Proceedings of 1998 IEEE International Conference on Systems, Man, and Cybernetics, pp. 2635-2640, 1998

A Study on Design Support System for Constructing Machine-Maintenance Training Environment Based on Virtual Reality Technology

H. Ishii, T. Tezuka and H. Yoshikawa
Graduate School of Energy Science
Kyoto University
Gokasho, Uji, Kyoto, 611-0011, Japan

ABSTRACT

A design support system has been developed for constructing VR-based training environment for machine maintenance work. The features of the developed design support system are : 1) users can construct various training environments under GUI (Graphical User Interface) environment, 2) the users need not any expert knowledge about computer programming, 3) Petri net model is utilized for representing the state transition of objects in virtual environment, and 4) the constructed environment can be easily changed and reused. In this report, the system configuration, how to model the state transition of objects with Petri net and the results of system validation experiments are described.

1. INTRODUCTION

Recently, Virtual Reality (VR) technology has emerged and been developing remarkably so that numerous attempts have been made to construct various training environments in virtual environment [1, 2]. The authors have developed a VR-based training environment for disassembling a check valve used in the Nuclear Power Plant (NPP), and it was found that the training environment based on virtual reality technology could be very effective for novice trainees to understand the procedure of maintenance work, the structure of the machines and so on[3].

However, there arises a serious problem for developing software systems for VR-based training. In fact, the workload of constructing the training environment is very large when a new training environment should be constructed for different kinds of training tasks.

On the other hand, there have been several systems developed which can reduce the workload for constructing virtual environments [4]. Division LTD., has developed a software called "dVISE" with which users can construct virtual environments without programming [5]. And Solidray Research, Inc., has developed a software called "RealMaster" which enables users to construct interactive virtual environments, in which trainees can manipulate objects with their hands [6]. With these systems, the users can construct a simple virtual environment under GUI. However, it is very difficult for users to construct complicated virtual environments in which trainees can disassemble/assemble machines by using input devices such as a 3D mouse.

In this study, we aim at constructing an effective user support system for multi-purpose VR-based training system, by which users can construct a VR-based training environment only by the guidance of GUI, by setting the information necessary for the training without coding programs. The major points in the system development are:

(1) Effective presentation of structured information on GUI for smooth guidance to construct the VR-based training environment, and
(2) Application of special Petri net model developed to model the state transition of objects in virtual environment.

Especially, the application of Petri net to the support system enables the users to design the relationship among each state of objects visually. Moreover by analyzing the developed Petri net, it is possible to search for the optimum procedure of maintenance work.

In this paper, a basic concept of the developed support system and how to model the state transition of objects are described. And after the system configuration is outlined, the results of the experiments to validate the developed support system are described.

2. BASIC CONCEPT

Firstly, it is necessary to define the term "object" for this training system in virtual environment; "object" includes various parts of the machines to be assembled or disassembled, tools to be used for assembly/disassembly work, and both hands.

To describe virtual environments without programming, the concept "state of objects" is introduced to a virtual environment. For example, assume that a trainee grasps a nut on a desk and drop it to the desk, the nut can take three kinds of states in virtual environment: "on a desk", "grasped by a hand" and "falling to a desk" (See Figure 1). And at each state, the nut behaves differently: at the state "grasped by a hand" the nut will attach to the

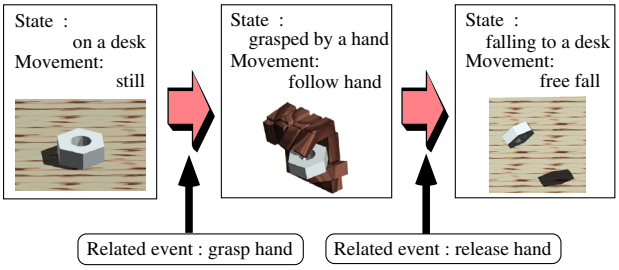

Figure 1: Relationship between state and its movement.

hand, at the state "falling to a desk" the nut will fall down (free-fall). And the state of the nut changes according to the relational events: an event "grasp a nut by a hand" translates the state of the nut from the state "on a desk" to the state "grasped by a hand".

Therefore, the management of "objects", their state transitions and their behaviors in a virtual environment can be described with two databases as follows:

- State-Transition Database
  The information on what kind of states the object can take and how the states of objects change on virtual environment.
- Motion Database
  The information on how the object moves at each state.

For constructing virtual environments with the developed system, the users have to construct these two databases. To make it easy for users to construct the databases, GUI environment has been constructed. Concerning about the Motion database, it is possible for users to make the Motion database by selecting items and setting the related numerical values. To make it possible to construct various training environments, many kinds of variable items must be prepared. In this system, about 60 parameters can be set: such as the surface of objects, the movement of objects, and so on. For the further details of these parameters, refer to the authors' preceding paper [7].

Concerning about the State-Transition database, a new method has been developed to model the state transition of objects in virtual environment, by which the users can input the state transition by constructing Petri net visually. The details of the way how to model the state transition of objects by Petri net will be explained in the next section.

3. PETRI NET

Petri net definitions

The Petri net model used in the developed support system follows the concepts and notations defined by ISO IEC [8]. Figure 2 shows the graphical notation of Petri net.

Terminology	Symbol	Terminology	Symbol
Single Place		Pool Place	
Normal Transition		Automatic Transition	
Normal Arc		Inhibitor Arc	
Reference Arc		Token	

Figure 2: Graphical notation of Petri net.

Hereafter, the color Token is also defined but it is used only at training simulation. Therefore, when constructing virtual environments, the users need not take care of the color Token. This means that the users need not any knowledge of color Petri net.

Definition 1 (HLP-net)
A HLP-net is defined as
$HLPN = (P, T, F, \sum; C, W_1, W_2, K, M)$ where

- $P = \{p_1, p_2, ..., p_m\}, m = |P| > 0$ is a finite set of elements called Places.
- $T = \{t_1, t_2, ..., t_n\}, n = |T| > 0$ is a finite set of elements called Transitions disjoint from $P \cap T = \emptyset$.
- $F \subseteq (P \times T) \cup (T \times P)$ is a finite set of elements called Arcs.
- $\sum = \{c_1, c_2, ..., c_l\}, l = \left|\sum\right| > 0$ is a finite set of elements called Colors.
- $C : P \rightarrow \sum$ is a color function. $C(p)$ is a color set of Token which can be marked at Place p.
- $W_1 : F \rightarrow \{0, 1\}$ is a weight function of Normal Arc. $W_1(p, t)$ is the weight of Normal Arc which connects p to t. $W_1(t, p)$ is the weight of Normal Arc which connects t to p.
- $W_2 : F \rightarrow \{0, 1\}$ is a weight function of Inhibitor Arc. $W_2(p, t)$ is the weight of Inhibitor Arc which connects p to t. In our support system, $W_2(t, p)$ is not defined.
- $K : P \rightarrow \{1, \infty\}$ is a capacity function of Places. $K(p)$ is the capacity of Place p.
- $M : P \rightarrow \sum$ is a set known as the marking of the net. $M(p)$ is a color set of Tokens which are marked in Place p.

where $\emptyset$ is an empty set and $|X|$ is the cardinality of a set X. A set of Input Places and a set of Output Places of Transition t are represented by $\cdot t$ and $t\cdot$, respectively. Likewise, a set of Input Transitions and a set of Output Transitions of Place p are represented by $\cdot p$ and $p\cdot$, respectively. And $W_1(p, t)$ and $W_2(p, t)$ can not be 1 at the same time.

Definition 2 (Single Place)
Single Place p_i is a Place which satisfies $K(p_i) = 1$ and is represented by a circle as shown in Figure 2.

Definition 3 (Pool Place)
Pool Place is a Place which satisfies $K(p_i) = \infty$ and is represented by a ellipsis as shown in Figure 2.

Definition 4 (Normal Transition)
For distinction from Automatic Transition defined at Definition 5, Normal Transition is defined as a Transition which does not fire before any events will occur even if the conditions for firing (Definition 9) are satisfied. Normal Transition is represented by a single bar as shown in Figure 2.

Definition 5 (Automatic Transition)
Automatic Transition is defined as a Transition which will fire whenever the condition for firing (Definition 9) is satisfied. Automatic Transition is represented by two parallel bars as shown in Figure 2.

Definition 6 (Normal Arc)
p_i is called as Input Place of a Transition t_j where $\exists p_i \in P$, $\exists t_j \in T$ and $W_1(p_j, t_i) = 1$. In graphical notation, it is represented by drawing an arrow from Place p_i to Transition t_j. The arc is called as Normal Arc for distinction from Inhibitor Arc defined in Definition 7. Likewise, p_i is called as Output Place of a Transition t_j where $W_1(t_j, p_i) = 1$. And it is represented by drawing an arrow from Transition t_j to Place p_i.

Definition 7 (Inhibitor Arc)
When $\exists p_i \in P$ and $\exists t_j \in T$ satisfies $W_2(p_i, t_j) = 1$, p_i is called as Inhibitor Input Place of Transition t_j. In graphical notation, it is represented by drawing an arc from Place p_i to Transition t_i which is terminated by a small circle at the Transition t_i. And the arc is called as Inhibitor Arc.

Definition 8 (Reference Arc)
Place p_i is called as Reference Place of $t_j \in T$ where $\exists p_i \in P$, $W_1(p_i, t_j) = 1$ and $W_1(t_j, p_i) = 1$. In graphical notation it is represented by connecting Place p_i and Transition $t_j \in T$ with a bi-directional arc. And the arc is called as Reference Arc.

Definition 9 (Enabling)
A Transition t is enabled when the all conditions as follows are satisfied:

1. If $W_2(p, t) = 1$ then $|M(p)| = 0$ where $\forall p \in \cdot t$
2. If $W_1(p, t) = 1$ then $|M(p)| > 0$ where $\forall p \in \cdot t$
3. If $|M(p)| \neq 0$ then $W_1(p, t) = W_1(t, p) = 1$ or $K(p) = \infty$ where $\forall p \in t\cdot$

When a Transition t fires, the marking of the net $M(p)$ is transformed to a new marking $M'(p)$ according to the following rule:

$$M'(p) = \begin{cases} M(p) \setminus m & \begin{pmatrix} p \in \cdot t, p \notin t\cdot, \\ W_2(p, t) = 0, \\ m \in C(p) \end{pmatrix} \\ M(p) \cup g & \begin{pmatrix} p \in t\cdot, p \notin \cdot t, \\ g \in C(p) \end{pmatrix} \\ M(p) & (p \in \cdot t \cap t\cdot) \end{cases} \quad (1)$$

where $\setminus$ is a relative complement.

Modeling with Petri net
Basically, a state of a object and a transition of states correspond to a Place and "fire" of a Transition, respectively. And marking a Place with Tokens represents that the object is in the corresponding state. The transition of objects' states is represented by the transition of Tokens. This means that one Token is assigned to one object. Concretely the procedure for modeling with Petri net is as follows:

(1) Decide the objects to be located in virtual environment
For example to model a task "Grasp a pen placed on a desk with a right hand and drop the pen to floor", the objects are desk, pen, right hand and floor.

(2) Extract the states which each object can take
For example, while a trainee grasps a pen on a desk with a right hand and drops the pen to floor, the states of the pen can be 4 kinds of states: "on the desk", "touched by the right hand", "grasped by the right hand" and "falling to the floor". Likewise the state of the right hand can take 3 kinds of states: "open", "touch the pen" and "grasp the pen".

(3) Decide events which cause the state transition of objects
In our support system, 5 kinds of items as shown in Table 1 can be selected as the event which causes the state transition of objects. For example, if the target task is "Grasp a pen placed on a desk with a right hand and drop the pen to floor", the events which cause the state transition are "touch the pen with the right hand", "grasp the pen by the right hand", "release the pen by the right hand" and "the pen contacts the floor". And in the support system the event "touch pen with right hand" can be set by selecting a prepared item "A contacts with B" and selecting "right hand" as "A" and "pen" as "B" respectively.

(4) Construct Petri net
The states extracted at step (2) and the events decided at step (3) correspond to Places and Transitions respectively. The states of objects before the event occurs and the states of objects after the event

Table 1: Items which can be selected as an event

Event	Example
A contacts with B	a spanner contacts with floor
detach A from B	detach right hand from a pen
open hand	open right hand
grasp hand	grasp left hand
satisfy a condition	a pen is grasped by right hand

occurs correspond to Input Places and Output Places respectively.

(5) Set initial states of objects in virtual environment
By marking initial Tokens in Petri net, the initial states of objects can be set for training. Here, the initial position of objects in virtual environment are determied by setting 3 dimensional position for each Token.

To model the state transition of objects more effectively, the following 5 definitions are introduced:

- Automatic Transition and Normal Transition
There might be situations that it is necessary to cause the state transition of objects without reference to any event. For example, to construct a virtual environment in which a trainee is warned when he got off all nuts which is not allowed to do so, the virtual environment must be designed to check whether the number of nuts is 0 or not. Therefore in this study, two Transitions are distinguished: "Normal Transition" which fires according to an event and "Automatic Transition" which fires without reference to an event.
- Inhibitor Arc
A virtual environment, "There are some nuts and volts in a box. Until a trainee gets off all nuts, he cannot get off any volts" can not be modeled if Petri net does not have any capacity to do zero-test [9]. Although some extensions can be proposed to compensate for this lack of capacity, Inhibitor Arc is introduced in this study, because of consideration of the simple conception. Not only this arc makes it possible to model the above virtual environment but also makes it easy to model a virtual environment in which a state of an object inhibits the state transition of the other object. For example, "A nut fixes a board on a wall" (The state of the nut inhibits a state transition of the board to a state "removed from the wall").
- Reference Arc
There is a state of an object which is the condition for a Transition to fire but does not change itself even if the Transition fires. For example, the condition for a trainee to operate an electric appliance is that the switch of power is up. But even if the trainee operates the appliance, "rotate dials", "slide levers" and so on, the power is still up. If the training environment is modeled with Petri net, the same Place comes to Input and Output Place of the same Transition. In this study, this Place is called as "Reference Place" and this arc is called as "Reference Arc".
- Pool Place
In case of locating plural objects which have same characteristics in virtual environment, the way to prepare a single Place for each object is redundant. Therefore in this study Pool Place is introduced which can be marked with plural Tokens. For example, by modeling the state "pen: on a desk" with Pool Place, it will be possible for trainees to locate plural pens on the desk.
- Color Token
In case that there are plural pens on a desk which a trainee can grasp, plural Tokens are located in the corresponding Pool Place. When the trainee grasps one of these pens, it is need to change the state of the pen which is touched by trainee's hand to the state "grasped by right hand" and need not to change the state of the other pens. In this way, in case that the state of objects changes, the objects must be distinguished. Therefore in this study, color Token is introduced.

Example of modeling with Petri net
Figure 3 shows an example of modeling the virtual environment "These are two pens on floor. A trainee can grasp and drop these pens with his right hand." In this figure, there are two Tokens in the Place which name is "pen:on floor" because the number of the pens on floor is two. Since the kind of the Place "pen: on floor" is Pool Place, a trainee can locate two pens on floor. But because the kind of the Place "pen: grasped by right hand" is Single Place, a trainee cannot grasp plural pens at the same time.

4. SYSTEM CONFIGURATION

As shown in Figure 4, the whole system consists of three sub-systems: (1) simulation sub-system using a graphic workstation, (2) display sub-system using CrystalEyes and a display monitor or a projector, and (3) sensing sub-system using a 2D, a 3D mouse and a keyboard. We have adopted OpenGL library for rendering 3D images, OSF/Motif library for constructing GUI environment and SGL library for stereo viewing with CrystalEyes. The 3D mouse is mainly used for pointing 3D position in virtual environment when a training environment is constructed. And when training, a trainee can choose gestures such as grasp hand, release hand, drop objects and so on by using the 3D mouse.

Figure 5 shows the interface of the developed system. All information necessary for constructing virtual envi-

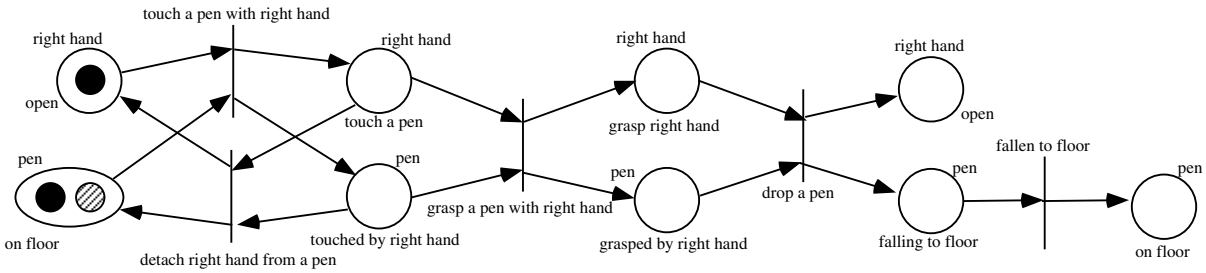

Figure 3: An example of modeling with Petri net.

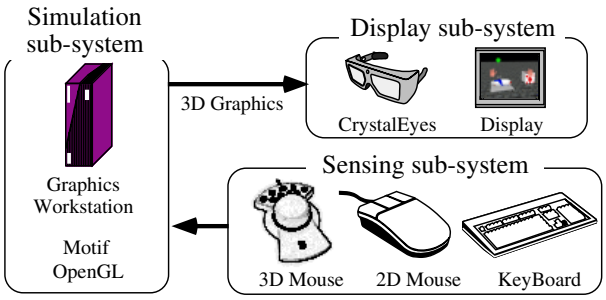

Figure 4: System configuration.

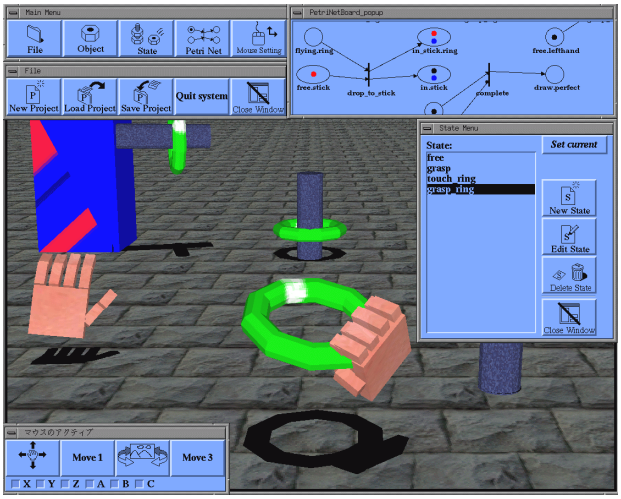

Figure 5: An example of Interface.

ronments can be fed under GUI environment without any programming. And in this system the users can input the state transition of objects in virtual environment by constructing Petri net visually.

5. SYSTEM VALIDATION EXPERIMENT

Construction of training environment

To verify that the efficiency for constructing virtual environment is improved by using the developed system, a training environment has been developed for disassembling/assembling a check valve used in NPP which is the same training environment developed with programming at the authors' previous work [3].

Figure 6 is a snapshot picture of the developed training environment. The developed training environment consists of 14 objects: pen, nut, volt, spanner and so on. To complete the maintenance of the check valve, about 100 actions are needed. To construct the training environment it costs 12 hours of system handling by single user. Table 2 shows the numerical result of modeling the training environment with Petri net.

Discussion

To be compared with the previous work which takes about 4 months by two programmers, the efficiency was remarkably improved. The major reasons for this remarkable improvement are as follows:

- In the previous work, it was necessary to invent algorithm for various simulations, since construction of training environment was made by coding programs. But using the developed system, the user did not need such kind of troublesome programming work.
- A large number of errors always occur during the course of constructing complicated training environment. When coding programs, debugging is very dif-

Table 2: Result of modeling with Petri net

Item	Number
Single Place	98
Pool Place	2
Normal Transition	128
Automatic Transition	4
Inhibitor Arc	16
Referece Arc	50

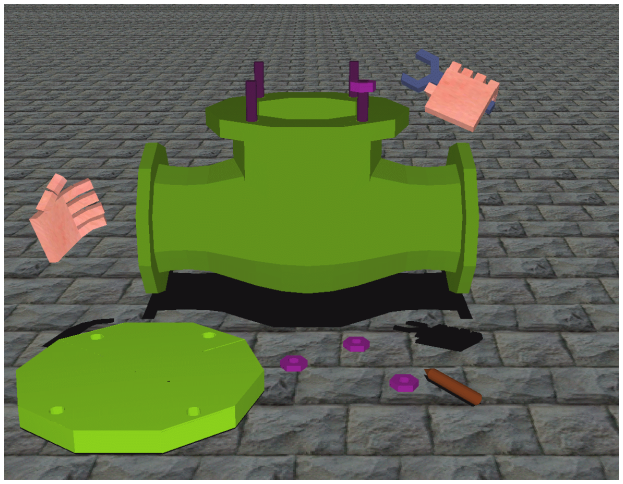

Figure 6: A snapshot picture of training environment.

ficult and time consuming. But using the developed system, errors can be corrected easily, because of the simple construction procedure.

But the other problems have risen. The more complex the target machine becomes, the larger Petri net must be constructed. If Petri net becomes too large, it becomes difficult to understand the structure of the net and to update it. In fact, the numbers of Places and Transitions of Petri net for the check-valve maintenance training are 100 and 132, respectively. It is too large to search for the firing sequence by constructing reachable tree [10]. To deal with this problem, it is necessary for us to design the following strategies for the futher work:

- The division of a large Petri net into sub-Petri nets.
- The application of the colored Petri net.

6. CONCLUSION

In this study, we have developed a construction support system for a VR-based training environment for machine maintenance work. The basic concepts of the system is that, (1) Users can construct a complicated training environment through GUI, and (2) An extended Petri net is applied to manage the object state transition. In order to design the support system, we have first pickd up the essential information to construct the VR-based training system, and then extended a Petri net in order to realize the efficient construction work for virtual training environment. In addition, we conducted the experiments to evaluate the system by comparing with our previous work. As the result, it was found that the system made the great improvement of the workload when constructing the training environment for a check valve maintenance. At the same time, it was pointed out that the complicated training tasks caused explosion of the number of Places and Transitions of the Petri net. For the futher work, a new strategy to divide and reconstruct the large Petri net should be developed.

7. REFERENCES

[1] R.J. Seidel and P.R. Chatellier, *Virtual Reality, Training's Future?*, NATO Defence Research Group, PLENUM PRESS, 1997.

[2] K. Arai, K. Abe and N. Kamizi, "Development of a simulator using virtual reality technology" *The Transaction of the Virtual Reality Society of Japan*, Vol. 2, No. 4, 1997, pp. 7-16.

[3] H. Yoshikawa, T. Tezuka, K. Kashiwa and H. Ishii, "Simulation of machine-maintenance training in virtual environment" *Journal of Atomic Energy Society of Japan*, Vol. 39, No. 12, 1997, pp. 72-83.

[4] J. Nomura and K. Sawada, *Virtual Reality*, Japan Society for Fuzzy Theory and Systems, Soft Computing Series 10, Asakura Publishing, 1997.

[5] R. Murakami, "dVISE" *Proceediings of the 5th IVR seminar*, Industrial Virtual Reality Show & Conference (IVR'97), 1997, pp. 69-80.

[6] K. Kamibe, "Real Master" *Proceediings of the 5th IVR seminar*, Industrial Virtual Reality Show & Conference (IVR'97), 1997, pp. 87-93.

[7] H. Ishii, T. Tezuka and H. Yoshikawa, "A Study on Design Support for Constructing Machine-Maintenace Training System by Using Virtual Reality Technology", (to be presented at 7th IFAC SYMPOSIUM MAN-MACHINE SYSTEMS, September 16-18, 1998, Kyoto, Japan).

[8] Committee Draft ISO/IEC 15909, *High-level Petri Nets - Concepts, Definitions and Graphical Notation*, Version 3.4, 1997.

[9] J. L. Peterson, *Petri net theory and the modeling of systems*, Prentice-Hall, Inc., 1981.

[10] R. Lipton, *The Reachability Problem Requires Exponential Space*, Reserch Report 62, Department of Computer Science, Yale University, 1976.