Robotica (2001) volume 19, pp. 371–380. Printed in the United Kingdom © 2001 Cambridge University Press

A model-based error recovery scheme for a multi-robot system

Chan Woo Moon and Beom Hee Lee

School of Electrical Engineering, Seoul National University, San 56-1, Shinrim-dong, Kwanak-ku, Seoul 151–742 (Korea). E-mail: mcw@robot1.snu.ac.kr

(Received in Final Form: August 10, 2000)

SUMMARY

A work cell with multiple robots increases manufacturing flexibility and productivity. Robots in the work cell equipped in a sparse area usually share a motion path, resources and workspace. In this paper, an embedded Markov chain model for a multi-robot system that has a common workspace is constructed on the basis of the concept of a multi-processor system. With the presented model, we measure the performance of error recovery schemes under different workloads and analyze the sensitivity of the execution time with respect to the robot speed. We verify the presented model with an experimental multi-robot work cell. This study is useful in evaluating the performance of a robotic work cell and presents a guide for designing a complex work cell.

KEYWORDS: Multi-robot system; Markov chain model; Error recovery.

1. INTRODUCTION

A robotic work cell with multiple robots can accomplish complex tasks that a single robot cannot manage. Recently, there have been increasing research and demand for a multi-robot system because of this benefit. Research issues for multi-robot systems include collision avoidance, task planning, communication and performance evaluation.[1–4]

Currently, robots perform simple and repetitive tasks such as assembly, welding and material handling, but the appearance of new computer technology, sensors and vision systems has enabled the robots to perform more complex tasks with flexibility and intelligence. The more flexibility and intelligence a robotic system has, the more difficult is the prediction of the behavior of the system. Therefore, modeling and performance evaluation is essential for designing a robotic work cell.

Petri net, the Markovian model, the queueing network and the finite state machine are widely used to model a robotic work cell, and many researches have been reported. A brief review of recent studies is as follows. Petri net is regarded as established methodology for the modeling of flexible manufacturing systems (FMS).[5] Based on Petri net, an algorithm of automatic robot assembly planning is presented.[6] Robinson[7] modeled and measured the performance of the robotic facility that consists of computers, robots, sensors and a vision system. In a stochastic Petri net, the firing time is an exponentially distributed random variable. Using a stochastic Petri net, Zhou[8] modeled a resource sharing manufacturing system, and implemented a deadlock free control system. Lima[9] proposed the interpretation of two distinct Petri net types, a generalized stochastic Petri net for task quantitative performance evaluation and an ordinary Petri net for task qualitative performance evaluation. The quantitative performance evaluation concerns time-related properties and qualitative performance evaluation concerns properties such as boundedness, properness and liveness. Lima applied his model to visual servoing and track following. The time-related properties such as execution time can be measured with the model that is based on a timed Petri net.[10] Sensor data and robot languages can be represented by a Petri net. Lee[11] integrated local sensors of multiple mobile robots with the AND/OR logic of a Petri net. Suh[12] proposed a Petri net-type graphical robot language to represent parallelism and synchronization of a multi-robot system. In a colored Petri net, each individual token associates information. Zuberek[13] used a colored Petri net to model the FMS, where the token indicates parts and the color indicates scheduling policies.

The Markovian model is a versatile tool for the precise performance analysis of small systems, and it is known that stochastic Petri nets are semi-Markov or Markov processes. Gopalakrishna[14] analyzed the sensitivity of throughput of failure-prone FMSs with the Markovian model.

The queueing network is applicable for the analysis of large scale FMSs. Aras[15] proposed a hierarchical multi-processor computer architecture to process robot sensory information, and the performance was analyzed using queueing networks.

The supervisory control of a system that is modeled with a finite state machine is a new research area. Brandin[16] modeled a robotic work cell that consists of conveyor belts, robots, programmable logic controllers (PLC) and vision systems, and he constructed a real-time supervisory controller. Park[17] constructed a fault tolerant supervisory controller for the work cell that consists of arc welding robots and conveyor belts.

As reviewed above, studies of measuring performance of a manufacturing system with robots are abundant, but they consider the robot as a part of the manufacturing system. Compared to other manufacturing systems, a multi-robot system has a few distinct features. First, the performance of the system depends on configuration; second, each robot is assigned a different workload for each task; and finally, we can adjust the speed of the robot simply by changing the program, so the analysis of sensitivity with respect to robot speed is important. Lee[18] used the concept of a multicomputer processor to model a multi-robot system and evaluated its performance. He presented three interconnection methods for a multi-robot system and modeled a robot failure recovery for each method, but he did not consider collision and interference between robots. Robots in the work cell equipped in a sparse area usually share motion paths, resources and workspace, and they affect each other. In particular, error occurrence in one robot brings the retardation of other robots through the common workspace. This fact has hardly been considered in previous studies.

In this paper, we develop an embedded Markovian model to describe the multi-robot system that has a common workspace, and mathematically analyze the effect that one robot has on other robots. The effect of the interference depends on the operation scheme. With the presented model, we measure the performance of the error recovery scheme of a robotic work cell. Here, we evaluate the performance of a robotic work cell with two measures, viz. completion time and reliability. This study is applicable to the performance evaluation of a robotic work cell and presents a guide for designing a complex one.

The organization is as follows: Section 2 presents a modeling of a multi-robot system. In Section 3, several fault recovery schemes of a multi-robot system in a common workspace and numerical examples are presented. Then, Section 4 presents experimental results to verify the presented model. Failure recovery schemes are discussed in Section 5 as an extended study of Lee's[18] work. Finally, conclusions and future aims are presented in Section 6.

2. MODELING OF MULTI-ROBOT SYSTEM

The use of multi-robots in a common workspace increases robot utility and productivity, but we have to consider collision between robots. To avoid collision between them, we detect collision and calculate a collision-free trajectory for each robot.[1] However, this approach is restricted because it is hard to calculate the path of each robot. In practice, the mutual exclusion method is widely used. Here, on the basis of the concept of the multi-processor system, we construct an embedded Markov chain model for a multi-robot system that has a common workspace. The Markovian model is an appropriate method for precise performance analysis, but the state space becomes intractably large as the system size increases. Typically, the number of robots that share a common workspace is 2 or 3, because of spatial constraints, even though the whole work cell is large.

2.1 Multi-robot system configuration

Tsai[19] classified multi-robot motions into simultaneous motion, coordinated motion and overlap motion. The simultaneous motion and the coordinated motion are applicable to complex tasks that a single robot cannot accomplish. Moving one part by two robots is an example. On the other hand, the overlap motion doesn't need to operate robots tightly together. The overlap motion increases productivity because robots accomplish various tasks without changing programs or tools. Here, we concentrate on the overlap motion for our multi-robot system model. This system has the following features:

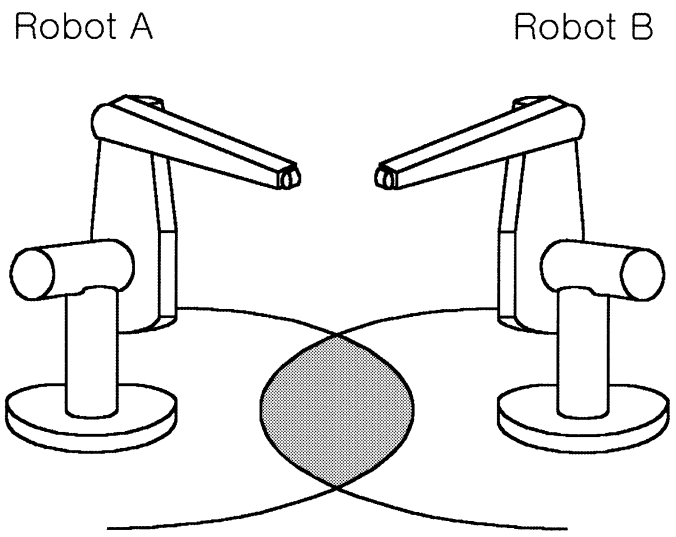

Fig. 1. Multi-robot system.

(i) Robots have a common workspace.
(ii) Each robot operates independently.
(iii) At each instant of time, only one robot operates in the common workspace to avoid collision.
(iv) Robots are assigned to different task loads respectively.

To model the above system, we divide the workspace into two areas, the exclusive workspace and the common workspace. A work cell that consists of two robots is shown in Figure 1, where the shaded area indicates the common workspace.

The robotic work cell consists of robots and an external controller. The controller coordinates the operation of robots. The PLC and the workstation are widely used as controllers. Notations are as follows:

Notation 1:

R:	The number of robots which share the common workspace
job:	One operation in the exclusive workspace + a successive operation in the common workspace
Cv:	Job vector, Vector of number of jobs which are allocated to each robot

Definition 1:

- Average Execution Time, u_k: Average time for robot k to finish one job.
- Average Execution Rate With Interference, μ^*_k: Average rate at which robot k finishes a job when robot k is interfered with by other robots that share common workspace.
- Average Execution Rate Without Interference, μ_k: Average rate at which robot k finishes a job if robot k is not interfered with by other robots.
- Average Completion Time, T: Average time taken to finish all of the allocated jobs
- Average Completion Time Without Interference, Ts: Average time taken to finish all the allocated jobs if robots do not have a common workspace.

2.2 Modeling of a multi-robot system that has the common workspace

In the work cell, each robot requests the controller's permission before it enters the common workspace in order to avoid collision. The controller admits the robot if the common workspace is empty. If the robot's operations are not synchronized, the robot experiences queue time before it enters the common workspace.[20] The average execution time u_k, $k \in (1, R)$ is changed when one of the robots in the work cell finishes an assigned task. Let τ_i denote the time duration from the (i-1)th instant at which one of the robots in the system finishes, to the ith instant at which the next robot finishes, then

$$T = \sum_i \tau_i \tag{2.1}$$

The average completion time T is estimated from the following steps:

Algorithm 1:

```
Step 1:
  $i = 0$
  $X_0 = \{x_1, \ldots, x_R\} = Cv$, the number of jobs to be executed
Step 2:
  If the number of remaining jobs is not zero
    $i = i + 1$
    Calculate $u_k$, $k \in (1, R)$
    $\tau_i = \min_{k \in (1,R),\, x_k \cdot u_k > 0} (x_k \cdot u_k)$
    $X_i = \{(x_1 - \tau_i/u_1)^+, \ldots, (x_R - \tau_i/u_R)^+\}$, the number of remaining jobs
    Repeat step 2

  Else
    $T = \sum_k \tau_k$

  End
```

For the case of a work cell with two robots A and B, and Job vector $\{C_A, C_B\}$, T is obtained as:

$$\begin{aligned} T &= \frac{C_B}{\mu*_B} + \left(C_A - \frac{C_B}{\mu*_B}\mu*_A\right)\frac{1}{\mu_A} \quad \text{if } \frac{C_A}{\mu*_A} > \frac{C_B}{\mu*_B} \\ &= \frac{C_A}{\mu*_A} + \left(C_B - \frac{C_A}{\mu*_A}\mu*_B\right)\frac{1}{\mu_B} \quad \text{otherwise} \end{aligned} \tag{2.2}$$

2.3 Calculation of average execution time

If the number of jobs is sufficiently large and robots that share the common workspace are not many, we can approximate u_k with the steady state average value, or throughput. To calculate u_k, $k \in (1, R)$, the following assumptions are made:

Assumption 1:

(i) Each robot enters the common workspace with exponential distribution, parameter λ_k for robot k.

(ii) The distribution of time that each robot spends in the common workspace is known.

PDF $G_k(x)$, pdf $g_k(x)$, average $\bar{g}_k$ for robot k

where g_k is the operation time of robot k in the common workspace plus processing time of the external controller.

(iii) Other robots are not admitted until the robot that is in the common workspace completes the task.

Assumption (i) is necessary in order not to have an infinite number of states when we build a Markov chain model. It is a reasonable assumption if the robot enters the common area uniformly through the work time.

The average execution time u_k is obtained as follows.

- With interference between robots:

$$u_k = \frac{1}{\mu*_k} = \frac{1}{\lambda_k} + w_k \tag{2.3}$$

- Without interference:

$$u_k = \frac{1}{\mu_k} = \frac{1}{\lambda_k} + \bar{g}_k \tag{2.4}$$

Here, w_k is the operation time of robot k in the common workspace plus the waiting time, and w_k, λ_k are functions of the schemes of controlling robots.

To calculate u_k, we drive an embedded Markov chain model for the system.[21,22] We select the time when any robot departs from the common workspace or arrives at an empty common workspace. System states are given by the state of the common workspace at that time. The feasible states of a work cell that consists of robot A and B has, are shown in Table I. S(k) indicates that robot k, $k \in (A,B)$ finishes the operation successfully and departs from the common workspace, and E(k) indicates that robot k arrives at an empty common workspace. Usually, an external controller determines the states. Table II shows the states of the work cell with robots A,B and C, where the controller checks the request in the order of A,B,C.

These states are embedded Markov chains and satisfy the following stationary equation.

$$\Pi \cdot P = \Pi \tag{2.5}$$

Table I. System states for the work cell with robots A and B.

State	Action	State of common workspace
$\pi 1$:	S(A)	0
$\pi 2$:	S(B)	0
$\pi 3$:	S(A)	B
$\pi 4$:	S(B)	A
$\pi 5$:	E(A)	A
$\pi 6$:	E(B)	B

Table II. System states for the work cell with robots A, B and C.

π1 : S(A) 0	π2 : S(B) 0	π3 : S(C) 0	π4 : S(A) B	π5 : S(A) C
π6 : S(B) A	π7 : S(B) C	π8 : S(C) A	π9 : S(C) B	π10 : S(A) BC
π11 : S(B) AC	π12 : S(C) AB	π13 : E(A) A	π14 : E(B) B	π15 : E(C) C

where $\Pi = \{\pi_i\}$ is the limiting probability of the Markov chain, satisfying $\sum \pi_i = 1$, and P is the state transition matrix. We define the index sets.

- I_k : Index set of all states, which indicate that robot k is in the exclusive workspace.
- J_k : Index set of all states, which indicate that robot k is in the common workspace.
- L_k : Index set of all states, which indicate that robot k moves out from the common workspace at the instant.

For example, from Table I, $I_A = \{\pi_1, \pi_2, \pi_3, \pi_6\}$, $J_A = \{\pi_4, \pi_5\}$, $L_A = \{\pi_1, \pi_3\}$

The average operation time that robot k spends in the common workspace is obtained as follows.[23]

$$w_k = \sum_{i \in I_k} \sum_{j \in J_k} \frac{\pi_i \cdot p_{ij}}{\sum_{i,j} \pi_i \cdot p_{ij}} E[D_k(i,j)] \qquad (2.6)$$

$E[D_k(i,j)]$ is the average operation time of robot k in the common workspace plus the average waiting time. Let m(i,j) denote average time to go from state i to state j, and S_k be the time to arrival at the common workspace of robot k during the time interval m(i,j).

Then,

$$E[D_k(i,j)] = E[R_k(i,j)] + E[F(j, L_k)] \qquad (2.7)$$

$$E[F(j, L_k)] = \sum_{l \notin L_k} p_{jl} E[F(l, L_k)] + \sum_{l} p_{jl} E[m(j,l)] \qquad (2.8)$$

$$E[R_k(i,j)] = \int_0^\infty \frac{1 - P[m(i,j) - S_k \le t]}{1 - P[m(i,j) - S_k \le 0]} dt \qquad (2.9)$$

3. FAULT RECOVERY SCHEMES OF THE MULTI-ROBOT SYSTEM

Robot errors are classified into fault and failure.[17]

Definition 2:

- Fault: Abnormal state that we can return to the normal state
- Failure: Abnormal state which is not a fault

This paper treats fault recovery and failure recovery of a multi-robot system separately.

3.1 Fault recovery scheme

A robot repeats or aborts the operation when a fault occurs. Abortion is to abandon the faulty operation and to execute the next operation. The process of abortion is as follows.

- Execution → Fault → Fault recovery

Here, fault recovery includes such operations as removing misplaced parts. To repeat is to re-execute the operation. The process of repeating is as follows:

- Execution → Fault → Fault recovery → Re-execution

The recovery process of the repeat scheme includes removing the misplaced part, re-gripping the part and so on.

Suppose that a robot succeeds with the probability p_k and a fault occurs with the rate $(1-p_k)$ in the common workspace and the time distribution is as follows.

- Execution time in the exclusive workspace: average $1/\lambda_k$
- Execution time in the common workspace: pdf $g_k(x)$, average $\bar{g}_k$
- Recovery time distribution: pdf $r_k(x)$, average $\bar{r}_k$

For the abortion scheme, the total operation time distribution in the common workspace can be expressed by g'_k and the system states are the same as in Table I, where g'_k is obtained as:

$$g'_k(x) = p_k g_k(x) + (1 - p_k) g_k(x) \otimes r_k(x) \qquad (3.1)$$

We classify the repeat scheme into exhaustive repeat (repeat-I) and preemptive repeat (repeat-II). The exhaustive repeat indicates that a robot repeats the task until the job is successful, and the preemptive repeat indicates that the fault recovery operation has a lower priority than the normal operation and robots recover faults only when the common workspace is empty. For the repeat-I scheme, system states are the same as in Table I and g'_k is obtained as:

$$g'_k(x) = p_k \cdot g_k(x) \otimes \sum_{i=0}^{\infty} (1-p_k)^i (r_k(x) \otimes g_k(x))^{(i)} \qquad (3.2)$$

In equation (3.2), (i) indicates an i-tuple convolution. System states for the repeat-II scheme with 2 robots are shown in Table III. In Table III, F(k) indicates that a fault occurred on robot k, and k∗ indicates that robot k is in fault and needs a fault recovery operation.

3.2 Numerical example of the fault recovery scheme

A work cell consists of two robots, robot A and robot B. The time distribution is as follows.

Table III. System state for the repeat-II scheme.

π1 : S(A) 0	π2 : S(B) 0	π3 : S(A*) 0	π4 : S(B*) 0	π5 : S(A) B
π6 : S(A) B*	π7 : S(B) A	π8 : S(B) A*	π9 : S(A*) B	π10 : S(A*) B*
π11 : S(B*) A	π12 : S(B*) A*	π13 : F(A) A*	π14 : F(B) B*	π15 : F(A*) A*
π16 : F(B*) B*	π17 : F(A) BA*	π18 : F(A) B*A*	π19 : F(B) AB*	π20 : F(B) A*B*
π21 : F(A*) B A*	π22 : F(A*) B*A*	π23 : F(B*) AB*	π24 : F(B*) A*B*	
π25 : E(A) A	π26 : E(B) B			

* : fault occurred on that robot

Robot A:

$\lambda_A = 1/30$ (1/sec)

operation time distribution in the common workspace:
constant 11 sec

fault recovery time distribution (abortion):
exponential distribution with
parameter 1/15

fault recovery time distribution (repeat):
exponential distribution with
parameter 1/25

fault rate: 0.05

Robot B:

$\lambda_B = 1/25$ (1/sec)

operation time distribution in the common workspace:
constant 15 sec

fault recovery time distribution (abortion):
exponential distribution with
parameter 1/25

fault recovery time distribution (repeat):
exponential distribution with
parameter 1/35

fault rate: 0.1

The number of jobs is (a) (20, 40), (b) (30, 30), (c) (40, 20). Table IV shows T, Ts, μ and $\mu*$ for the abortion, repeat-I, repeat-II and repeat-III schemes. The repeat-III scheme is a tentative scheme in which robot A recovers the fault with the abortion scheme and robot B does so with the exhaustive scheme. The average completion time depends on the number of jobs and the fault recovery scheme. For example, from Table IV, if the load of robot B is smaller than for robot A, the repeat-II scheme is more efficient than the repeat-I scheme.

The robot speed can be adjusted by changing the program. In a common workspace, it is reasonable that all robots operate with maximum speed, in order not to interfere with other robots. The robot with the longest completion time, say robot k, determines the total completion time of the work cell. Therefore, in order to reduce the completion time of a system, robot k has to operate at a maximum speed in the whole area. For the work cell that consists of two robots with a common workspace and satisfies Assumption 1, the total completion time does not depend on λ_j , j ≠ k, even though the execution time of robot k depends on λ_j. The proof is shown in Property 1. The abortion, repeat-I and repeat-III schemes belong to this class. For the case of the repeat-II scheme, that is not generally true.

Property 1: A work cell consists of robot A and B, and the job vector to be executed is Cv, Cv = { M,N }. Suppose that the work cell satisfies Assumption 1 and $M \cdot u_A \geq N \cdot u_B$, that is, robot B finishes the work faster than robot A. Then, the average completion time T does not depend on λ_B, the operation speed of robot B in the exclusive workspace.

Proof: From the states in Table I and equation (2.6), by letting $D_{i,j} = E[D_A(i,j)]$, it follows that

$$w_A - \bar{g}_A = \frac{\pi_1 \cdot p_{1,5} \cdot D_{1,5} + \pi_2 \cdot p_{2,5} \cdot D_{2,5} \cdot \pi_3 \cdot p_{3,4} \cdot D_{3,4} + \pi_6 \cdot p_{6,4} \cdot D_{6,4}}{\pi_1 \cdot p_{1,5} + \pi_2 \cdot p_{2,5} + \pi_3 \cdot p_{3,4} + \pi_6 \cdot p_{6,4}} - \bar{g}_A$$

Table IV. Comparison of the completion times.

Scheme	Job set	μ_A	μ_A*	μ_B	μ_B*	T (sec)	Ts (sec)
	(a)					1751.0	1700
Abortion	(b)	0.0240	0.0214	0.0235	0.0222	1398.6	1275
	(c)					1767.4	1670
	(a)					1891.0	1822.2
Repeat-I	(b)	0.0233	0.0198	0.0220	0.0205	1508.9	1366.7
	(c)					1863.9	1715.8
	(a)					1893.4	1822.2
Repeat-II	(b)	0.0233	0.0202	0.0220	0.0204	1485.5	1366.7
	(c)					1848.2	1715.8
	(a)					1873.2	1822.2
Repeat-III	(b)	0.0240	0.0203	0.0220	0.0208	1474.6	1366.7
	(c)					1818.1	1670.0
	(a)					1642.0	1600
No fault	(b)	0.0244	0.0225	0.0250	0.0238	1325.9	1230
	(c)					1703.9	1640

$$= \frac{\pi_3+\pi_6}{\pi_4+\pi_5} \cdot \int_0^\infty 1 - e^{-\lambda_A s}\, dG_B(s)$$

$$\cdot \int_0^\infty \frac{1-\int_0^\infty G_B(t+s)\cdot \lambda_A e^{-\lambda_A s}\, ds}{1-\int_0^\infty G_B(s)\cdot \lambda_A e^{-\Lambda_A s}\, ds}\, dt$$

$$= \frac{\pi_3+\pi_6}{\pi_4+\pi_5}\cdot C \qquad = \frac{\frac{1}{\lambda_A}+w_A}{\frac{1}{\lambda_B}+w_B}\cdot C,\ \text{C is constant w.r.t } \lambda_B$$

$$\text{so, } \left(\frac{1}{\lambda_B}+w_B\right)\left(1-\frac{\frac{1}{\lambda_A}+\bar{g}_A}{\frac{1}{\lambda_A}+w_A}\right)=C$$

Suppose that the speed of robot B is changed from λ_B to λ_B' and the average completion times are T, T′ respectively. Then, $T-T'=N\cdot C-N\cdot C=0$ ■

We can find the optimal load distribution ratio to obtain the minimal completion time for a load distribution problem. The ratio with the minimal average completion time depends on the fault recovery scheme. Table V shows ratios to obtain the minimal average completion time, and Figure 1 shows the average completion time with respect to the load ratio, where the load ratio is the proportion of the load of robot A to the total load. The work cell with a common workspace has a different optimal load distribution ratio from that of a work cell without a common workspace.

4. EXPERIMENTAL RESULT

The experimental robotic work cell consists of three robots as shown in Figure 3, and each robot is connected to a PLC and a supervisory PC. We perform an experiment on two robots among them. Each robot works in the exclusive workspace and the common workspace under the control of the PLC, and executes the program shown in Figure 4 for one job. Figure 5 shows the workspace of each robot.

The experimental task is typical assembly and material transporting. Each robot executes an assembly task in the exclusive workspace and carries it to the common workspace. In the common workspace, the operation times of robot A and robot B are constant, 13.57 sec and 12.54 sec, respectively. In the exclusive workspace, the average operation time is 40.71 sec and 24.81 sec respectively, and varies uniformly in the range of ±3 sec and ±2 sec. The number of jobs is (50,20), (35,35), (25,45), (20,50). The completion time T, estimated T and Ts are shown in Table VI and Figure 6. To obtain T, we averaged the completion time after four time experiments. The estimated value that is obtained from Algorithm 1 is close to the experimental result, even if the operation time in the exclusive workspace is not an exact exponential distribution. The optimal load distribution ratio is 0.4158, 0.4103 and 0.4076 for T, estimated T and Ts respectively (Table VII).

5. FAILURE RECOVERY OF A MULTI-ROBOT SYSTEM

A failure recovery method for a multi-robot system is to have another robot take the place of the robot that fails.

Table V. Load distribution ratio with minimal average completion time.

	Abortion	Repeat-I	Repeat-II	Repeat-III	No fault
With interference	0.4880	0.4920	0.4960	0.4920	0.4880
No interference	0.5040	0.5160	0.5160	0.5200	0.4920

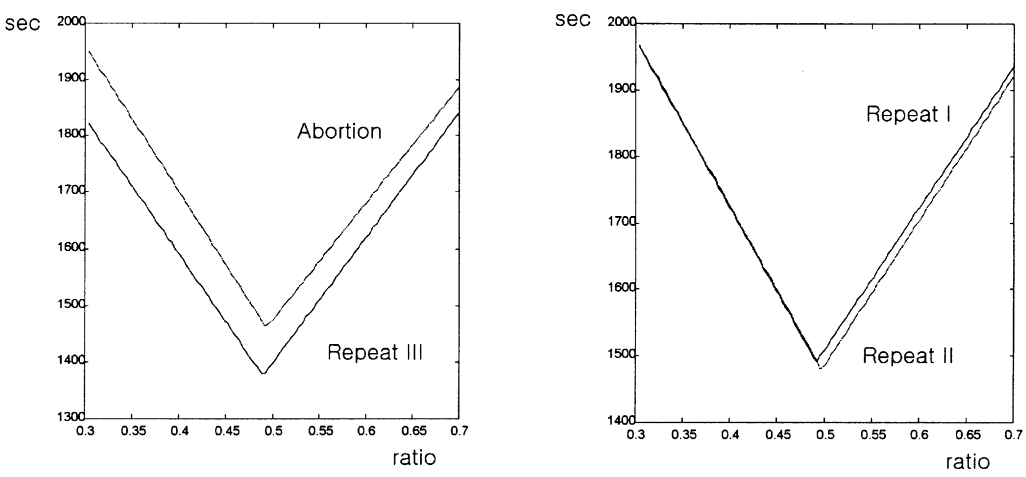

Fig. 2. Average completion time with respect to load ratio.

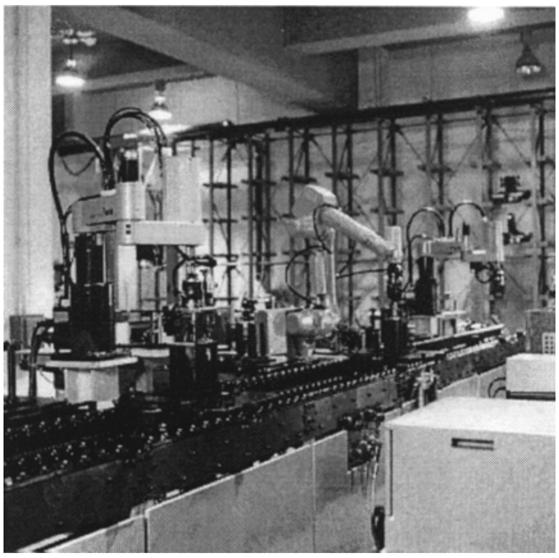

Fig. 3. Experimental multi-robot system.

Lee[18] proposed three interconnection methods for multi-robot systems, a single processor system (SPS), a hierarchically structured system (HSS) and a master slave system (MSS). Ignoring the command transmission time that is much smaller than the operation time, we find that the HSS and MSS have the same properties. Here, we measure the failure recovery performance of a multi-robot system that has a common workspace. Let the total number of jobs be C, and suppose that robots A and B fail with rates f_A, f_B per job, respectively. We measure the performance with T, PS, Pno and Tno, where PS is the probability that the work cell fails and the assigned work is not accomplished, Pno is the probability that no robot in the work cell fails and the work is finished successfully, and Tno is the completion time for this case.

```
        GOSUB EXCLUSIVE
        SIGNAL      1
10      IFSIG-1    GOTO 10
        GOSUB COMMON
        SIGNAL    -1
```

Fig. 4. Robot program for one job.

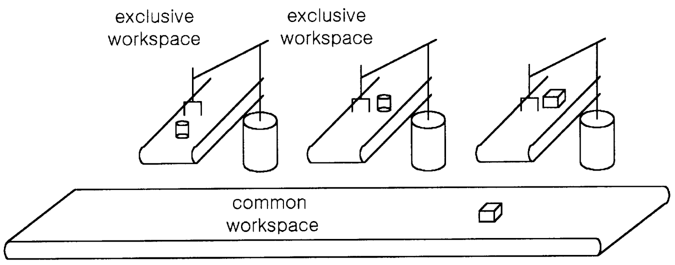

Fig. 5. Workspace defined in the experiment.

Table VI. Comparison of the completion time.

Job set	(50,20)	(35,35)	(25,45)	(20,50)
T (sec)	2242.8	1952.0	1757.4	1938.0
Estimated T (sec)	2223.6	1961.0	1758.7	1929.9
Ts (sec)	2171.2	1899.8	1680.7	1867.5

T : Experimental result.
Estimated T : The completion time obtained from Algorithm 1.
Ts : The completion time without interference

5.1 Failure recovery of the SPS

The SPS is a system that operates only one robot at each instant time as shown in Figure 7. The performance measures PS, T, P_{no} and T_{no} can be obtained by following the flow of Lee's[18] work since there is no queue time in the common workspace.

5.2 Failure recovery of the HSS/MSS

A HSS/MSS consists of several robots that work independently and an external controller, as shown in Figure 8. Let

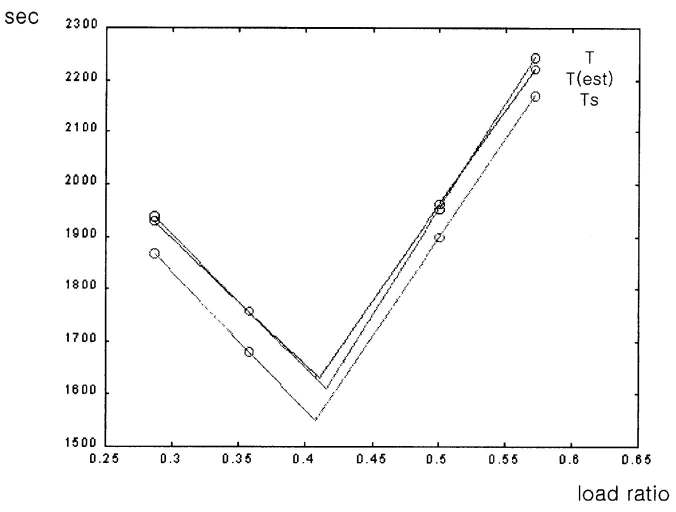

Fig. 6. Comparison of the completion time under the various load ratios.

Table VII. Systems utilized in the experiment.

Machine	Spec
Robot (2 sets)	SM3, Samsung electronic (Korea), Scara type Robot
Robot	ArcMate, Daewoo-Fanuc (Korea), 6DOF Revolute type
PLC	Master-K250, LG industrial system (Korea)
Main conveyor belt	LG industrial system (Korea)
Sub conveyor belt (3 sets)	LG industrial system (Korea)
Supervisory computer	IBM compatible PC

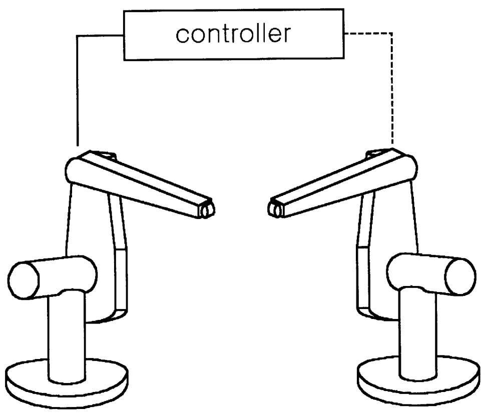

Fig. 7. SPS (Single Processor System).

robots A and B be assigned for jobs C_A, C_B and $C_A + C_B = C$. Then P_{no}, T_{no} is obtained as:

$$P_{no} = (1-f_A)^{C_A}(1-f_B)^{C_B} \tag{5.1}$$

$$T_{no} = \frac{C_B}{\mu*_B} + \left(C_A - \frac{C_B}{\mu*_B}\mu*_A\right)\frac{1}{\mu_A} \quad \text{if } \frac{C_A}{\mu*_A} > \frac{C_B}{\mu*_B} \tag{5.2}$$

$$= \frac{C_A}{\mu*_A} + \left(C_B - \frac{C_A}{\mu*_A}\mu*_B\right)\frac{1}{\mu_B} \quad \text{otherwise}$$

To obtain PS, we calculate PS_{jk}, j,k $\in$ (A,B) first, where PS_{jk} is the probability that robot j fails with the subsequent failure in robot k. $C_{kj}(i)$, which appears in Lee's[18] work, denotes the number of jobs that robot k finishes while robot j executes i jobs. That is,

$$C_{kj}(i) = \min\left\{ i\frac{\mu*_k}{\mu*_j}, C_k \right\} \tag{5.3}$$

Then, PS_{jk} is obtained as:

$$PS_{jk} = f_j f_k \sum_{i=0}^{C_j-1} [(1-f_j)^i (1-f_k)^{\text{int}(C_{kj}(i+1))}$$

$$\sum_{m=0}^{[C-i-\text{int}(C_{kj}(i+1))-1]} (1-f_k)^m] \tag{5.4}$$

Therefore, PS becomes

$$PS = PS_{AB} + PS_{BA} \tag{5.5}$$

In order to obtain T, we calculate the auxiliary variables, P_k and T_k, first. P_k is the probability that robot k fails but the work is accomplished by robot j, and T_k is the average completion time for this case.
Then, P_k is obtained as:

$$P_k = f_k(1-f_j)^{C_j} \sum_{i=0}^{C_k-1} (1-f_k)^i (1-f_j)^{C_k-i} \tag{5.6}$$

And T_k is calculated as:

$$T_k = f_k(1-f_j)^{C_j} \sum_{i=0}^{C_k-1} [(1-f_k)^i (1-f_j)^{C_k-i} A_k(i)]/P_k \tag{5.7}$$

where,

$$A_k(i) = I_{jk}(i)\left(\frac{C_j}{\mu*_j} + \left(i+1-\frac{C_j}{\mu*_j}\mu*_k\right)\frac{1}{\mu_k}\right)$$

$$+(1-I_{jk}(i))\left(\frac{i+1}{\mu*_k} + \frac{C_j - C_{jk}(i+1)}{\mu_j}\right) + \frac{C_k - i}{\mu_j}$$

$$I_{jk}(i) = 0 \quad \text{if} \quad C_{jk}(i+1) < C_j$$

$$= 1 \quad \text{otherwise}$$

Finally, the expected average completion time T is calculated as:

$$T = \frac{P_{no}}{P_{no}+P_A+P_B}T_{no} + \frac{P_A}{P_{no}+P_A+P_B}T_A + \frac{P_B}{P_{no}+P_A+P_B}T_B \tag{5.8}$$

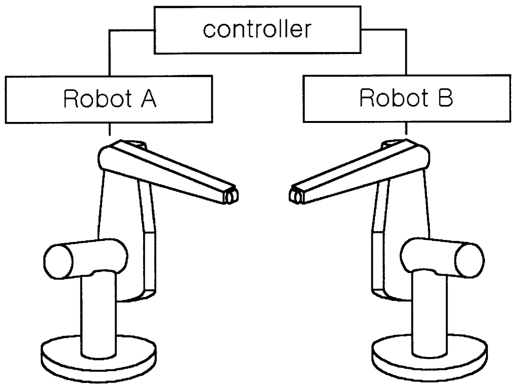

Fig. 8. HSS (Hierarchically Structured System).

5.3 Numerical example of the failure recovery scheme

Operation time distributions have the same parameters as given in Section 3. The number of jobs is (d) 1000, (e) 3000, (f) 5000, and the load distribution ratio is 1:1 for the HSS/MSS. For the SPS, the fastest robot, robot B in this case, takes full charge of the work and the slower one plays the part of standby. Robot A and robot B fail with rates 0.0001 and 0.0002 per job, respectively. Table VIII shows T, PS, Pno and Tno for the case of the SPS and HSS/MSS. We have the same value of PS for the SPS and HSS/MSS but different Pno depending on the failure rates of robots A and B. If the failure rate of the faster robot is larger than that of the slower one, the SPS has smaller Pno than the HSS/MSS.

Table VIII. Comparison of failure recovery performance.

System	Job set	T (sec)	PS	Tno (sec)	Pno
SPS	(d)	40095	0.0090	40000	0.8187
	(e)	120670	0.0671	120000	0.5488
	(f)	201550	0.1548	200000	0.3678
HSS/MSS	(d)	23305	0.0090	22098	0.8607
	(e)	75079	0.0671	66294	0.6376
	(f)	131111	0.1548	110490	0.4723
HSS/MSS (no interference)	(d)	21822	0.0090	20500	0.8607
	(e)	71124	0.0671	61500	0.6376
	(f)	125080	0.1548	102500	0.4723

6. CONCLUSION AND FUTURE WORK

In this paper, we build an embedded Markov chain model for the robotic work cell that has a common workspace. Robots in the work cell equipped in a sparse area affect each other. In particular, error occurrence in one robot brings a retardation of other robots through the common workspace.

For each fault recovery scheme, we measure the performance of the multi-robot system with an average completion time. We can select the most efficient recovery method for a given task as shown in Table IV, or we can find optimal load distribution ratio for a load distribition problem as shown in Table V. The work cell with a common workspace has a different optimal load distribution ratio from that of the work cell without a common workspace.

To recover the failure of one robot, the other robot takes the place of the robot that fails. In terms of average completion time, the HSS/MSS has a better performance than the SPS, as shown in Table VIII. However the HSS/MSS including N robots does not have N times as good a performance due to interference between robots. The SPS has larger Pno than HSS/MSS, if a large load is assigned to the robot that has a small failure rate.

We use the embedded Markov chain model; therefore, in the case that the operation time distribution in the exclusive workspace does not approximate to an exponential distribution, we incur a modeling error. Deterministic distribution is an example and in this case, even if Section 2.2 is still valid, it is hard to drive the analytic model, therefore we need other methods such as the timetable and experimental measurements. However, the experiment in Section 4 shows that the presented model is valid in general cases. We measure the performance of the system with a mean value. If robots that share the common workspace are many and the number of jobs is few, we need transient analysis. In future work, we need to extend this study to evaluate a transient performance.

References

1. B.H.Lee & C.S.G. Lee, "Collision-Free Motion Planning of Two Robots", *IEEE Transaction on Systems, Man and Cybernetics*, **SMC-17**, No. 1, 21–32 (1987).
2. Tadashi Nagata, Kunihiko Honda & Yoshiaki Teramoto, "Multirobot Plan Generation in a Continuous Domain: Planning by Use of Plan Graph and Avoiding Collisions Among Robots", *IEEE Journal of Robotics and Automation*, **4**, No. 1, 2–13 (1988).
3. John W. Roach & Michael N. Boaz, "Coordinating the Motions of Robot Arms in a Common Workspace", *IEEE Journal of Robotics and Automation*, **RA-3**, No. 5, 437–444 (1987).
4. David Gauthier, Paul Freedman, Gregory Carayannis & Alfred S. Malowany, "Interprocess Communication for Distributed Robotics", *IEEE Journal of Robotics and Automation*, **RA-3**, No. 6, 493-504 (1987)
5. James L. Peterson, *Petri Net Theory and The Modeling of Systems* (Prentice-Hall, 1981).
6. Weixiong Zhang, "Representation of Assembly and Automatic Robot Planning by Petri Net", *IEEE Transactions on Systems, Man and Cybernetics*, **19**, 418–422 (1989).
7. Jane Robinson & Alan A. Desrochers, "Performance analysis of a robotic testbed control architecture", *Proceedings IEEE International Conference on Robotics and Automation* (1990) **Vol. 3**, pp. 1782–1787.
8. Mengchu Zhou, Frank Dicesare & Dianlong Guo, "Modeling and Performance Analysis of a Resource-Sharing Manufacturing System Using Stochastic Petri Nets", *IEEE International Symposium on Intelligent Control*, (1990) **Vol. 2,** pp. 1005–1010.
9. Pedro Lima, Hugo Gracio, Vasco Veiga & Anders Karlsson, "Petri Nets for Modeling and Coordination of Robotic Tasks", *IEEE International Conference on Systems, Man and Cybernetics* (1988) **Vol. 1**, 190–195.
10. Paul Freedman, "Time, Petri Nets, and Robotics", *IEEE Transactions on Robotics and Automation*, **74**, 417–433 (1991).
11. K.N. Lee & D.Y. Lee, "An Approach to Control Design for Cooperative Multiple Mobile Robots", *IEEE International Conference on Systems, Man and Cybernetics* (1996) **Vol. 1**, 1–6.
12. I.H. Suh, H.J. Yeo, J.H. Kim, J.S. Ryoo, S.R. Oh, C.W. Lee and B.H. Lee, " Design of a Supervisory Control System for Multiple Robotic Systems", *IEEE International Conference on Intelligent Robots and Systems, IROS 96*, (1996) **1**, pp. 332–339.
13. W.M. Zuberek, "Schedules of flexible manufacturing cells and their colored Petri net models", *IEEE International Conference on Systems, Man and Cybernetics, Intelligent Systems for the 21st Century*, (1995) **Vol 3**, pp. 2142–2147.
14. V. Gopalakrishna, N. Viswanadham & Krishna. R. Pattipati, "Sensitivity Analysis of Failure-Prone Flexible Manufacturing Systems", *IEEE International Conference on Robotics and Automation* (1994) **Vol. 1**, 181–186.
15. Çağlan M. Aras & Ren C. Luo, "HMP: A Hierarchical Multiprocessor Computer Architecture for Multi-sensor Based Robotic Tasks", *IEEE International Symposium on Intelligent Control*, (1989) pp. 487–492.
16. Bertil A. Brandin, "The Real-Time Supervisory Control of an Experimental Manufacturing Cell", *IEEE Transactions on Robotics and Automation*, **12**, No. 1, 1–14 (1996).
17. S.J. Park & J.T. Lim, "Fault-Tolerant Robust Supervisor for Discrete Event Systems with Model Uncertainty and Its

Application to Workcell", *IEEE Transactions on Robotics and Automation*, 15, No. 2, 386–391 (1999)
18. K.D Lee, B.H Lee & M.S Ko, "A comparative model-based analysis and design for multi-robot systems", *Robotica*, **13**, Part 1, 65–76 (1995).
19. Chi-keng Tsai, "Multiple Robot Coordination and Programming," *IEEE International Conference on Robotics and Automation*, (1991). pp. 978-985 (1991)
20. Kleinrock, *Queueing systems*, Wiley (1975).
21. Ross. S, *Stochastic Processes*, Second Edition, (Wiley, 1996).
22. N. Viswanadham & Y. Narahari, *Performance Modeling of Automated Manufacturing Systems* (Prentice Hall, 1992).
23. Tapas K. Das & Martin A. Wortman, "Analysis of asymmetric patrolling repairman system", *European Journal of Operational Research*, **64**, 45–60 (1993).