A System Allowing Concurrent Design and Implementation of Both Virtual and Real Robots

Toshiaki OOMORI, Norihiro ABE
Kyushu Institute of Technology
680-4 Kawazu, Iizuka, Fukuoka 820-8502, Japan
Email: oomori@sein.mse.kyutech.ac.jp

Hirokazu TAKI
Wakayama University
930 Sakaedani, Wakayama-shi
Wakayama 680-8510, Japan

Shoujie He
Eastman Kodak Company,
Plano, Texas, USA

Abstract

By the development of the VR technology in recent years, virtual debugging that uses a virtual environment to debug the robot is paid to attention. However, at the present stage the reliability of virtual debugging is low, and it is difficult to create and operate virtual robots without exclusive knowledge. Then, in this research, we use a virtual space according to rigid body dynamics, and aim the systems that we can create and operate virtual robots without exclusive knowledge.

Keywords: *VR, Virtual Robots, Virtual Space*

1 Introduction

In late years, more large-scale and complex embedded software is required. This makes it difficult to develop high-quality embedded software in a short term, and the delay of development is caused because developers cannot debug software until a real machine is completed. Consequently, a virtual machine should be built concurrently with a real one. Then this concurrent development allows developers to write and debug using a virtual robot the software to be installed into the corresponding real one. This makes it possible for developers to further examine the real machine in detail by downloading the software developed with the virtual one to the real one.

And as robots developed so far are supposed to act in safe environment, developers do not try to make them work in dangerous environment for fear that they may be broken owing to a tumble on a convex-concave road or falling from a high position.

In such cases, virtual robots will help them analyze the cause of accident, and improve fragile components of robots or incomplete control program leading robots to destruction.

In this research, reproducing the motor and the sensor, etc. in a virtual space according to a physical low, we aim at building the virtual robot that can operate as if it were the real machine. In addition, we want the system to assist a naive user in both designing a virtual robot and debugging the software.

2 Construct of Virtual Space

A virtual machine designed based on the traditional static does not contribute to develop the corresponding real one, as it does not work well in real environment comparison with the fact that the virtual one works well in virtual environment because of the lack of dynamics including inertia. A real time solid body physics engine VORTEX (developed by CMLabs Simulations, Inc.) introducing dynamics into virtual environment is exploited to solve the above problem.

VORTEX provides following two restrictions:

Joint, which joins two elementary parts to construct an articulated body.

Contacts, which restricts movement of two geometrical models within tolerance.

As a sample of VORTEX, the robot arm is shown in Figure 1. This simulation faithfully reproduces the gravity, friction, and the torque of the motor, etc. in real time. Using VORTEX, we can execute the simulation of complicated models with a lot of physical parameters in real time.

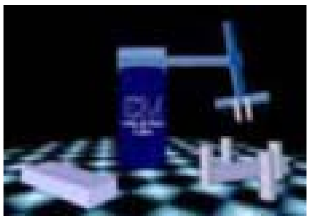


Figure 1: Robot arm simulation by VORTEX

3 Development of Virtual Robot

VORTEX has many features but some of them are unnecessary for constructing a virtual robot. Then we took out only function necessary for the robot, and redefined them as the C++ class for facilitating to construct a virtual robot. As a result, we can change the specification of virtual robots, and shorten the debugging time. Using this class, we made a four-legged virtual robot and a two-legged one shown in Figure 2 and Figure 3, respectively.

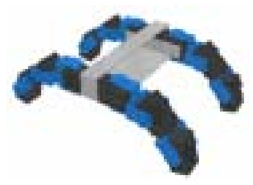


Figure 2: Four-legged robot

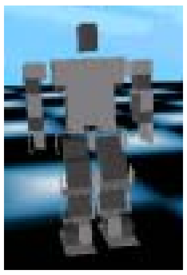


Figure 3: Two-legged Robot

To model a servomotor of virtual robot, a hinge joint of VORTEX is used and controlled. The hinge joint restrains the two parts to rotate around one axis as shown in Figure 4.

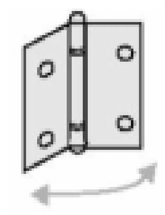


Figure 4: Hinge-joint image

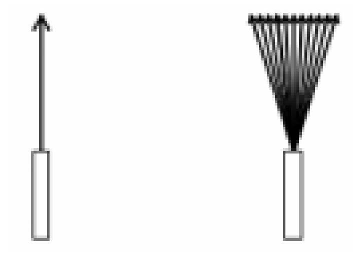
Moreover, we built in the walking program in both robots. It is difficult to have a two-legged robot walk stably though it is easy to have a four-legged robot walk. But, to make a four-legged robot pass the centerline of a corridor, a virtual gradient sensor is needed. Virtual sensors introduced into the system will be shown below.

4 Virtual Sensor

The robot needs a variety of kinds of sensors for both detecting its pose and localizing itself. Then, a virtual distance sensor and a virtual gradient sensor were invented. We configured the function of these two virtual sensors as C++ classes to make it easily for a developer to exploit them. This system allows a developer to implement the movement algorithm of the robot easily.

4.1 Virtual Distance Sensor

We made a sensor with high directivity like the laser distance sensor. And by combining the sensors together, a low directional sensor is obtained. Figure 4 shows these two sensors.


High directivity Low directivity
Figure 4: Distance sensor

4.2 Virtual Gradient Sensor

The mechanism of this sensor is very easy. Non-graphical two objects are set up in the model to be measured its gradient, and the grade is calculated from coordinates of the two objects.

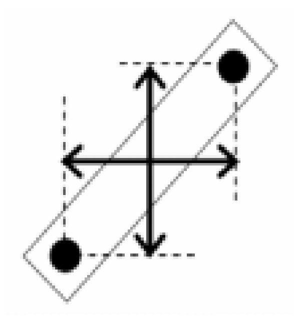


Figure 5: Gradient sensor

5 Installation of Virtual Sensor to Virtual Robot

Because a developer can make the sensors of various specifications expeditiously, the virtual sensor class makes it brief to devise the movement algorithm of the robot by the trial and error.

Figure 6 shows a four-legged virtual robot in which four distance sensors are installed. This robot measures the distance to the wall with the sensor, and it is possible to walk without knocking against the wall. A developer can easily adjust the number and the time base range etc. of the sensor, and this leads the developer to the best operational condition. For instance, when we install two sensors on the forefoot of the robot, as it collides against the wall, it was not able to walk well. It was, however, possible to walk well when 4 sensors were installed on all feet of the robot. Thus, the developer can judge appropriate allocation of the sensors using a virtual robot instead of a real one.

Figure 7 shows a two-legged virtual robot equipped with a gradient sensor at the center of its body. So as not to fall, this robot corrects the angle of the joint by detecting the inclination of the body. It is very useful for making a real robot to make such a correction algorithm in a virtual one. The reason is clear considering the case where the algorithm is developed from the scratch using a real machine. The virtual debug allows a developer to operate both a real and virtual robot at ease, as the result the real machine debugging will complete briefly. The risk a real machine falls will decrease, and its breakdown also decrease as a result. On the other hand, it is difficult to expect the same result when a real machine is debugged from the scratch.

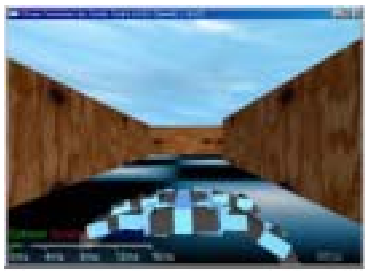


Figure 6: Four-legged robot with distance sensor

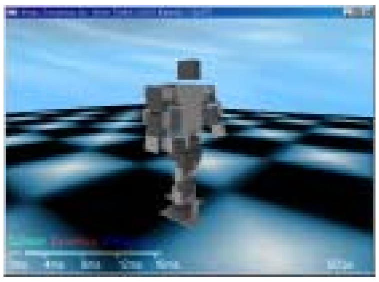


Figure 7: Two-legged robot with gradient sensor

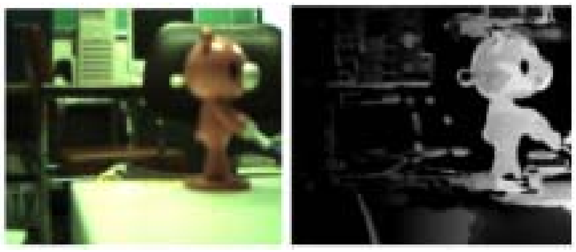
6 Stereo Vision System

It is difficult for robots to detect complex external world only by the distance sensor, then stereovision system is often built into it recently. This system calculates the distance to the object by the image data processing with two cameras.

At first, the parallax is calculated by extracting corresponding parts (called stereo matching) from two images captured with two cameras. Next, the distance to the object is obtained by using the acquired parallax and the distance between two cameras. For example, Figure 8 shows result of a stereo matching.

For simulating the stereo matching in virtual space, the aspects (two frame buffer of GL) captured with two cameras corresponding to the robot's eyes must be converted into the bit map images. Next a stereo matching algorithm is applied to them. This procedure is shown in Figure 9.

C++ class for implementing the stereovision system is so that the user might easily make eyes of virtual robots.


Left camera Image Depth image
Figure 8: Stereo matching example

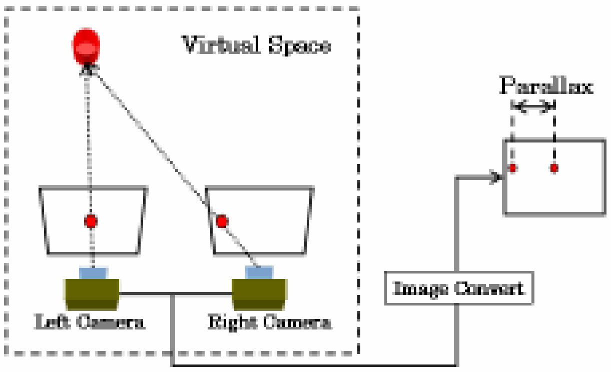


Figure 9: Procedure of stereo vision system

7 Combine virtual robot and stereo vision

We expect a virtual robot equipped with the stereo vision to following behavior.

A four-legged virtual robot will successfully avoid obstacles with the stereo vision system and reach the destination. The distance measurement using both stereo vision and virtual distance sensor will help the robot make an elaborate plan.

We want a two-legged virtual robot go up and down the stairs of which height and size are unknown. Many robots which can go up and down the stairs are given in advance such parameters. On the other hand, when our robot finds the stairs, it measures the height with the stereo vision. If there is enough space on the stair and the height is less than the tolerance, it will go up on the stair. At the moment, the robot will measure only the vicinity of its feet for conducting measurement of high-speed and high-accuracy.

8 Conclusions and Future Work

This research has aimed at making a virtual robot with the reality. Building virtual sensor and servomotor, etc. into VORTEX, we can make virtual robot with high reproducibility. And, by creating a variety of C++ classes, developers can make and change a virtual robot easily. In the future, various sensors and actuators should be provided to virtual machines and this will help us develop complex virtual robots that we cannot construct up to now. We would like to show how concurrent robots development would be able to solve various problems including development of a complex robot like a humanoid.

Acknowledgements

We greatly appreciate the aid of Ministry of Internal Affairs and Communications (MIC) and the Grant-in-Aid for Scientific Research.

References

[1]Yi PAN, Norihiro ABE, Kazuaki TANAKA：The virtual debugging system for embedded software development, ICAT2002, pp.119-p124, 2002

[2]Yi PAN, Norihiro ABE, Kazuaki TANAKA：The Virtual Debugging System for Embedded Software Development, SCI2003,pp78-84 , 2003

[3]Yi PAN, Norihiro ABE, Kazuaki TANAKA：The Virtual Debugging System for Embedded Software Development, VRAI2003, 2003

[4]Yi PAN, Norihiro ABE, Kazuaki TANAKA, Hirokazu TAKI：The Virtual Debugging System for Developing Embedded Software using Virtual machinery, EUC2004

[5]Toshiaki Oomori, Norihiro Abe, Kazuaki Tanaka, Hiroaki Taki, Shoujie He "Concurrent Development of Virtual Robots and Real Robots Based on Physical Law", 15th International Symposium on Measurement and Control in Robotics (ISMCR2005)