Proceedings of the 7th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Man-Machine Systems, pp. 389-394, 1998

A STUDY ON DESIGN SUPPORT FOR CONSTRUCTING MACHINE-MAINTENANCE TRAINING SYSTEM BY USING VIRTUAL REALITY TECHNOLOGY

H. ISHII *, **T. TEZUKA** * **and H. YOSHIKAWA** *

**Graduate School of Energy Science, Kyoto University, Gokasho, Uji-shi, Kyoto-fu, JAPAN*

Abstract. A design support system has been developed for constructing VR-based training environments for machine maintenance work without any expertise knowledge and programming effort on VR. Using the developed system, the users can easily construct various training environments under GUI environment. It was verified through some experiments that the developed system can reduce the working hours remarkably and that novice users who have no prior knowledge on the system could construct a training environment successfully after a few hours of tutorial on the system. In this paper, the system configuration and experimental results are described.

Key Words. Virtual reality ; Design systems ; Training ; Petri-nets ; User interfaces

1. INTRODUCTION

Recently, Virtual Reality (VR) technology has emerged and been developing remarkably so that it becomes now possible for us to apply VR technology for various training purposes (Miwa *et al.*, 1995; Arai *et al.*, 1997). In fact, it was reported by NASA that VR-based training had been successfully used for on-the-ground pre-training of space shuttle staffs who were in charge of Hubble telescope repairing in outer space (Loftin and Kenny, 1995). The authors have developed a VR-based training environment in which trainees can disassemble a check valve used in the nuclear power plant (Yoshikawa *et al.*, 1997).

Compared with the training systems based on the real machines or real-size mockups, VR-based training system has a number of advantages: economical, safe for the trainee, no need of large space, and so on. However, from the author's past experience, there arises a serious problem for developing software systems for VR-based training. In fact, the workload of constructing the training environments is very large when a new training environment should be constructed for different kinds of training tasks. The important point is that expert knowledge and skills on computer programming are required to construct the VR-based training system. And it is very difficult for those who are not so familiar with the programming technology to construct the VR-based training system for various kinds of machine, which are necessary for practical and effective training.

In this study, the user support system has been developed for constructing a VR-based machine maintenance training environment, which has the following features:

- The users can construct a training environment without programming effort,
- The users can feed the necessary information through Graphical User Interface (GUI),
- The users can set the state transition of objects in the virtual environment by visual construction of Petri net model (Peterson, 1981),
- The constructed environment can be easily changed and reused, and
- The users can execute the training simulation by the same system.

There have been several systems developed for constructing the virtual environment under GUI environment (Fujii *et al.*, 1996), but those existing systems are designed only for constructing simple virtual environments in which objects cannot be manipulated just like in real world. No support system has been yet developed for constructing complicated virtual environments for training purpose, where various kinds of machines and equipments can be freely assembled and/or disassembled with input devices.

In this paper, the basic idea for supporting the construction is described in section 2, the system configuration is illustrated in section 3, and then the results to validate the system in section 4.

2. CONSTRUCTION OF VR-BASED TRAINING ENVIRONMENT

2.1. *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 make it possible for trainees to manipulate object models just like in real world, the interaction between objects and physical laws must be simulated; for example, if a pen is grasped by a hand, the pen must attach to the hand. It is possible to detect the contact of each object with others by using the surface model information. But it is very difficult to judge whether or not each object would contact with each other and then if contacted, to describe the movement of contacted objects. It takes a lot of computation time to do so even by high performance computer.

So in this study, the concept "state of objects" is introduced to virtual environment and the information on how an object in virtual environment moves in accordance to the trainee gestures is prepared. To put it concretely, the users prepare "state transition database" and "motion database". The state transition database gives information on what kind of states the object can take and how the states of objects change in virtual environment. And the motion database gives the information on how the object moves at each state.

In the training system, the movement of each object is decided based on the motion database according to the present state. And if an event (e. g. a collision of objects) happens, all the states of the concerned objects are changed based on the state transition database. For example in case of an action "grasp a nut on a table", the relationship between the state and its movement is described as shown in Fig. 1. In this system, a user interface is constructed by which the users can construct these two databases efficiently under GUI environment.

Concerning the state transition database especially, 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 detail of how to model the state transition of objects by Petri net was published in the authors' preceding paper (Yoshikawa *et al.*, 1997). And concerning 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: on the surface of objects, on the movement of objects, and so on. The detail of these parameters will be explained in the next subsection.

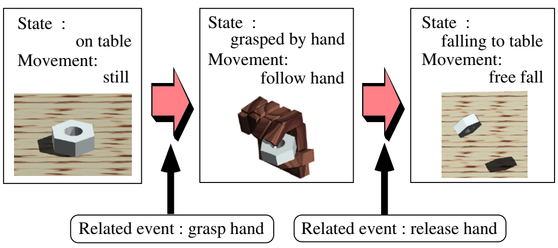

Fig. 1. Relationship between states and its movement.

But if the number of parameters to be set is so large, the user's work for setting those parameters becomes so time consuming that the efficiency of the system would not be expected. To cope with it, the system is equipped with "object template function" by which frequently used parameter set can be registered when it first appears and re-used repeatedly afterwards under GUI environment.

2.2. *Variable parameters*

The major parameters, which can be set in the developed system, are summarized in Table 1, with respect to types of the information, parameter numbers and the examples. By setting those parameters, various objects such as "open/close hand" , "falling object" and "flying balloon" can be rightly represented in virtual environment, by the way as will be explained below.

- Surface model of object
  In some cases, it is necessary for describing how an object will change its shape according to the states. For example, the shape of 'open hand' and that of 'grasping something' must be different. In the system, therefore, the file name of object's shape and texture must be set for each state.
- Initial position and direction after object changes its state
  In some cases, it is necessary for describing how an object will change its position and direction according to the states. For example, if a trainee grasps a pen, the position of the pen must change with the center of the hand. In the system, therefore, an initial position can be set for each state.
- Movement depends on time
  To simulate the movement of a motor, "free fall", and so on, it is necessary for the system to support to describe "movement depends on time". In the system, the movement of an object can be represented by time function and the user can select type of preset time function and set its coefficients.
- Limit of movement
  To simulate the interference of objects, it is necessary to limit the movement of objects. For example, in real world, an object can not

Table 1 Variable parameters

Information item	Number of parameters	Example
Surface model of objects	4	Texture (RGB format), Shape (DXF format)
Initial position and direction	7	Position (x, y, z), Direction (yaw, pitch, roll)
Movement depends on time	12	Sin., Cos. and linear function can be set.
Limit of movement	18	The limit of position and direction can be set.
Movement depends on gesture	12	Attach to the hand

penetrate into the other object. But to simulate this, the method of detecting the contact of objects by exact calculation is inappropriate because of large amount of computation time. In the system, the limit of movement can be set by numerical values.

- Movement according to trainee's gesture
  To make it possible for a trainee to manipulate objects with his hand, it is necessary for objects to move according to trainee's gesture. If trainee grasp a pen, for example, the pen must move by the same way as the trainee's hand. In the system, the relation between the movement of an object and trainee's gesture motion can be set freely by setting over 10 parameters.

Besides these parameters, some useful parameters can be set in the system; the sound can be set which is played when any event has occurred. Using this parameter, the collision of objects can be presented with real sound. Moreover, a simple shadow of an object can be drawn to make it easy for the trainee to manipulate objects in 3D world.

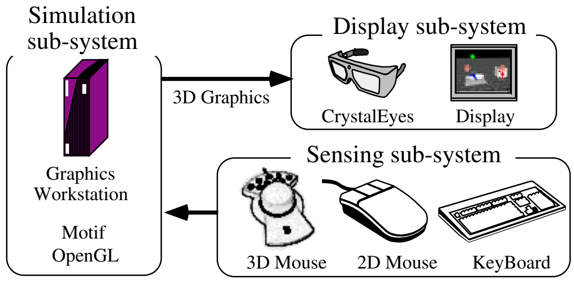

Fig. 2. System configuration.

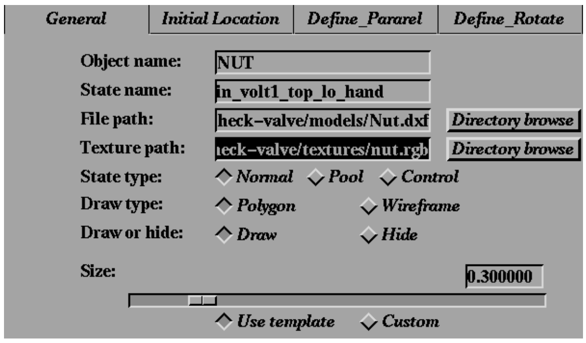

Fig. 3. An example of GUI for setting parameters for states.

3. SYSTEM CONFIGURATION

3.1. *System configuration*

As shown in Fig. 2 of system configuration, the system consists of, (i) simulation sub-system, (ii) display sub-system, and (iii) sensing sub-system. The simulation sub-system corresponds to the graphic workstation, the display sub-system includes CrystalEyes and a display, and the sensing sub-system corresponds to a keyboard, a 2D and a 3D mouse. We have adopted the OpenGL library for rendering the 3D images, the Motif library for constructing GUI environment and the SGL library for stereo viewing with CrystalEyes.

For feeding the necessary information, the user manipulates the 2D and 3D mouse, with viewing 3D images on the display. The 2D mouse is mainly used for selecting information and inputting object's name, and the 3D mouse is used for pointing the 3D position in virtual environment.

For training, the trainee manipulates a pack and buttons of the 3D mouse with stereo viewing by CrystalEyes. Through the 3D mouse, the trainee can choose gestures such as grasp a hand, release a hand, drop objects, and so on.

3.2. *The interface of developed system*

Examples of the interface of the developed system are shown in Figs. 3 and 4, respectively, for setting parameters and constructing Petri net. The users feed the necessary information for constructing training environments through selecting toggle buttons and setting numerical values. And in this system, the users can input the state transition of objects in virtual environment by constructing Petri net visually. Besides, the buttons of the system is designed as shown in Fig. 5, in order that the users can easily imagine the function at first sight of the buttons.

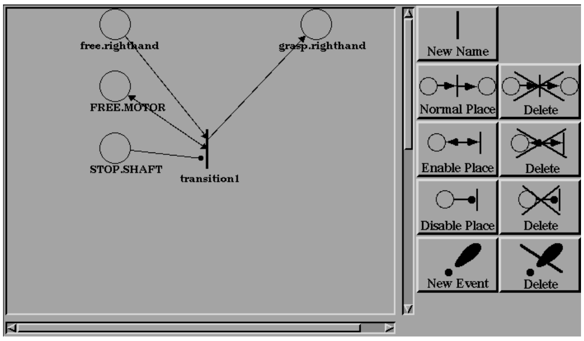

Fig. 4. An example of GUI for constructing Petri net.

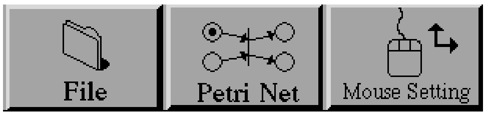

Fig. 5. An example of icon buttons.

3.3. *Procedure of constructing the training environment*

The construction of a training environment is made by the following steps:

1. Preparation of materials for training,
   The materials necessary for training are the 3D surface model of objects, texture and sound. Those materials are prepared by using an appropriate application software such as CAD(Computer Aided Design).
2. Preparation of "states" of objects,
   Secondly, the possible states the objects will take and the motion database must be provided.
3. Construction of Petri net,
   By constructing Petri net visually, the state transition database is created.
4. Setting of initial state of objects,
   By marking tokens in Petri net, the initial state of objects is set.
5. Training execution.

4. SYSTEM VALIDATION EXPERIMENTS

4.1. *Construction of complicated training environment*

The workload for constructing a training environment was experimented by the developed system, for assembling/disassembling a check valve, as an example of constructing a complicated training environment. Then it was compared with that of the authors' previous study (Yoshikawa *et al.*, 1997).

4.1.1. *The target machine.* The structure of a check valve is shown in Fig. 6. For simplification, the maintenance place is limited to the lid of the valve, and the number of volts and nuts

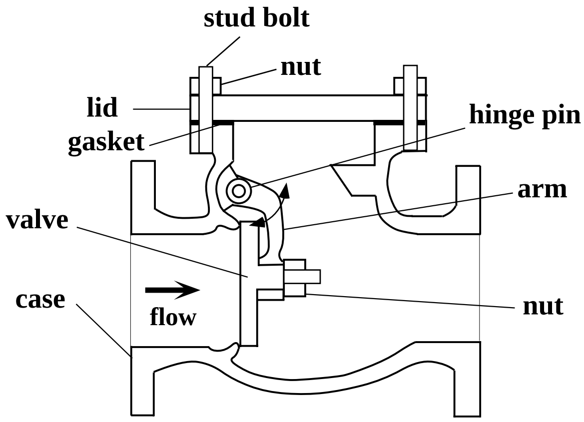

Fig. 6. Structure of check valve.

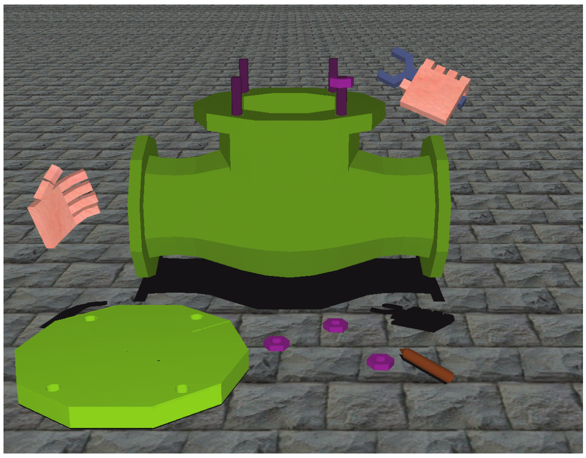

Fig. 7. An example scene of maintenance training of check valve.

is decreased from 16 to 4. The developed training environment consists of 14 objects: pen, nut, volt, spanner and so on. In this training environment, trainees can take various actions such as loose nuts with a spanner, check a mark on a valve with a pen, get off a lid of a valve, and so on. And the training environment is constructed by the same way, or as real as possible; for example, before a nut is loosen by a spanner, the trainee cannot loose the nut with his hand. To complete maintenance of the check valve, about 100 actions are needed. These conditions are almost the same as the previous study except for utilizing textures and sounds by the present support system.

4.1.2. *Results.* An example scene of the developed training environment is as shown in Fig. 7. In this experiment, it costs about 12 hours for a skilled user to construct this training environment. This is the time except for constructing 3D surface models, but compared with the previous study construction of almost same training environment (it costs about 4 months by two programmers), the efficiency of the workload is remarkably improved (see Table 2).

Table 2 Comparison of construction time

	previous work	with support
persons	2 (programmer)	1 (system user)
working time	4 months	12 hours

4.1.3. *Discussion.* The reasons for the remarkable improvement of efficiency by the developed system are as follows:

- In the previous work, it was necessary to invent the algorithm for various simulations, since the construction of training environments 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 will always occur during the course of constructing complicated training environments. In coding programs, debugging is very difficult and time consuming. But using the developed system, errors can be corrected easily, because of the simple construction procedure.

4.2. *Construction capability by novice users*

To verify that even novice users can construct a training environment using the developed system, two students were asked to construct a simple training environment.

4.2.1. *Subject.* The number of subjects was two (subject A, B). Both of them had no experience of constructing virtual environments with coding programs. They were accustomed to using the 2D mouse and the keyboard, so they could input necessary information very easily but they used the developed system for the first time.

4.2.2. *Methods.* The surface models of the target machine and textures were prepared beforehand with other application. Prior to the experiment, guidance course of 30 minutes was allocated to the subjects, to explain how to use the developed system and show them the procedure to construct an example environment, in which a trainee can grasp a pen and release a pen with his right hand.

The time necessary for constructing the indicated environment was measured and the results of the construction were also recorded. Moreover, after the experiment, questionnaires were given to the subjects about the developed system.

4.2.3. *Target environment.* The target environment is such that it is composed of a motor, a rotating shaft and a fan, and a motor can start and stop its rotation by touching a switch. The number of prepared surface models is 4; motor, shaft, switch and base of switch. Necessary information for setting the construction of rotating a shaft are shown in Table 3 and Fig. 8. It was predicted that the operation of constructing Petri net would be easy, but that feeding the information for the movement of a shaft would be difficult.

Table 3 State information of rotating shaft

parameter	value
state name	rotate
surface model	shaft.dxf
texture	shaft.rgb
state type	normal
draw type	polygon
draw size	1.0
initial position	none
translation	disable
rotation	able
axis of rotation	local z-axis
mouse operation	disable

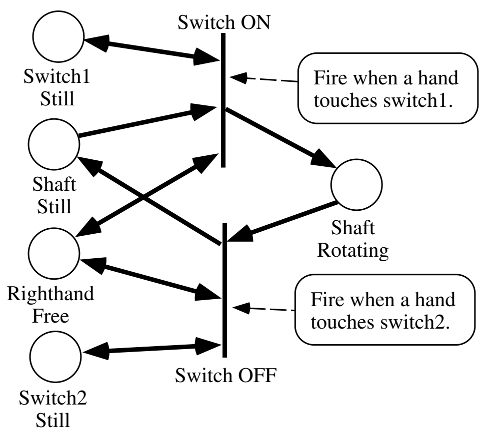

Fig. 8. Example of Petri net (state transition of rotating shaft).

4.2.4. *Results.* Both of the subjects could construct the objective training environment correctly. An example of the constructed training environment is shown in Fig. 9. The working time for the construction were 41 minutes(subject A) and 38 minutes(subject B). They could feed the necessary information for setting shaft movement, which was assumed difficult to feed. Subject B confused how to construct Petri net, but after trial and error, he could construct Petri net correctly. The results of the questionnaires from both subject are summarized bellow:

- At first I could not understand how each button works, but using the system for a few hours, I could understand how to use these buttons.

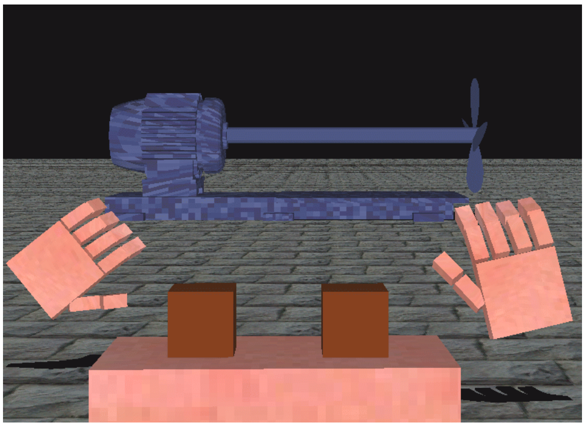

Fig. 9. A scene of training motor operation.

- With more familiarization with the system, I felt the motivation that I would like to construct more complex virtual environments.
- The positioning work of setting objects in 3D world was very difficult.
- If I would use the system for a longer time, I will be able to construct virtual environments more freely.

4.2.5. *Discussion.* It can be said from the above results that using the developed system, even novice users can construct VR-based training environments through trial and error if the environment is simple. Both of subjects have no experience of constructing virtual environments by coding programs, so if they try to construct virtual environments from the beginning, it will require much more time and larger workload because they need to study the basic knowledge of programming technique.

On the other hand, it was pointed out from the questionnaires that the interface for positioning the objects in 3D world should be improved. Since novice users who are not familiar with a 3D mouse are difficult to place objects correctly in VR space, alternative interface will be required for them, which will use a 2D mouse under GUI environment.

5. CONCLUSION

In this study, a support system was developed for constructing VR-based training environments without coding. The effectiveness of the system was confirmed through the validation experiments with respect to the reduction of workload and working hours and the familiarity with even novice users.

The developed system can be applied not only for training environments but also for generating other interactive virtual environments such as virtual show room, game, and so on. But in conjunction with the application of this system for actual training of maintenance personnel in the nuclear power plant, the following additional functions should be required:

- The function that offers alarms or advice when the trainee executes an improper task during "self-learning" course, and
- The function that offers, in virtual environment, the automatic performance of tasks to be executed next, for "demonstrative" teaching.

There is also a problem in the present system, that the more complex the machine for training 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. To deal with this problem, the application of colored Petri net is in consideration for future works.

6. REFERENCES

Arai, K., K. Abe and N. Kamizi (1997). Development of a simulator using virtual reality technology. *The Transaction of the Virtual Reality Society of Japan* **2**(4), 7–16.

Fujii, T., T. Yasuda, S. Yokoi and J. Toriwaki (1996). Visual simulation system for city planning with realtime interactive processing. *The Transaction of The Institute of Electrical Engineers of Japan* **116-C**(1), 36–42.

Loftin, R.B. and P.J. Kenny (1995). Training the Hubble space telescope flight team. *IEEE, Computer Graphics and Applications* **15**(5), 21–37.

Miwa, S., T. Ueda and S. Nishida (1995). A learing environment for maintenance of plants and equipments based on virtual reality. *The Transaction of The Institute of Electrical Engineers of Japan* **115-C**(2), 203–211.

Peterson, J.L. (1981). *Petri net theory and the modelling of systems.* Prentice-Hall,Inc.

Yoshikawa, H., T. Tezuka, K. Kashiwa and H. Ishii (1997). Simulation of machine-maintenance training in virtual environment. *Journal of the Atomic Energy Society of Japan* **39**(12), 72–83.