (EA) and arbitrates collision avoidance among robots during the printing task. The host computer further includes a graphic user interface (GUI), which comprises three sections: (i) *a user input section*, which receives the input data from the user and converts it to operation data that enables the printing process; (ii) *a task allocation optimization section*, which optimizes the amount of tasks for each of the printer-robots; and (iii) *a printer-robot control and communication section*, which arbitrates the collision-free paths among the printer-robots, enables data transmission and reception among the robots, and displays their states at every sampling time. The printer-robots used in this paper are omnidirectional mobile robots (OMR); that is, they can move in any direction in the configuration space [19]. On a 2D plane, the configuration of a mobile robot is determined by its position $(x, y)$ and orientation $\theta$ with respect to a global coordinate system. Therefore, an OMR moving on a 2D plane must be able to steer its $\dot{x}$, $\dot{y}$, and $\dot{\theta}$ in any direction. A brief illustration of the specifications of the robot will follow in Section 5.

Typically, measures of MPS performance are printing time and printing quality. In this paper, we focus on printing time, assuming that printing quality is the same. Hence, the aim of the MPS here is to finish the printing task in the shortest possible time.

When the MPS using multi-robot configuration is implemented, the following issues are considered: (i) how to generate a collision-avoidance mechanism among robots; (ii) how to complete the printing task in the shortest possible time; and (iii) how to distribute the task equally among the robots.

Computer simulations and evolutionary optimization are carried out with the following conditions: (i) letters and pictures can be segmented by lines; (ii) the ability of each robot is equal; (iii) each robot knows its absolute position by dead-reckoning using encoder signals; (iv) the time taken to optimize task allocation is not part of the printing time; (v) internet time delay and the turning time of each robot are not considered; (vi) the reference to total printing time is the time required for the robot to finish its task completely.

In the simulator, each robot has a manipulator, which can move up and down, for printing. In particular, note that condition (i) means that the format of a sample sent as an input in the GUI consists of line segments, where the pixel size and length are set by the user. For example, to input a circle or an ellipse, the user should make some approximation by using a polygon. This simplifies the task allocation.

## 3. Control architecture for mobile printer system

In this section, two kinds of control architectures for the mobile printer system are described: a decentralized control architecture (DCA), which uses a behavior-based method, and a centralized control architecture (CCA), which employs an evolutionary algorithm (EA) to assign tasks to each robot optimally. The arbitration architecture for collision avoidance among the printer-robots is also introduced.

### 3.1. Decentralized control architecture

This section presents a multiple subsumption architecture in which each robot stimulates other robots through wireless

Table 1
Line look-up table

| Line index | Starting-point | Ending-point | Drawn-line flag | Drawing-line flag |
|---|---|---|---|---|
| 1 | $(x_{1s}, y_{1s})$ | $(x_{1e}, y_{1e})$ | *false* | *false* |
| 2 | $(x_{2s}, y_{2s})$ | $(x_{2e}, y_{2e})$ | *false* | *false* |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| *n* | $(x_{ns}, y_{ns})$ | $(x_{ne}, y_{ne})$ | *false* | *false* |

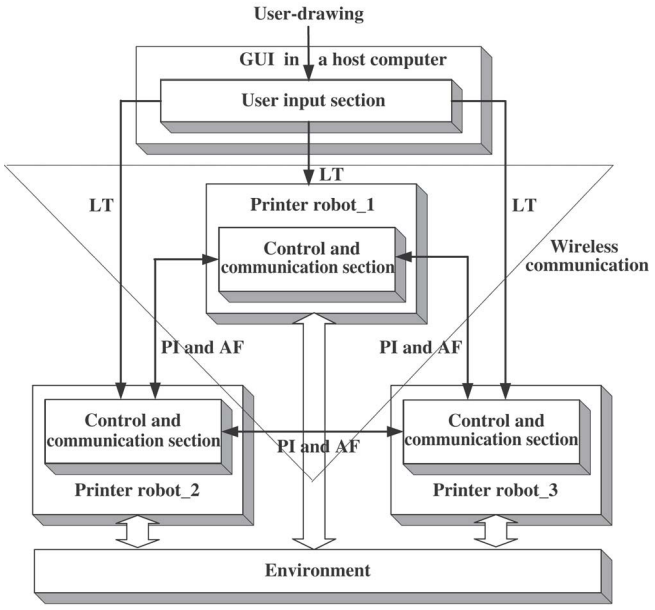



Fig. 2. Decentralized control architecture for the MPS with three robots.

communication in order to share printing information. Fig. 2 shows the case of three robots employed in the MPS. Here a decentralized control architecture is employed, where each robot is autonomous and homogeneous and its behavior is based on a multiple subsumption architecture.

A user draws a picture on *a user input section* of the GUI, and then the *user input section* transmits the look-up table of input lines to each robot and commands it to start printing. As shown in Table 1, the line look-up table consists of four components for each line index. Each time a user draws a line, the line index increases by one and the positions of both ends of each line are saved in the line look-up table. Finally, the total number of input lines becomes *n* and the line look-up table will have *n* indices. The initial values of *drawn-line flag* and *drawing-line flag* of all the lines are set to "*false*". The values are thereupon changed to "*true*" through behavioral rules, which will be discussed later. This look-up table will be also used in the centralized control architecture in Section 3.2.

During the printing process, each robot communicates with other robots and the host computer by a simple protocol, as shown in Fig. 3, through messages such as *drawn-line index*, *drawing-line index*, *pose information*, *stop flag*, and *move backward flag* for display and collision avoidance. Message ID

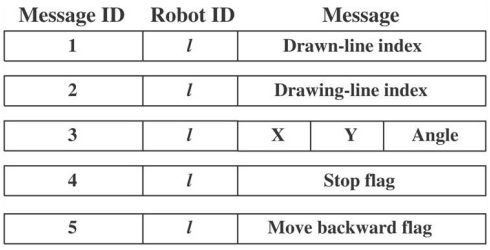



Fig. 3. Communication protocol for printing information transfer and collision-avoidance arbitration in the multi-robot control architecture for the MPS. Message IDs 1, 2, and 3 are sent to the host computer by Robot_*l*. Message IDs 4 and 5 are sent to Robot_*l* by the main commander robot in DCA or the host computer in CCA.

1 indicates that Robot_*l* has finished drawing the line of *drawn-line index* and this message is sent to the host computer and other robots. Message ID 2 indicates that Robot_*l* will draw the line of *drawing-line index* and this message is sent. Through message ID 3, which is sent by Robot_*l*, the host computer and each robot always know the positions and postures of other robots at every sampling time. By sending message ID 4 and message ID 5, the main commander robot commands Robot_*l* to stop or to move backward for collision avoidance. When arbitration of collision avoidance among robots is necessary, the first robot that recognizes such a state becomes the main commander robot of the arbitration. If it develops to MPS that uses many robots in the future, there will be a condition where two or more robots meet the same task simultaneously and become the commander together. To solve this problem, specific task priority and communication rate among robots should be considered a priori. To perform these autonomous arbitrations, each robot has a *control and communication section* for collision avoidance. The detailed arbitration process will be illustrated in Section 3.3.

Fig. 4 illustrates the multiple subsumption architecture for the MPS with three robots which the decentralized control architecture is based on. The followings are the behaviors each robot utilizes:

- **Line-searching**: determine the starting point of the nearest line by examining the distances from the present position of the robot itself to the starting point or the ending points of all lines. Then, if the ending point is closer than the starting point, substitute the ending point with the starting point.
- **Moving-to-goal**: after determining the nearest line, set its *drawing-line flag* to "*true*", move to it, and provide this information to the other robots (Message ID 2) so that they may search other lines excluding this one.

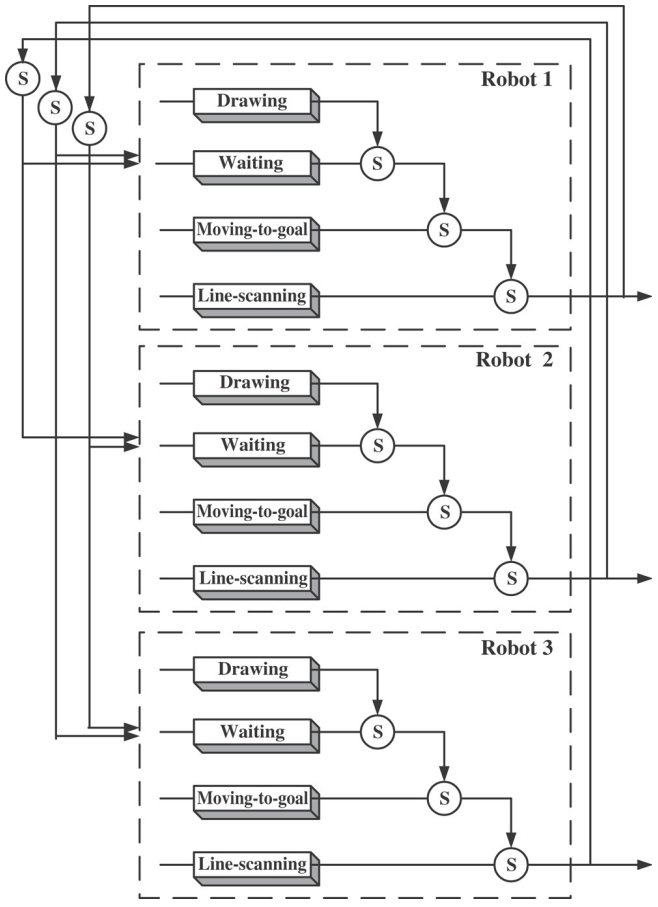



Fig. 4. Multiple subsumption architecture for the MPS with three robots.

- **Waiting**: stop according to the command of other robots for collision avoidance (Message ID 4).
- **Drawing**: draw the found line. At the ending point, set *drawn-line flag* of the drawn line to "*true*" and send this information to the other robots (Message ID 1).

Through the chain of the above behaviors, the found line is drawn successfully and its *drawing-line flag* and *drawn-line flag* subsequently become "*true*". The main concept of the subsumption architecture was employed in [17] and here it is expanded to a Stimulus-Response (SR) structure by multiple robots. Each behavior in the system is encoded as a set of rules. In the original SR diagram, priority-based arbitration is the coordination mechanism, and the robot executes only one behavioral rule at any time. Likewise, considering only one robot in Fig. 4, note in particular that when the robot finds the nearest line, Line-searching is suppressed; when the nearest line is drawn, Drawing suppresses Waiting (sequentially Waiting suppresses Moving-to-goal and then Moving-to-goal suppresses Line-searching). However, in the multiple subsumption architecture, Line-searching at each robot can also suppress Waiting of the other robots through the wireless LAN communication to avoid collisions among robots. By these simple behavioral rules, the mobile printing task can be completed successfully without pre-planning.

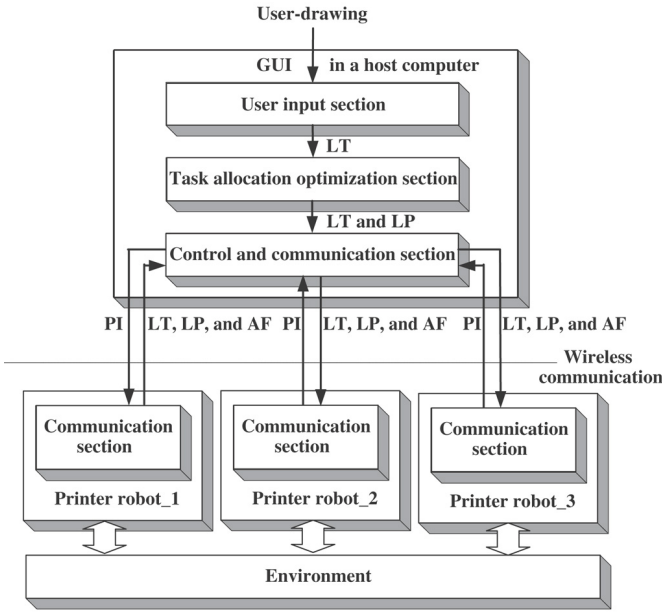



Fig. 5. Centralized control architecture for the MPS with three robots.

### 3.2. Centralized control architecture

This section presents a centralized control architecture for the MPS employing an EA for task-allocation to each robot. Fig. 5 shows the control architecture for the MPS with three robots.

The user draws a picture on *a user input section* of the GUI, and then the *user input section* transmits the look-up table corresponding to the input lines to the *task allocation optimization section*, which calculates time-optimal paths for each robot by the proposed EA for the MPS. It then sends the paths to each of the robots through the data sender in the *printer robot control and communication section*.

During the printing process, each robot communicates with the host computer by the simple protocol, as shown in Fig. 3, through messages such as *drawn-line index*, *drawing-line index*, *pose information*, *stop flag*, and *move backward flag* for display and collision avoidance. Message ID 1 indicates that Robot_*1* has finished drawing the line of *drawn-line index* and sends the message to the host computer. Message ID 2 indicates that Robot_*1* will draw the line of *drawing-line index* and sends the message. Through message ID 3, which is sent by Robot_*1*, the host computer always knows the positions and postures of the robots at every sampling time. By sending message ID 4 and message ID 5 the host computer commands Robot_*1* to stop or to move backward for collision avoidance. To perform the centralized arbitration, the host computer has a *control and communication section* for collision avoidance and can perform arbitration among robots through this protocol. The detailed scheme arbitration process will be illustrated in Section 3.3.

As mentioned earlier, the proposed centralized control architecture for the MPS employs an EA for task-allocation

```
procedure the EA for the MPS
begin
    t ← 0
    initialize P(t)
    evaluate P(t)
    store the best solution b among P(t)
    while (not termination-condition) do
    begin
        t ← t + 1
        select P(t) from P(t − 1)
        alter P(t)
        evaluate P(t) heuristically
        store the best solution b among P(t)
    end
end
```

Fig. 6. Procedure the EA for the MPS.

and adopts a heuristic method to evaluate each allocated task internally. The EA is a stochastic search algorithm that maintains a population of individuals, $P(t) = \{\mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_m^t\}$ at iteration $t$. Each individual represents a potential solution to the problem at hand and, as in any EA, is implemented as some (possibly complex) data structure $S$. Each individual $\mathbf{x}_i^t$ is evaluated to give some measure of its "fitness". Then, a new population (at iteration $t + 1$) is formed by selecting the fitter individuals (selection step). Some members of the new population undergo transformations (alternation step) by means of "genetic" operators to form new individuals. Mutation transformation $m_i$ creates new individuals by a small change in a single individual ($m_i$: $S \rightarrow S$), where $S$ is a data structure of individuals. Crossover transformation $c_j$ creates a new individual by combining parts from several (two or more) individuals ($c_j$: $S \times \cdots \times S \rightarrow S$). After a small number of generations, the program converges and the best individual should represent a near-optimum (reasonable) solution [9].

The EA for task-allocation to each robot is summarized as shown in Fig. 6. The EA for the MPS maintains a population of individuals, $P(t) = \{\mathbf{x}_1^t, \mathbf{x}_2^t, \ldots, \mathbf{x}_m^t\}$ at generation $t$, where $m$ is the size of the population, and $\mathbf{x}_j^t$ is an individual defined as

$$\mathbf{x}_j^t = (i_1, i_2, \ldots, i_n), \tag{1}$$

where $n$ is the total number of lines to be drawn by the robots, i.e., the string length of the individual or the number of genes, and $j = 1, 2, \ldots, m$. $\mathbf{x}_j^t$ is a permutation of $\{1, 2, \ldots, l\}$, where $l$ is the total number of robots and $i_k$, $k = 1, 2, \ldots, n$, is the index of the corresponding robot to draw the $k$th line segments. Hence, $\mathbf{x}_j^t$ represents one individual in the population of "$m$" individuals, with a total $n$ line segments to be drawn by a total of $l$ robots, where $i_k$, $k = 1, 2, \ldots, n$, represents one robot that will draw the $k$th line segment. For example, for the case of $l = 3$, if an individual is $(1, 1, 2, 2, 1, \ldots, 3, 3, 2, \ldots, 2, 3)$, the 1st, 2nd, and 5th lines are grouped into *Group*_1, which should be drawn by Robot_1. The 3rd and 4th lines are grouped into *Group*_2, drawn by Robot_2. The same procedure is also applied to Robot_3. For simplicity, the case of three robots ($l = 3$) is considered in the codes in Fig. 7.

```
procedure evaluate (x)
begin
    Cost_1, Cost_2, Cost_3 ← 0
    best Cost ← 0
    best population ← x_1^t
    j ← 1
    while j ≤ m do
    begin
        k ← 1
        while (k ≤ n) do
        begin
            group i_k of x_j^t by the same genes
        end
        Cost_1 ← heuristics(Group_1)
        Cost_2 ← heuristics(Group_2)
        Cost_3 ← heuristics(Group_3)
        Cost ← Max(Cost_1, Cost_2, Cost_3)
        if best Cost > Cost then
        begin
            best Cost ← Cost
            best population ← x_j^t
        end
    end
end
```

Fig. 7. Procedure evaluate.

In the step of "evaluate $P(t)$", Cost of each individual in a population is measured. The goal of the EA is to minimize *Cost*, defined as $Cost = \text{Max}(Cost\_1, Cost\_2, Cost\_3)$, where *Cost*_1 is the total trajectory distance that Robot_$l$ moves in order to draw the allotted line segments, which is computed by the heuristic method. Here the elapsed time required for each robot to draw its line segments is proportional to each trajectory distance. After the task-allocation part groups the index of lines by the same gene for one particular individual, each group is transferred to the heuristic function, as shown in Fig. 8, where the operator $D$ (*position a*, *line b*) denotes the distance from *position a* to the end point on *line b*, i.e., whichever is closest to *position a*. The operator num($x$) means the number of elements in the set $x$. As shown in Fig. 9, a line path for each robot is created by codes in Fig. 8. The procedure heuristics searches for the nearest lines in sequence starting from the initial position for Robot_1. Then, if the end point of the nearest line is closer than its start point, the end point is exchanged with the start point of the nearest line before proceeding. In *heuristics* function Robot_1 searches *Group*_1 only for the next nearest line from the end point of the line already drawn. The search continues until all the lines in *Group*_1 join the path of Robot_1. The same procedure is applied to *Group*_2 and *Group*_3, repeatedly.

In the step of "select $P(t)$ from $P(t-1)$", the roulette wheel method or any other selection method can be used [9].

In the step of "alter $P(t)$", offsprings (new parental generations) are made by crossover and mutation. In

```
procedure heuristics (Group_1) returns total distance
begin
    total distance ← 0
    p ← 1
    oldEndPose ← InitialPose(Robot_1)
    while p ≤ num(Group_1) do
    begin
        find the nearest line in Group_1 from oldEndPose
        distance ← D(oldEndPose, nearest line)
        total distance ← total distance + distance
        oldEndPose ← EndPose(nearest line)
        p ← p + 1
    end
    store the line path of lines in Group_1 created by above procedure
end
```

Fig. 8. Procedure heuristics.

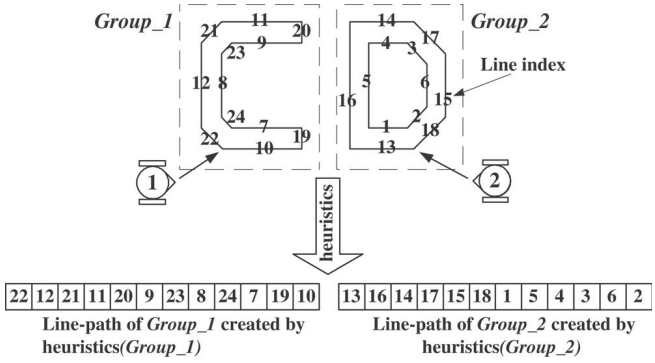



Fig. 9. Line paths created by heuristics function in sample example "CD" where the total number of inputted lines is 24.

simulations and experiments, two-point crossover and four kinds of mutations, i.e., inversion, insertion, displacement, and reciprocal exchange, were used.

### 3.3. Arbitration architecture for collision avoidance

The most important issue in motion planning for multiple robot cooperation is the collision avoidance among robots. In this paper, the developed MPS uses no sensors and implements the collision avoidance by wireless communication among robots. The arbitration operates for collision avoidance between two robots, if $D_{rr} \leq 3R$, where $D_{rr}$ is the distance between two robots and $R$ is the radius of each robot. Arbiters for the arbitration are included in a host computer and each printer-robot. When arbitration of collision avoidance among robots is necessary, the first robot that recognizes such a state becomes the main commander or arbiter in the decentralized control architecture. On the other hand, the host computer becomes the arbiter in the centralized control architecture.

Figs. 10 and 11 show cases when arbitrations are required. Note that in Figs. 10 and 11, Robot_1 is assumed to have a shorter trajectory distance from the starting point than Robot_2. As shown in Fig. 10, most cases in mobile printing correspond to general cases, where Robot_1 has a priority over Robot_2 and

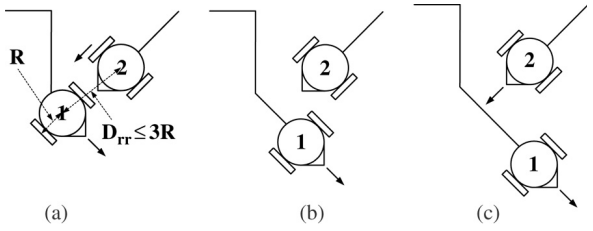



Fig. 10. A general arbitration.

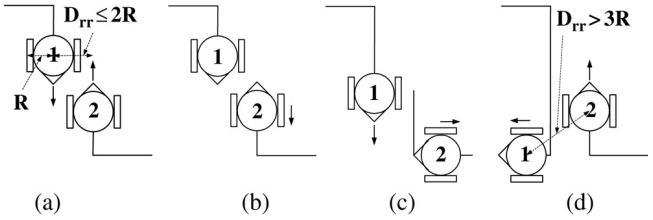



Fig. 11. A special arbitration.

Robot_2 has to wait until Robot_1 passes by it. Otherwise, other cases correspond to special cases, where Robot_2 is within the range of Robot_1, for example, as shown in Fig. 11(a). As illustrated in Figs. 10 and 11, the arbitration procedure between Robot_1 and Robot_2 can be summarized as follows:

Step 1. Check trajectory distances that two robots have moved before the arbitration commences (Fig. 10(a)).

Step 2. If one robot has a shorter trajectory than the other robot, the former (Robot_1) that has priority over the latter (Robot_2) continues to move, but the latter (Robot_2) stops by message ID 4 until the possibility of collision has been eliminated (Fig. 10(b)).

Step 3. If Robot_2 confronts Robot_1 and is within the range of Robot_1 (minimum $D_{rr} \leq 2R$) as shown in Fig. 11(a), the host computer or the main commander robot commands Robot_2 to move backward to proceed out of the range of Robot_1 through message ID 5 (Fig. 11(b) and (c)).

Step 4. After the possibility of collision is eliminated ($D_{rr} > 3R$), Robot_2, which has stopped or moved backward, returns to its original position and resumes performance of its task (Figs. 10(c) and 11(d)).

## 4. Computer simulations

Fig. 12 shows two samples for printing drawn in the input window of the GUI. Sample1 has 48 line segments in total; the sum of lengths of the segments is 591.7 cm and the size of its workspace is approximately 150.0 cm × 30.0 cm. Sample 2 has 106 line segments in total, the sum of which lengths is 652.0 cm and the workspace size is approximately 100.0 cm × 50.0 cm. Since these sums include only the actual lengths of lines, the total length of optimal trajectories cannot be shorter than these values. The samples and the environmental conditions are equally applied to both control architectures.

This paper focused not on the speed to find the optimal solution but on finding a more optimal solution even if the optimization takes more time. Regarding the robotics aspects, even if issues of localization, precision of printing and so on are finally solved, two factors such as the total trajectory distance and the elapsed time must still be critical factors in a mobile printing task using multiple robots.

Table 2 shows the simulation results of the control architectures for the MPS with sample 1 and sample 2, respectively. A Pentium-III 800 MHz was used, running with Visual C++6.0.

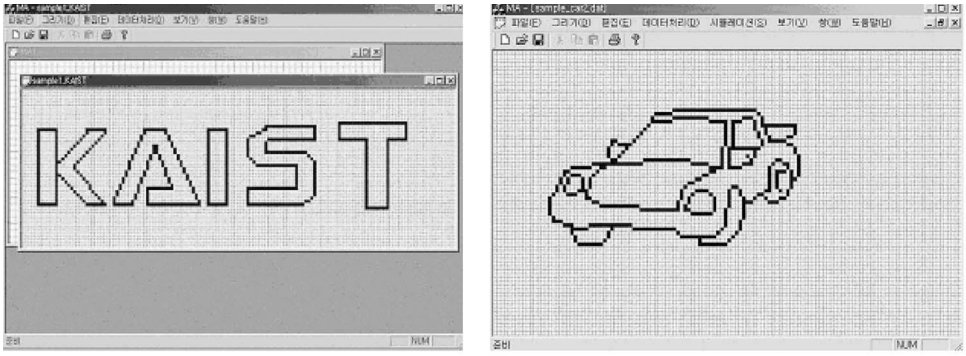
Table 2
Simulation results: CA stands for control architectures for MPS, DCA for the decentralized control architecture, and CCA for the centralized control architecture

| CA | DCA | | CCA | |
|---|---|---|---|---|
| Samples | 1 | 2 | 1 | 2 |
| Robot_1 (cm) | 256.5 | 296.0 | 224.0 | 278.0 |
| Robot_2 (cm) | 215.0 | 298.0 | 225.0 | 254.5 |
| Robot_3 (cm) | 255.0 | 311.0 | 227.0 | 261.0 |
| Total trajectory distance (cm) | 726.5 | 905.0 | 676.0 | 793.5 |
| Elapsed time (s) | 60.8 | 74.9 | 52.2 | 64.0 |

Above data are best cases.

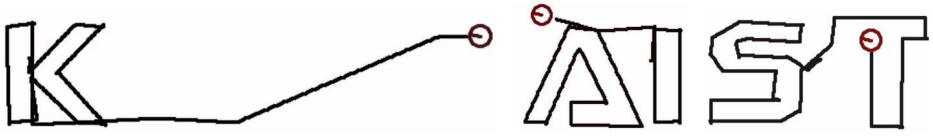
### 4.1. Decentralized control architecture

Figs. 13 and 14 are simulation results of the MPS with the decentralized control architecture (DCA). The figures show the trajectory of each robot on printing paper. As the initial position of each robot was properly selected for sample 1 and sample 2 with equal distribution of lines and clear boundaries, they seldom had ineffectual trajectories and drew lines near themselves sequentially well. Also, there were no duplications with respect to selection of a line as a result of intercommunication. Consequently, all the robots performed mobile printing well without pre-planning. Total trajectory distance and elapsed time of each robot were almost similar. In Fig. 13(a) we can see that Robot_1 moved to the upper part of 'T' at the cost of its trajectory in order to minimize the total elapsed time. This case is a defect of the behavior-based control architecture. If Robot_3, which was nearest to the last line drawn by Robot_1, drew it at the cost of slightly longer elapsed time, it would be very effective with regard to energy efficiency. This defect can be solved by applying some restrictions to the control architecture. For example, if energy cost is much higher than time cost in the case of dealing with some remaining tasks, the robot with responsibility for these tasks must hand over the responsibility to other robots. Accordingly, the decentralized control architecture is very flexible and fault-tolerant, because




(a) (b)

Fig. 12. Samples for printing; (a) sample 1: KAIST logo, (b) sample 2: automobile.



(a) (b) (c)

Fig. 13. Simulation of the decentralized control architecture for sample 1.

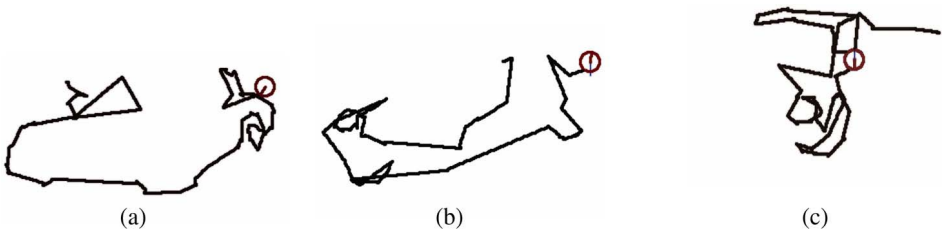



Fig. 14. Simulation of the decentralized control architecture for sample 2.

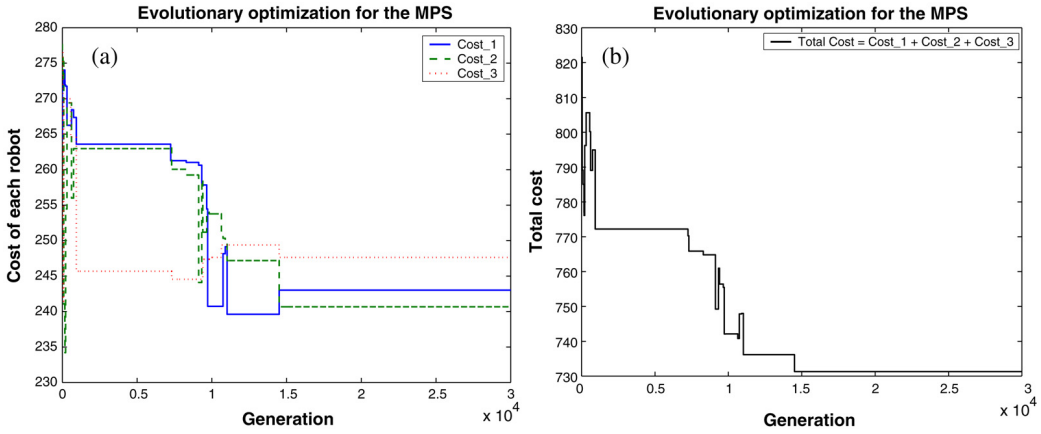



Fig. 15. Evolutionary task-allocation optimization for the MPS in the time-optimal manner (sample 1).

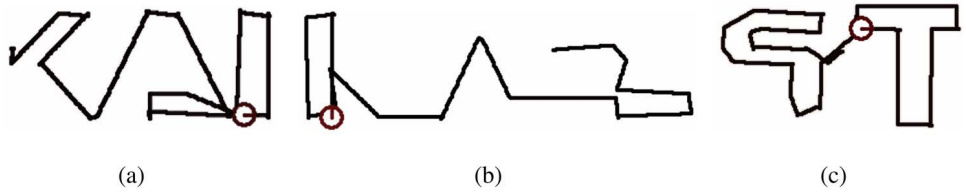



Fig. 16. Simulation of the centralized control architecture for sample 1.

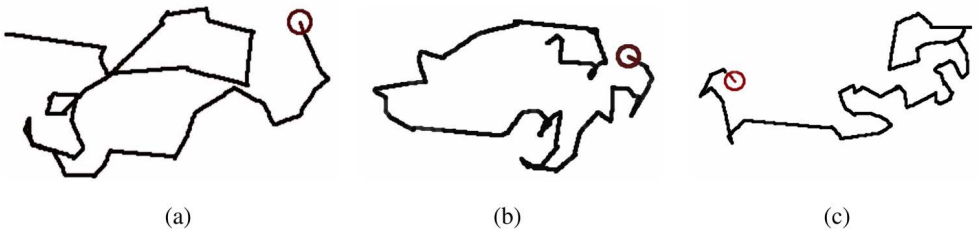


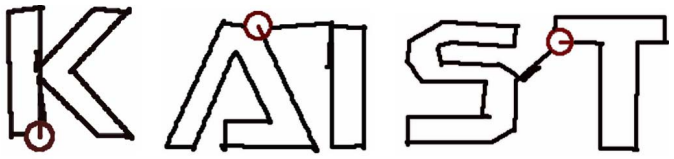

Fig. 17. Simulation of the centralized control architecture for sample 2.

no pre-planning is needed and the remaining robots can replace the failed robot.

### 4.2. Centralized control architecture

Fig. 15 shows the progress of each of *Cost*_1, *Cost*_2, *Cost*_3 and their sum (*Total Cost*) by the evolutionary task-allocation in the centralized control architecture (CCA) for the MPS. In the case of sample 1, *Total Cost* converged to a global optimum at approximately 15,000 generations. Then, *Cost*_1, *Cost*_2, and *Cost*_3 converged to 243.0, 241.0, and 247.0 cm uniformly. Since this algorithm selects the maximum among *Cost*_1, *Cost*_2, and *Cost*_3 as *Cost function* per generation, the values change complementarily, while their sum approaches the neighborhood of the global optimum with a fast convergence rate. As a result of these characteristics, the tasks of the three robots can be uniformly distributed. Also, in the case of sample 2, the control architecture showed such a global search ability that *Cost*_1, *Cost*_2 and *Cost*_3 converged to uniformly distributed optimal values.

As shown in Figs. 13 (output display in Fig. 19(a)) and 16 (output display in Fig. 19(b)), the trajectories of the robots in the decentralized control architecture were discriminated from each other as ‘K’, ‘AI’, and ‘ST’, while in the centralized control architecture Robot_1 has drawn some parts of ‘K’



(a) (b) (c)

Fig. 18. Trajectory optimization of the centralized control architecture for sample 1.

and 'A', Robot_2 has drawn its share of lines via 'S', 'I', 'A', and 'K' sequentially, and Robot_3 has drawn 'S', and 'T'. Figs. 14 (output display in Fig. 19(c)) and 17 (output display in Fig. 19(d)) also similarly display the aforementioned characteristics. These results show the global searching ability for a solution of the centralized control architecture with respect to time-optimality or cost-optimality factors. As shown in Table 2, simulation results of the MPS by the centralized control architecture were improved in terms of total trajectory distance and elapsed time relative to those by the decentralized control architecture.

Optionally, *Cost function* can be defined as the sum of trajectory distances of the three robots, instead of being defined as the maximum among *Cost*_1, *Cost*_2 and *Cost*_3. Then, the evolutionary task-allocation for the MPS becomes an optimization problem of the total trajectory distance, not an optimization problem of the elapsed time. A simulation for the optimal total trajectories was conducted for sample 1, as shown in Fig. 18. As a result, the best solution was found within 1000 generations and the tasks of three robots were non-uniformly distributed, compared to the optimization of the elapsed time. Deviations in the allotted task amounts of the robots were very large but the total trajectories were 640.0 cm; this optimized value is shorter than that of the other control architecture. Total elapsed time was 107.0 s, and depended on Robot_3, which had a large amount of assigned tasks.

In the centralized control architecture, the mobile printer system can have the option to choose optimization based on either the elapsed time or total trajectory distance by choosing a proper cost function. This approach also yields effective performance with respect to one optimal task-allocation system in a single multi-robot environment. Fig. 19 shows the results of mobile printing by three robots for each sample by the DCA and the CCA, which is displayed in a host computer. For example, Fig. 19(a) is obtained by assembling Fig. 13(a)–(c) together.

As the robots become smaller, it will be necessary to have many thousands of these robots to accomplish a non-trivial print task. An algorithm that creates look-up tables efficiently based on many lines and figures, a global task-allocation that uses EA or different competent algorithms, and a locally optimized path-planning within an allocated task must be essentially guaranteed. In addition, if there exists a failed robot, it must be able to transfer its own task to other robots. This explains the significance of a hybrid control architecture, which combines advantages of both centralized control architecture and decentralized control architecture.

## 5. Experiments

MPS experiments were carried out on the centralized control architecture, which shows better performance with respect to time-optimality than the decentralized control architecture. An optimization method based on the elapsed time is adopted.

The parameters of the robot, which is named Omni-Kity III in [18,19], are as follows. The size and the weight are 18.0 cm × 18.0 cm × 26.0 cm and 2.1 kg, respectively. The CPU is a Pentium-II 233 MHz, which is mounted on an on-board single PC. There are three DC motors, each with a resolution of 180 pulses/revolution. The communication protocol is a wireless TCP/IP using a Lucent Ethernet card (11 Mbps). The maximum velocity of each robot was limited to 5 cm $s^{-1}$ for accuracy in this experiment. As the robot was omnidirectional, the direction of the pen mounted on the upper body of the robot was fixed parallel to the absolute axis. The size of the real experimental workspace was about 250 cm × 50 cm.

The printing results, Figs. 20–22, were captured by one digital camera. Consequently, the deviation in allotted task of each robot was similar to the simulation results and the robots performed the mobile printing task without collision within a short span of time with their maximum velocity.

Fig. 20 shows the result from each robot for drawing sample 1. Fig. 16 for the simulation and Fig. 20 for the experiment are quite similar. Figs. 21 and 22 show the results using all three robots drawing the lines allocated to them simultaneously. Elapsed times of Robot_1, Robot_2, and Robot_3 were 250.0, 257.0, and 257.0 s, respectively. These experimental results demonstrate that the centralized control architecture is suitable for a time-optimal mobile printer system.

## 6. Conclusions

This paper introduced a new concept of a mobile printer system (MPS) and proposed multi-robot control architectures for its implementation along with an interface architecture. The proposed system was simulated and real experiments were conducted. The results demonstrated the effectiveness and applicability of the proposed control architectures and its interface. The decentralized control architecture based on behavior-based architecture showed fault tolerance and flexibility without pre-planning. In particular, the EA for the MPS in the centralized control architecture showed good performance in a time-optimal manner. It also offered flexibility in that the user can choose a trajectory-optimal manner. Furthermore, the individual representation and the fitness function can be used in many different situations involving task-allocation problems.

In the future, the printing tasks of extensive sites such as pavements, parks, and hazardous construction will be undoubtedly performed by the robot for efficiency and accuracy. Also, the bigger the site will be, the more robots they will need. Thus the problem of the task management for operation efficiency such as Operational Research (OR) will appear definitely and it will be concluded as a problem to

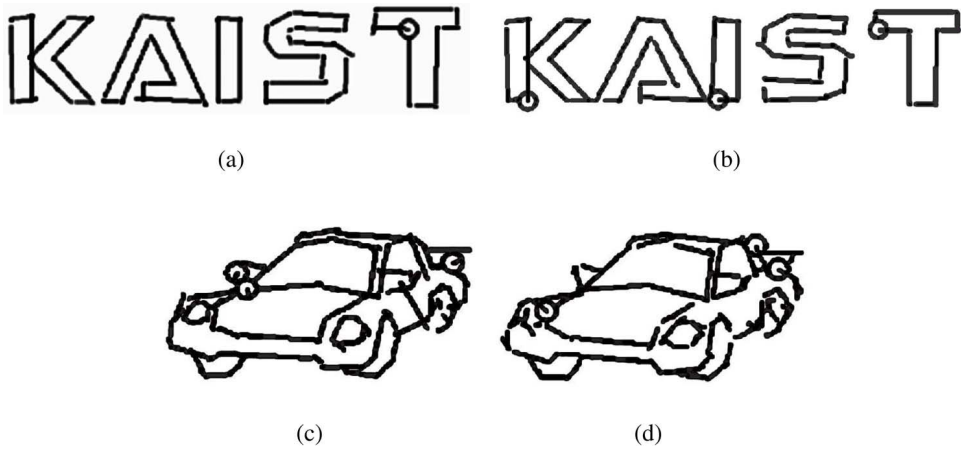



Fig. 19. Simulation results of mobile printing displayed in a host computer. (a) Sample 1 by DCA. (b) Sample 1 by CCA. (c) Sample 2 by DCA. (d) Sample 2 by CCA.

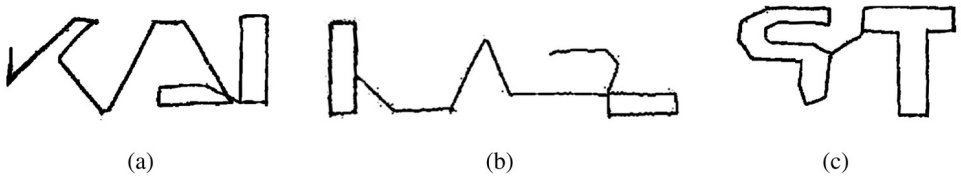



Fig. 20. Mobile printing results of each robot for sample 1 with the centralized control architecture.

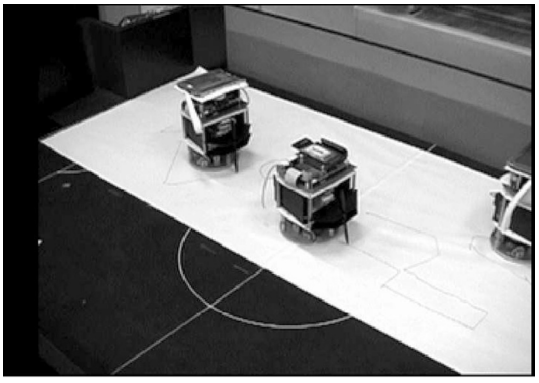


Fig. 21. Experiment with three mobile printing robots.

Fig. 22. Total mobile printing result of three robots for sample 1 with the centralized control architecture.

allocate tasks to multiple robots most effectively. The proposed MPS design will be applicable to these vast sites.

For further research, a global positioning system using sensors such as a laser ranger finder or a vision camera would have to be mounted on the MPS to improve the mobile print performance. Also, the mobile printing task would have to be carried out by printer-robots with a manipulator with high DOF and various research environments for the multi-robot system should be investigated. Furthermore, the GUI for the MPS would be improved if images of several other formats could be input in addition to lines so that the control architectures could be applied to more complicated samples.

## Acknowledgement

This work was supported by the ITRC-IRRC (Intelligent Robot Research Center) of the Korea Ministry of Information and Communication.

## References

[1] P. Hertling, L. Hog, R. Larsen, J.W. Perram, H.G. Pertersen, Task curve planning for painting robots. I. Process modeling and calibration, IEEE Transaction on Robotics and Automation 12 (1996) 324–330.
[2] S.-H. Suh, I.-K. Woo, S.-K. Noh, Development of an automatic trajectory planning system (ATPS) for spray painting robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, CA, 1991.
[3] A. Pichler, M. Vincze, H. Andersen, O. Madsen, K. Hausler, A method for automatic spray painting of unknown part, in: Proceedings of the IEEE International Conference on Robotics and Automation, Washington DC, 2002.
[4] M.R. Stein, Interactive internet artistry, IEEE Robotics and Automation Magazine 7 (2) (2000) 28–32.
[5] L.F. Penin, C. Balaguer, J.M. Pastor, F.J. Rodriguez, A. Barrientos, R. Aracil, Robotized spraying of prefabricated panels, IEEE Robotics and Automation Magazine 5 (3) (1998) 18–29.
[6] S. Kotani, H. Mori, S. Shigihara, Y. Matsumuro, Development of a lane mark drawing robot, in: Proceedings of the IEEE International Symposium on Industrial Electronics, Santiago, Chile, 1994.
[7] M.-J. Jung, H.-S. Shim, H.-S. Kim, J.-H. Kim, The Miniature omni-directional mobile robot OmniKity-I (OK-I), in: Proceedings of the IEEE International Conference on Robotics and Automation, Detroit, Michigan, 1999.

[8] E. Falkenauer, A new representation and operators for GAs applied to grouping problems, Evolutionary Computation 2 (2) (1994) 123–144.
[9] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, New York, 1999.
[10] G. Von Laszewski, Intelligent structural operators for the k-way graph partitioning problem, in: Proceedings of the International Conference on Genetic Algorithms, San Mateo, CA, 1991.
[11] D.R. Jones, M.A. Beltramo, Solving partitioning problems with genetic algorithms, in: Proceedings of the International Conference on Genetic Algorithms, San Mateo, CA, 1991.
[12] T.D. Barfoot, G.M.T. D'Eleuterio, An evolutionary approach to multiagent heap formation, in: Proceedings of the IEEE Congress on Evolutionary Computation, Washington DC, 1999.
[13] E. Pagello, C. Ferrari, A. D'Angelo, F. Montesello, Intelligent multirobot systems performing cooperative tasks, in: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Tokyo, Japan, 1999.
[14] J. Liu, J. Wu, X. Lai, Analytical and experimental results on multiagent cooperative behavior evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, Washington DC, 1999.
[15] W. Sheng, N. Xi, H. Chen, Y. Chen, M. Song, Surface partitioning in automated CAD-guided tool planning for additive manufacturing, in: Proceedings of the International Conference on Intelligent Robots and Systems, Las Vegas, 2003.
[16] R. Brooks, A robust layered control system for a mobile robot, IEEE Journal of Robotics and Automation RA-2 (1) (1986) 12–23.
[17] R.C. Arkin, Behavior-based Robotics, MIT Press, Cambridge, MA, 1999.
[18] M.-J. Jung, J.-H. Kim, Mobility augmentation of conventional wheeled bases for omnidirectional motion, IEEE Transactions on Robotics and Automation 18 (1) (2002) 81–87.
[19] M.-J. Jung, J.-H Kim, Development of fault-tolerant omnidirectional wheeled mobile robot using non-holonomic constraints, The International Journal of Robotics Research 21 (5) (2002) 527–539.

**Kang-Hee Lee** received his BSc and MSc degrees in electrical engineering and computer science from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Republic of Korea, in 1999 and 2001, respectively, and is currently working toward the Ph.D. degree. His present research activities focus on ubiquitous robot, genetic robot, evolutionary optimization and software engineering.

**Jong-Hwan Kim** received his BSc, MSc, and Ph.D. degrees in Electronics Engineering from Seoul National University, Korea, in 1981, 1983 and 1987, respectively. Since 1988, he has been with the Department of Electrical Engineering and Computer Science, the Korea Advanced Institute of Science and Technology (KAIST), where he is currently Professor and Chairperson for KAIST Robotics Program. He is the "Father of Robot Football", as described by The Times on September 18, 1997. He founded FIRA (Federation of International Robot-soccer Association; www.FIRA.net) in June 1997 and IROC (International Robot Olympiad Committee; www.IROC.org) in October 1998. He is President of FIRA, Chairman of IROC, and founding President of KRSA (Korea Robot Soccer Association). He is Director of MRDEC (Micro-Robot Design Education Center) supported by the Ministry of Commerce, Industry and Energy (MOCIE) of the Republic of Korea and of ITRC-IRRC (Intelligent Robot Research Center) supported by the Ministry of Information and Communication. His name is included in the Barons 500 Leaders for the New Century as a Founder of FIRA and Robot Olympiad. At present his research interests are in the areas of evolutionary multi-agent robotic systems and ubiquitous robots.