Advanced Robotics, Vol. 14, No. 5, pp. 347–349 (2000)

Autonomous acquisition and correction of navigation knowledge in a dynamic environment by intelligent data carriers

DAISUKE KURABAYASHI [1,*], HAJIME ASAMA [1] and KATSUMI KONISHI [2]

[1] *RIKEN: The Institute of Physical and Chemical Research, 2-1 Hirosawa, Wako-shi, Saitama 351-0198, Japan*
[2] *University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, Japan*

Keywords: Intelligent data carrier; autonomous knowledge acquisition; cooperation.

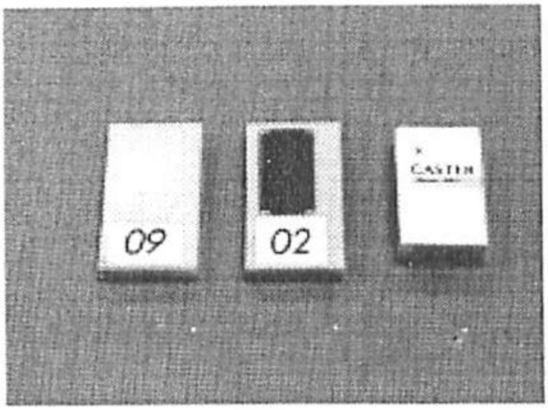
In this study, we develop a device for the cooperation of multiple autonomous mobile robots and propose an algorithm to navigate autonomous mobile robots without a global map.

The authors have developed a device named 'intelligent data carrier (IDC)' to reduce the traffic of global communication by providing local communication links and local information management functions. Robots can use the IDCs as a medium for inter-robot communication like pheromones of social insects. Furthermore, by putting the IDCs at arbitrary locations in an environment, a robot can allocate an agent for information storage and management.

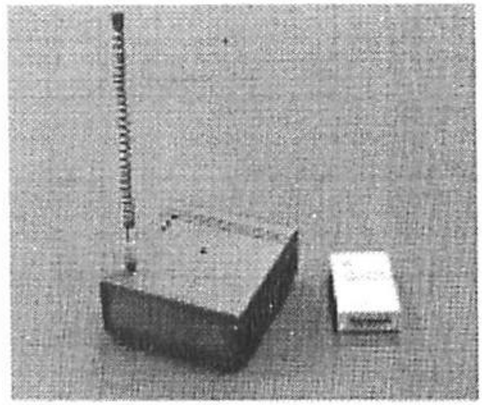
The IDC is an application of RF-ID (Radio Frequency IDentification). The IDC system consists of portable information storages (tags) and reader/writer devices carried by robots (Fig. 1). The reader/writer plays an active role in the system by initiating communication. Although a tag only replies to requests from reader/writers, it has its own CPU, memory and batteries. A user can download and execute an original program into a tag.

In this study, we consider an iterative transportation task (Fig. 2a). A robot has to carry objects to an ordered destination. We assume that a robot has no map, because a fixed map may spoil the flexibility of autonomous robotic systems. For the transportation task, a robot is told only the ID number of the destination. It has

*E-mail: dkura@riken.go.jp

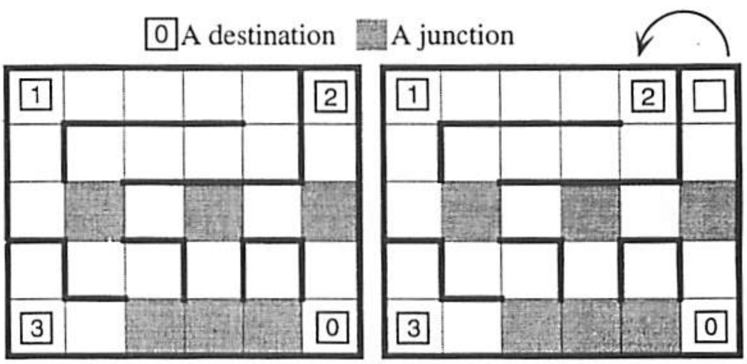


(a) Tags



(b) A reader/writer

Figure 1. IDC system.



(a) (b)

Figure 2. An environment for the transportation task.

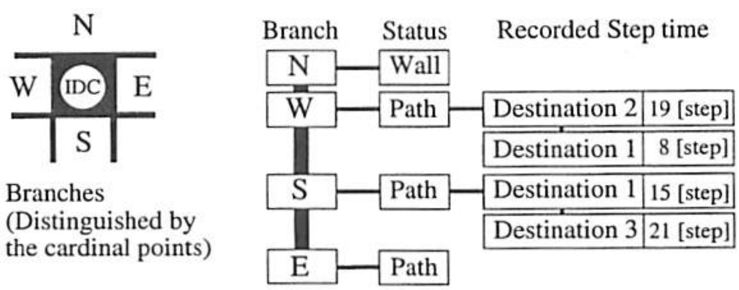


Figure 3. Data structure in an IDC.

to explore its destination. When a robot arrives at the current destination, it receives its next destination ID at random. We do not consider robot collisions.

A robot should select the 'correct' branch at a junction that takes the shortest path to the current destination for effective transportation. We set IDCs at junctions to aid the decision making of autonomous robots. Robots can store and share fragments of knowledge about the environment obtained by their experience. We

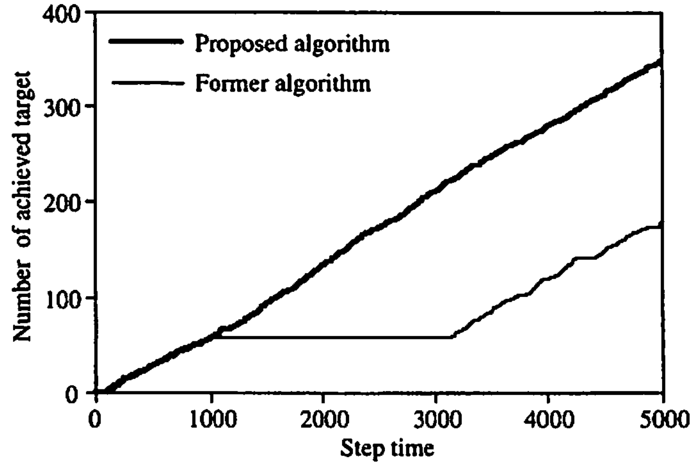


Figure 4. Comparison of achieved targets.

propose an algorithm to find the connection of branches to destinations according to information of the last visited destination and the entrance branch of a robot. For example, when a robot that started at destination 1 comes to a junction through the south branch, other robots can expect that the south branch may lead them to destination 1 (Fig. 3).

The algorithm performs very well in static environments. However, robots loose their way easily in dynamic environments. We assume that some destination points are moved during the execution of the transportation task (Fig. 2b). Inconsistent knowledge misleads the robots and decreases their performance. We propose an algorithm to find and avoid inconsistent data. An IDC judges the validity of its stored data according to the running time of robots that it communicates with. Let us assume that an IDC which has knowledge about destination i, j as d_i, d_j. A robot that wants to go destination j started from destination i and meets the IDC after t_r steps. We set (1) to calculate the probability of erasing knowledge about destination j. The equation compares stored data and the current steps of robots based on the logistic function.

$$p_{\mathrm{del}} = K\frac{e^{m(t_r-(d_i+d_j+d_j^2))}}{1+e^{m(t_r-(d_i+d_j+d_j^2))}}, \quad K=0.5, \quad m=\frac{\log 99}{d_j^2}. \tag{1}$$

Simulation results show the effectiveness of the proposed algorithm. We set a working environment as in Fig. 4. We change the location of destination 2 at 1000 steps in the simulations. A robot with the proposed algorithm achieved about 200% more destinations compared with a robot without it during 5000 step times.