

Cooperative Control of Mobile Robots Based on Petri Net

Hideharu KASHIMA, Ryosuke MASUDA

School of Information Technology and Electronics, Tokai University

1117 Kitakaname Hiratsuka-shi Kanagawa, JAPAN

E-mail: masuda@tokai.ac.jp

Abstract

A simulation and control method for multi-mobile robots based on an event condition system is described. The cooperative behavior of robots is expressed using an extended Petri net that uses events, conditions, and special interconnected arcs. Petri net representation is used for the lower-level control of single robot motion, and unified Petri net representation is used for the upper-level control of multi-robot cooperation. Both representation and control are carried out by a Petri net tool that uses a client/server system connected to an Ethernet.

The effectiveness of this technique is confirmed using fundamental experiments on cooperative motion with two intelligent mobile robots that have a multi-sensor system.

1. Introduction

Recently, there have been many studies carried out on the cooperation of multiple mobile robots, and various types of control methods have been proposed. However, there is still no standard representation for multi-robot cooperation, and thus researchers continue to use their own representation methods. Therefore, we believe it is worthwhile to develop a control method for a complex total system that can handle control problems related to single movement, cooperative motion, collision avoidance, etc. A unified system representation and a control method for multi-robot tasks in which "task sharing" and "cooperation" are included are necessary. In the past few years, attempts have been made to design an automation system by using Petri nets, which was developed for the modeling of a parallel information processing system [1] to be used as an interface between humans and robot systems. Hasegawa et al. proposed a practical representation method for

automation and robotics [2]. Koyama et al. proposed a simple but higher-level control language called "cell language"[3]. C. Renji et al. used Petri net for the cooperative assembly by multi-agent system [4]. Iwai et al. were working to develop effective motion planning by using several features of graph representation [5]. Most of these applications were developed for planning of simple task. In this paper, we propose a practical planning and control method for a multi-robot cooperation system using Petri net [6].

2. Representation of Robot Motion by Petri Net

A multi-robot system is considered to be an event-condition system. All of the motions and tasks are decided according to events and conditions. An event is an action that is generated in a system, and the activation of an event is managed by the system conditions. To generate a new event, it is necessary to establish a specific condition. The unique features of multi-robot cooperation are asynchronous, sequential, and parallel characteristics. Petri net is capable of representing all of these characteristics effectively. An event can be activated if and only if all of the preconditions are satisfied and all of the post conditions are satisfied. That is, the asynchronous and sequential conditions. In Petri net, multiple conditions may be activated simultaneously, and each event can be individually generated to transfer multiple tokens through each event. Furthermore, Petri net has the ability to represent the concurrency with which the change in conditions related to multiple events is decided

according to the timing of the occurrence of these events. Using these special features, Petri net can be used as a multi-task Turing machine. As for the robot system, place (condition) represents the state of motion, and transition (event) represents the change in robot motion. Therefore, the holding of robot motion is shown by the token on the place, and the task sequence of a single robot is graphically represented by the flow of tokens through the events. By combining this single robot net, cooperative motion of multiple robots can be made possible.

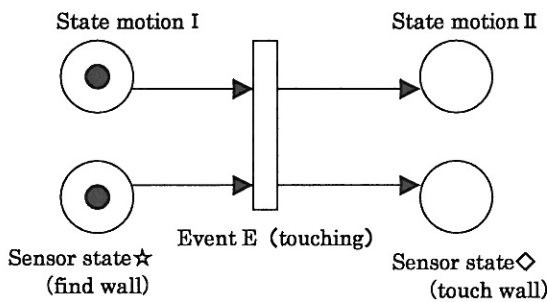


Fig. 1 Petri net representation

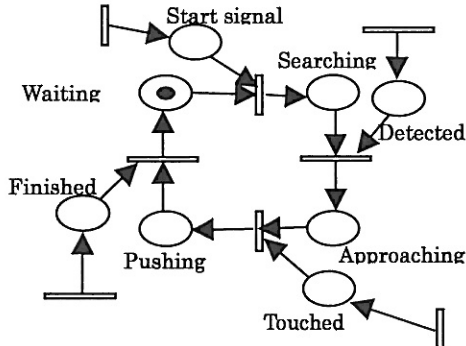


Fig. 2 Petri net for pushing box task.

Fig.1 shows a fundamental Petri net. In this figure, a condition (place) is represented by a circle, an event (transition) is represented by a bar, and holding of the condition is represented by a black mark called a token. Each element is connected by directed arcs. If the input side conditions (I and ☆) are satisfied, an event E is activated, and as a result, new output conditions (II and ◇) are established. This activation of an event is called “fire” or “firing”, and the tokens are transferred to output side conditions. The condition ☆ can be thought of as a control signal from a sensor system,

and the state motion II can be thought of as an output to the control system.

As an example of a robot task, think about one robot being given the task of pushing a box. In this task, if the robot finds the box while searching, it starts to carry the box to the goal position. This task can be represented by the Petri net, seen in Fig.2. In this Petri net, total motion is decomposed into four simple motions and aligned in a fixed order. When a robot is under the condition of “searching” and given the condition of “find a box,” the event of motion is activated to move the token and then transfer to the condition “following a box.” After touching the box, the robot pushes it to the goal point, and then the task is either in the “end of pushing” or “standing by” state.

2. Cooperative Control using a Multi-Sensor System

Here, we describe the task of pushing a box with two robots. For the representation of the cooperative task motions, the inhibiting arc and the enabling arc are added to the fundamental Petri net. By using this extended Petri net, the interaction with two robots can be clearly represented. Fig.3 shows the Petri net representation of the cooperative robot motions. This net shows an exclusive robot motion in which the motion of one robot is inhibited, while the other robot pushes the box. The box is pushed by the two robots, one robot at a time.

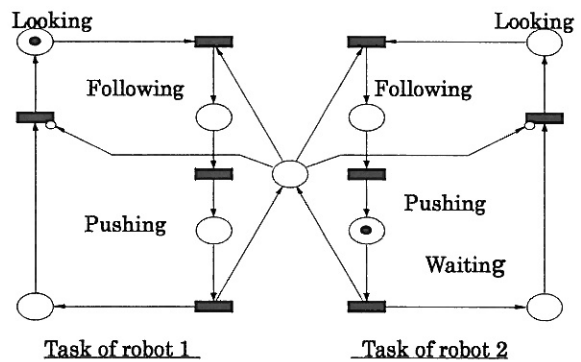


Fig.3 Pushing a box using two robots by exclusive control.

The cooperative task is represented by the exchange of the information between individual tasks. When the cooperation task is finished, the tokens are transmitted to

each part, and the individual task will start again. Using this method, the task can be executed on the graph as a simulation. The firing of the event is determined by investigating the sensory signal of each robot. At the time of token transmission, the message from the robot controller is transmitted to the Petri net.

4. Control System based on Petri Net

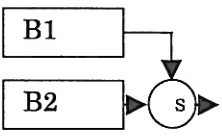
We have developed a method to represent the sensory control motions of a single robot [2]. But here, a new control method is utilized, because when the usual Petri net is used to represent a multiple robot cooperation system at every control layer, the net may be too complex to be managed by a designer. So in this approach, the system is decomposed into a robot part and a cooperation control part. The advantage of this method is the fact that the robot controller side and the Petri net side are perfectly independent of the cooperative control even though the robot controller is also written using the Petri net. Therefore, the exchange of information between the net and the robots is carried out using a “message.” Petri net describes the sequence of the task and collects information regarding the system condition from which the token departs and the other condition to which the token reaches. The Petri net also sends the command to the robot controller as a “message” according to the transformation of the tokens. The controller uses this information to change its action, and then sends back the occurrence of events judged from sensing of the situation. After receiving the command from the Petri net, the robots can be controlled using a simple rule based on reflective motion. By repeating the exchange of such information, the Petri net can manage the complete cooperative task motion of the robots.

Petri net can be used for the description of task motions as well as for communication tools between robots. One unique feature of a Petri net is that it has the flexibility to apply to any type of robot. Once a certain transition such as pushing a box is allotted as a robot motion, only changing the lower level control rule of an individual robot can make it possible to begin use of a new type of robot. This method can be applied to any type of multi-robot or multi-sensory control system.

For the mobile robot (Khepera) used in this study, we

use a subsumption control architecture. Based on the rules shown in Table 1, motions B1 and B2 were defined. The reflective motion shown in Table 3 was activated according to the sensing information given in Table 2. Here, the avoidance action while moving forward to the goal position (B1 = AVOID, B2 = FORWARD) is given as the fundamental motion. This robot has eight proximity sensors, two touch sensors, and two light sensors. Suppose that the robot recognizes the number of proximity sensors that output the maximum signal, $I = \max(\text{proximity_data})$. This sensor is considered to be located at the position closest to the wall. The robot can then choose the motion listed in Table 4 (this case, H_AVOID) and send the drive signal to the wheel drivers.

Table 1 Examples of motion for each transition.



Motion	Expression
Follow the light with avoiding	B1=H_AVOID B2=F_LIGHT
Follow the light with pushing	B1=F_LIGHT B2=PUSH
Move forward	B1=FORWARD

Table 2 Sensor and detected information

Sensor	Detected information
Proximity sensor (Surrounding 8)	proximity_data="0,0,394,540, ... 0 (far) ~1023 (close) distance within 2cm for white paper
Photo sensor (center 1)	light_data = "500,500,230,240,..." 0 (intense light) ~500 (no light)
Touch sensor (front bothside 2)	push = 0 (non touch) ,1(touch right), 2 (touch left), 3 (touch both)

Table 3 Sensor processing and related motion

Name	Sensor processing	Motion
H_AVOID	$I = \max(\text{proximity_data})$	Act sensitive for avoiding
PUSH	if push=1 then move=right if push=2 then move=left	Pushing a box
F_LIGHT	$J = \min(\text{light_data})$	Following
LEFT	Non	Going right
RIGHT	Non	Going left
FORWARD	Non	Going forward

At the initial state, if the motions are defined as “B1 = AVOID” and “B2 = FORWARD,” then the robot proceeds forward and is able to avoid obstacles that it detects. In the Petri net, either B1 or B2 is selected depending on the transition of the token.

The motion of the robot can be determined by allotting the subsumption relation of B1 and B2. Before finding the next condition, the robot moves according to the command given previously.

Table 4 Example of "H_AVOID"

Number of maximum detecting sensor I	Command to robot	Motion
---		No change
0	"D,4,-2"	Turn right slightly
1	"D,8,-5"	Move forward right
2	"D,5,-5"	Turn right large
3	"D,-5,5"	Turn left large
4	"D,-5,8"	Move forward left
5	"D,-2,4"	Turn left slightly
6	"D,8,8"	Move forward fast
7	"D,8,8"	Move forward fast

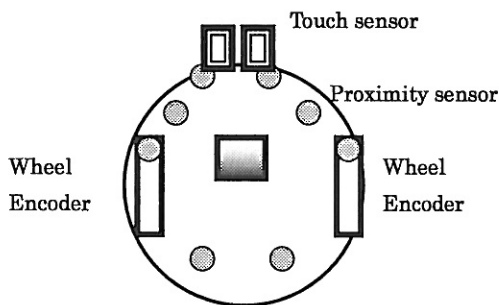


Fig.5 Sensor allocation of the robot.

In the robot control system, the occurrence of a condition represented by the generation of tokens, is dependent on the system's sensing information. Before beginning a task, the data base is created using the sensing data from Table 1. The controller that actually controls the robot must not only control the robot but also determine the timing with which to fire the events in order to generate the token for the next state.

5. Simulation System

A cooperation task can be simulated when the Petri net representation of the total control system has been completed. The reason behind the occurrence of an inappropriate behavior or the generation of a deadlock state can also be checked through a simulation.

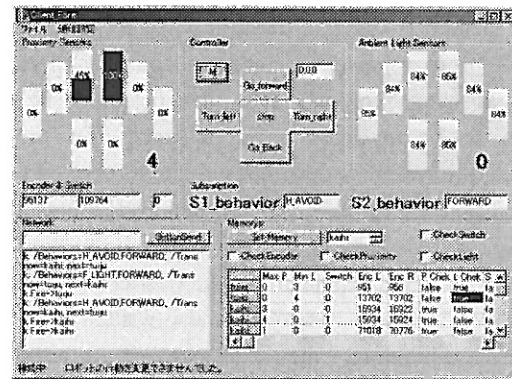
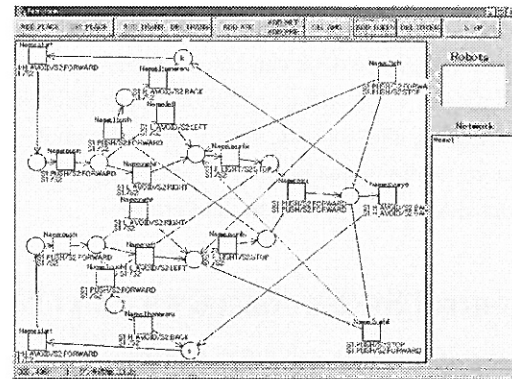


Fig.6 Petri net tool and robot controller.

Fig.6 shows the Petri net tool we used in our research [4]. This system carries out the total task simulation by manually transforming the tokens one by one. The upper part of the image in Fig.6 shows the Petri net server that is connected to the TCP/IP by more than one robot, and the lower part of Fig.6 is the display image of a robot client. Each robot should have sensor conditions, a subsumption structure, and transmission control of messages from the robot to the Petri net. Actually, the debugging process in which deadlock analysis and re-design are included, and the implementation process are carried out by utilizing the system with Petri net in the same manner.

The process for control design of a robot system from the method described in this paper is as follows.

- (1) expression of motion
- (2) selection of sensor information
- (3) simulation (deadlock analysis and re-design)
- (4) allocation of control signal
- (5) execution of a task

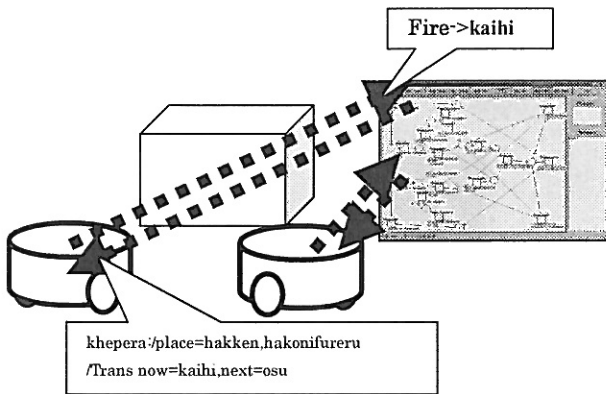


Fig.7 Information transmitted between the robots and the Petri net controller.

Table 5 Message contents.

place/now= a,next= b	Actual motion ;a , Next condition to do ;b
trans/now= c,next= d	Actual motion ; c , Next condition to do ;d
Fire->T_NAME	Next firing transition (action) ;T_NAME

The designed Petri net is connected to a client/server system by an Ethernet. In other words, each robot can get information regarding the task by connecting a robot client (controller) to a server (Petri net). When the robot determines that the tokens for the condition can be transformed using the sensor information, the event named “T_NAME” is forced to fire by inputting the command “Fire->kaihi” to the Petri net server. As a result, the conditions that must be done now and checked next, are given by the message. “Name/place now=hakken next=fureru /Trans now=kaihi next=osu” The total task is completed using this continuance. The robot client must have the knowledge base necessary to recognize changes in conditions from the sensor information in advance.

Execution of the cooperation task can be shown to be possible if an experiment on the unity robot task is successfully conducted using the Petri net because the system is not dependent on the number of the robots. Even if the number of robots increases, there is no need to make drastic changes in the control system.

6.Experiments on a Cooperation Task

We conducted an experiment on control for the task of pushing a box by using a Petri net with a single robot.

An experiment on the cooperation between two robots was also conducted. The robots used were both intelligent mobile robots (Khepera) that have eight proximity sensors, two photo sensors, two touch sensors and, two encoders. The robot was surrounded by a white wall measuring 30cm x 50cm, and on the working plane there were two 5cm x 5cm white cubic obstacles, and a cylindrical object (in the task description we call it a “box”) 6 cm in diameter with a light source.

The Petri net expression for a single robot is shown in Fig.9. The robot was made to act based on this Petri net expression, and by using the token flow in Petri net, it was possible to observe the intended behavior of the robot.

This net was designed for the task of finding a box, avoiding obstacles while following the box, pushing the box to the goal, leaving from the box, and stopping .

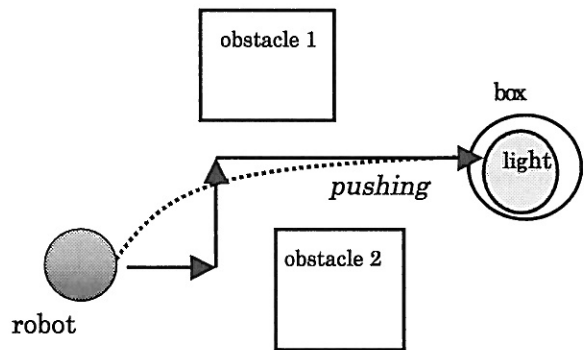


Fig.8 Task of pushing a box by single robot

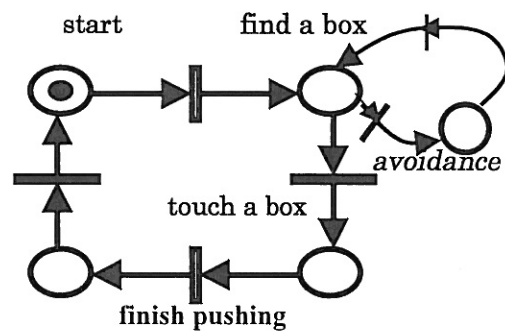


Fig.9 Petri net of a single robot task

The robot began to move forward, beginning with the condition of “looking for a box.” After the light was found a “following a box” action was started according to the flow of the token in the net, and the

obstacle was avoided while moving. The task of “pushing a box” was conducted until it changed to the “it is pushed” action and the value of the encoder reached the setup value.

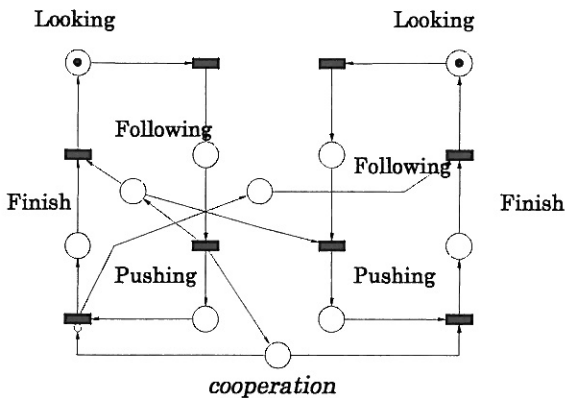


Fig.10 Petri net of a cooperation task

Next, the same task was done using two robots. Fig.10 shows the Petri net representation of the cooperation task. The robot began from the state of “pushing a box” after moving a given distance judged by the encoder, the token was transmitted to the next state, and then the action of “leaving from the box” was observed. In our experiments we were able to effectively express the cooperation behavior, and this expression enables us to control the cooperation behavior.

The results of the experiments and consideration for them are as follows. For each fundamental motion and cooperation task, we made experiments ten or more times. Most of the trials the tasks were completely executed, but a few percent of the trials could not reach to the goal. The task failures that the transition did not fire were caused by,

- (1) miss-recognition of condition by proximity sensor.
- (2) Measurement error of encoder counter.
- (3) Communication delay to the server.

These points are the next problems to be solved.

7. Conclusion

A technique for expressing the cooperation behavior of a multi-mobile robot with a multi-sensor system by using Petri net was described. The work to describe a program to compose the behavior of the robot was substituted for the work to construct a net intuitively by

this method. In addition, we found that it was effective to consider the lower-level control of a single robot and the higher-level control of cooperative action separately.

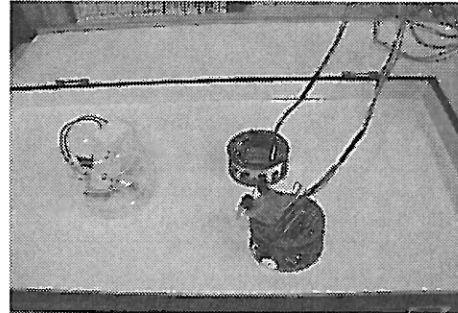


Fig.11 Experiment of cooperated task

This finding is applicable to behavior planning with obstacle avoidance for mobile robots and cooperation task planning for multi-robots in a general environment. We have shown that it is possible to describe control with a behavior-based approach of Petri net. The treatment of larger numbers of events and the use of sensor fusion processing to recognize a condition are problems left for the future work. Therefore, a sensor that has larger amounts of information (for example, a vision sensor or a range sensor) will be necessary to make cooperative control by Petri net more effective.

References

- [1] J. L. Peterson: Petri net Theory and the Modeling of Systems, Prentice Hall (1981)
- [2] Koyama, Miyake: Development of FA Cell Control Language by using Petri Net, Journal of the Robotics Society of Japan(RSJ), Vol.17, No.5, pp649-657 (1999)
- [3] C. Renji et al. : Cooperative Assembly Algorithm of Multiple Distributed Agent Based Robotic System, Proc. on 30th ISR, pp.545-552 (1999)
- [4] K. Ishii, N.Hiyama: Motion Planning of a Mobile Robot with Petri Nets, proc.of 17thRSJ, pp605-606 (1999)
- [5] Hasegawa, Masuda et al.: Proposal of Mark Flow Graph for Discrete System Control, Proc. of SICE Vol.20, No.9, pp.122-129, (1984)
- [6] Kashima, Masuda: Cooperation of Multi-mobile Robot by Event Driven Control Theory, Proc. of Robotics Symposia Japan, Vol.2, pp713-718, (1999).
- [7] Petri net simulator:
<http://home.arcor-online.de/wolf.garbe/petrisoft.html>