

Sensing in the Virtual World – Work in Progress

Mohamed Dekhil¹, Jed Marti¹, and Thomas C. Henderson²

¹Center for Engineering Design

²Department of Computer Science

University of Utah

Salt Lake City, Utah 84112, USA

Abstract

We examine the simulation of notional sensors and sensor fusion systems embedded in Virtual Reality training systems. Without building either the hardware or (much) software for such systems, we simulate their characteristics in a real-time, Distributed Interactive Simulation (DIS) environment. Cost-benefits analysis of simulation outcome permits us to refine or reject designs without committing to implementation. This work is part of a project building a practical simulation-based design framework for teleoperated mobile robots. Virtual Environments provide a structured synthetic background for notional sensor simulation.

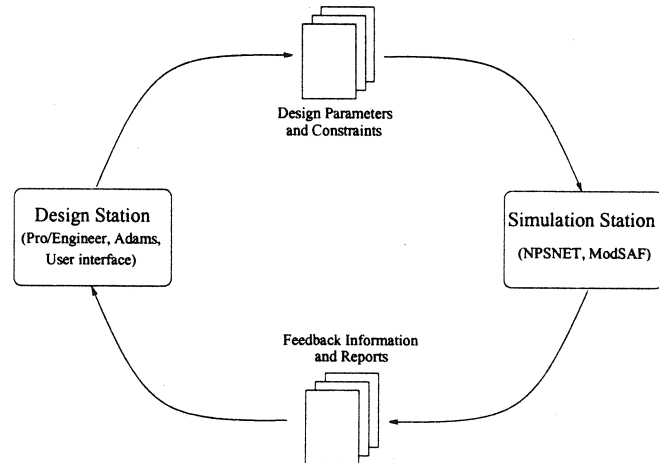


Figure 1: Simulation-based design cycle.

1 Introduction

The emerging concept of Virtual Reality is proven useful in such diverse fields as medicine, military training, entertainment, architectural design, and education. A Virtual Reality system consists of a *synthetic environment* and a *human-computer interface*. The real-time interaction between the user and the synthetic environment is mediated by multiple sensor modalities and displays. Some of the topics related to the use of sensors systems in virtual reality can be found in [Burdea and Coiffet, 1994, Kim *et al.*, 1994, National Research Council, 1995].

Simulation-based Design (SBD) is a cost-effective, time-saving approach for designing and testing new systems. It utilizes virtual reality tools for modeling, testing and analyzing new systems before attempting to build them. This is particularly useful if creating the virtual world is cheaper or less hazardous than implementing a prototype. A research project for building a practical framework for designing and testing teleoperated mobile robots using the SBD approach, at the Center for Engineering Design at the University of Utah, consists of a design station and a simulation station (see Figure 1). The design station is used to define the robot in terms of geometry, kinematics, dynamics, actuation, sensing, and communi-

cations. This information is passed to the simulation station which generates a 3D representation of the synthetic environment. We are using this framework to design and test these teleoperated robotic entities for specific tasks and operations.

An intelligent sentry uses many sensors; a telerobotic vehicle must "see" its environment, an automatic target designator must locate its target. Sensors are essential components of mobile robots. To simulate sensors and sensor fusion systems, we must model sensor characteristics and augment the synthetic environment to support sensor perception.

In simulating sensors, there are two approaches to generate sensor data:

1. Using synthetic data given the sensor model and parameters, its position and orientation, and a description of the scene in the sensor's range.
2. Using pre-recorded real data from a real sensor in a real environment in similar situations and conditions.

These approaches are discussed in [Chen *et al.*, 1994] and the simulation of several types of sensors is described. We will be using the first approach to generate the sensed data from the 3D model of the environment.

This work was supported by the Advanced Research Projects agency under Army Research Office grants number DAAH04-93-G-0420 and N00014-92-J-1447.

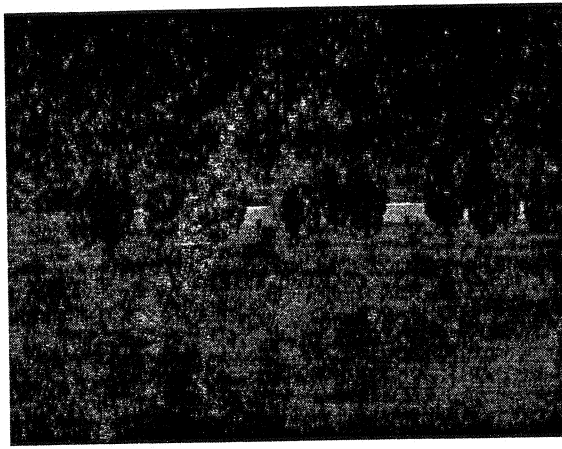


Figure 2: A scene from DIS simulation.

2 Background

Distributed Interactive Simulation (DIS) is used to perform multiuser simulations on networked computers. The main advantage of DIS is that it allows participants to join a simulation session and interact with the virtual environment and with other participants in the simulation. DIS has a standard protocol called IEEE 1278.1 which allows different simulators on different platforms to participate in the same simulation [Ins, 1994]. The DIS protocol does not specify implementation of hardware. Our simulation is in fact a suite of programs communicating over a standard Ethernet using the DIS protocol standard. For example, NPSNET is an interactive simulation system developed at the Naval Postgraduate School [Macedonia *et al.*, 1994, Nav, 1995]. NPSNET uses the DIS protocols for the interaction of multiple users on networked computers. NPSNET represents a good example of using virtual environments and modeling the environmental parameters that affect the visual perception of the scene.

NPSNET requires an advanced Silicon Graphics Workstations for effective operation. It is written in C++ using the IRIS Performer library which provides routines for building 3D real-time animations [Sil, 1994]. Figure 2 shows a scene from NPSNET. We augment NPSNET to simulate different sensors and to accommodate mobility, sensing, and communication constraints.

3 Visual Perception of Sensor Output

Visualization of sensor output can take several forms. Choosing the most suitable form is a challenging problem in itself. Sensor output can be classified based on the following factors:

- type of output.
- number of data items in a single reading.
- range of output.
- resolution.

For example, the output of a certain model of black and white camera is an intensity image with 460×240 data items, and with range from 0 to 255 and resolution of 200 pixel/mm², while the output of a certain model of sonar sensor is a single value with range 20mm to 3000mm and resolution of 0.5mm.

The output of a camera can be represented by the 2D version of the field of view, after adding a predefined distortion and environmental effects such as time-of-day, fog, etc. The output of a sonar sensor can be represented by a line starting at the sonar sensor and ending at the nearest object within the sonar field of view.

The communication method used also affects the quality and rate of the generated output. For example, noisy communication channels will produce lower quality images and in some cases might totally distort the image. Also, a low bandwidth channel will lower the rate of displayed images. Simulating the communication channels provides a more realistic sensor output which is required in military training application.

4 An Example: Simulating a Camera

Simulating a camera requires the generation of "realistic" images from the 3D scene. For example, a zoom lens with a wide field of view, mounted on a small mobile robot, might distort its image so badly that a human driver would have trouble locating significant obstacles in time to avoid them.

The following factors affect the generated images:

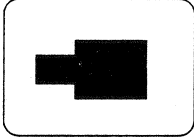
- Camera parameters such as focal length, resolution, radial distortion, etc. These parameters are defined using a data-entry system with a graphical user interface as shown in Figure 3.
- Surface characteristics such as specular, ambient, diffuse, emission, etc. These parameters are associated with the geometric database for the terrain and the objects constituting the virtual environment. It is important to note that most existing terrain databases do not provide this information, and it must be inferred. Attempts have been made to integrate actual sensor data into the synthetic terrain [Brendley and Grossman, 1992].
- Environmental effects such as time-of-day, fog, smoke, etc. These effects are supplied by the IRIS Performer library used in drawing and animating the scene.

The following are the steps for creating the simulated image given the previous factors and parameters:

1. The 3D scene is generated using the IRIS Performer library and the environmental effects are added to the scene as desired.
2. A snapshot of the scene is taken that represents one animation frame. This is done using a perspective projection with pinhole camera geometry.

Sensor Name

Sensor Type



Visual Icon

Characteristics:

Resolution lines rows

Focal Length Accuracy

Range from to Distortion Factor

Zoom Range to Pixel Inner-distance

Probability of Failure Cost

Figure 3: user interface for defining sensors.

3. The sensor perception module is used to modify the generated snapshot according to the sensor type and parameters. For example, focal length is used to determine the field of view for the generated image.
4. The processing module takes the generated images and performs any necessary image processing tasks such as edge detection, grouping, etc.

Figure 4 shows a schematic diagram of the simulation station with its different modules.

As an example, consider modifying the image in the sensor perception module to simulate radial distortion. The radial distortion factor causes geometric displacements and warping to the generated image due to the laws of geometric lens optics [Wolff *et al.*, 1992]. The equations for calculating the distorted pixel locations as defined in [Ame, 1980] are as follows:

$$X_d + D_x = X_u$$

$$Y_d + D_y = Y_u$$

where, (X_d, Y_d) is the distorted image coordinate on the image plane, (X_u, Y_u) is the undistorted (ideal) image coordinate, and D_x and D_y are defined as:

$$D_x = X_d(k_1 r^2 + k_2 r^4 + \dots)$$

$$D_y = Y_d(k_1 r^2 + k_2 r^4 + \dots)$$

where k_i are the distortion coefficients. In most cases, it is sufficient to consider only the first term, k_1 . In this case the distorted coordinates can be calculated as:

$$X_d = \frac{X_u}{1 + k_1 r^2}$$

$$Y_d = \frac{Y_u}{1 + k_1 r^2}$$

Figures 5, 6, 7 show the effect of applying these equations to a synthetic image of vertical and horizontal lines and to a snapshot image generated from the DIS simulation using IRIS Performer. Two different values for the radial coefficient k were used, 0.005 and 0.01.

The processing module can be either software or special hardware component (e.g., Datacube). The idea is that, once the synthetic image has been created, we can use "traditional" image processing techniques to analyze the generated image and to supply feedback information about the scene.

5 Conclusion

Sensing in the virtual world is an essential part of creating a realistic virtual reality application. Also it is useful as a testing framework for new sensors and sensing techniques. An example of simulating a camera was presented. Currently we are working on the implementation of the sensor perception module to handle different types of sensors.

References

- [Ame, 1980] American Society of Photogrammetry. *Manual of photogrammetry*, 4th ed., 1980.
- [Brendley and Grossman, 1992] K. W. Brendley and J. Grossman. War-game simulations challenged by advanced sensors. *Photonics Spectra*, June 1992.
- [Burdea and Coiffet, 1994] G. Burdea and P. Coiffet. *Virtual reality technology*. John Wiley and Sons, Inc., 1994.

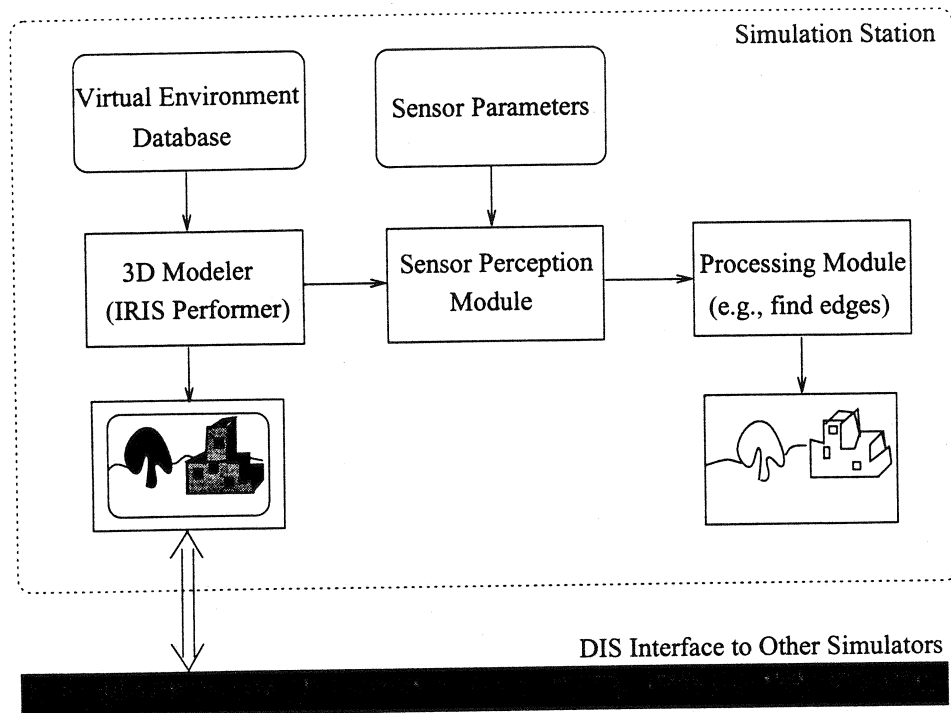


Figure 4: Sensing modules in the simulation station.

- [Chen *et al.*, 1994] C. Chen, M. M. Trivedi, and C. R. Bidlack. Simulation and animation of sensor-driven robots. *IEEE Trans. Robotics and Automation*, 10(5):pp. 684–704, October 1994.
- [Ins, 1994] Institute for Simulation and Training, University of Central Florida. *Enumeration and bit-encoded values for use with IEEE 1278.1 - 1994 DIS application protocols.*, 1994.
- [Kim *et al.*, 1994] Y. Kim, H. Ko, and B. Choe. Virtual reality infrastructure and its application to telerobotics. *Computer Graphics*, 18(5):667–673, 1994.
- [Macedonia *et al.*, 1994] M. R. Macedonia, M. J. Zyda, D. R. Pratt, P. T. Barham, and S. Zeswitz. NPSNET: a framework software architecture for large-scale virtual environments. *Presence: Teleoperation and Virtual Environments.*, 3(4):265–287, Fall 1994.
- [National Research Council, 1995] National Research Council. *Virtual reality, scientific and technological challenges.* National Academy Press, 1995.
- [Nav, 1995] Naval Postgraduate School. *NPSNET IV.7J*, 1995.
- [Sil, 1994] Silicon Graphics Computer Systems. *IRIS Performer: Programming Guide.*, 1994.
- [Wolff *et al.*, 1992] L. Wolff, S. Shafer, and G. Healey. *Radiometry—(Physics-Based Vision).* Springer-Verlag, 1992.

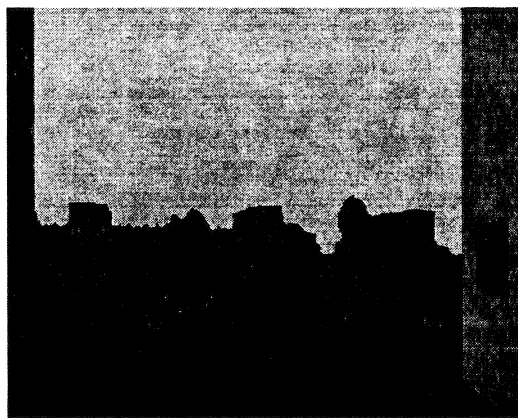
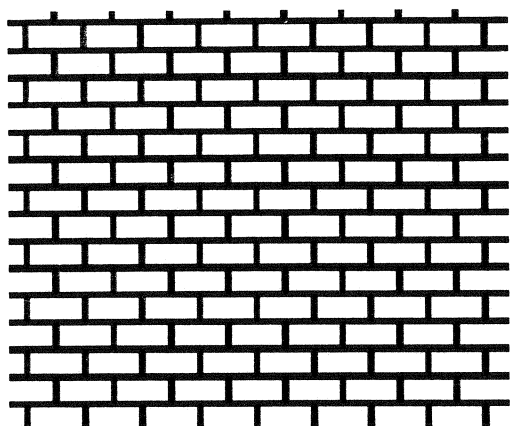


Figure 5: The generated images without radial distortion.

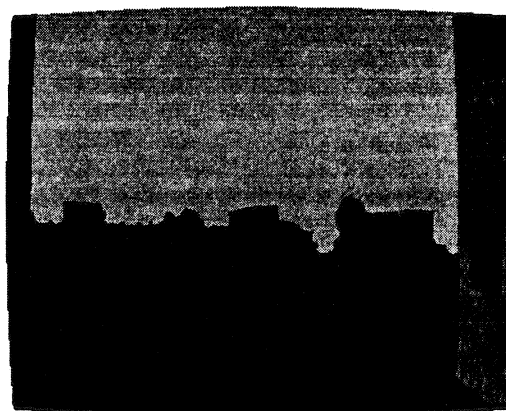
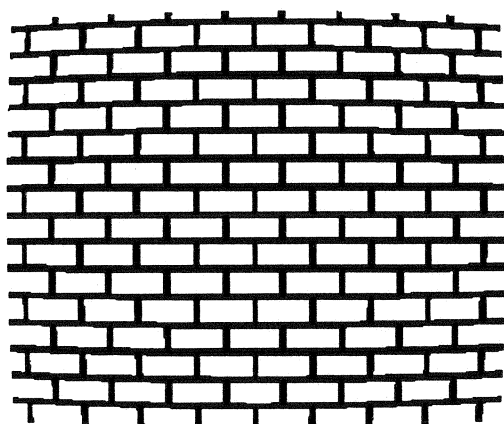


Figure 6: The effect of the radial distortion when $k=0.01$.

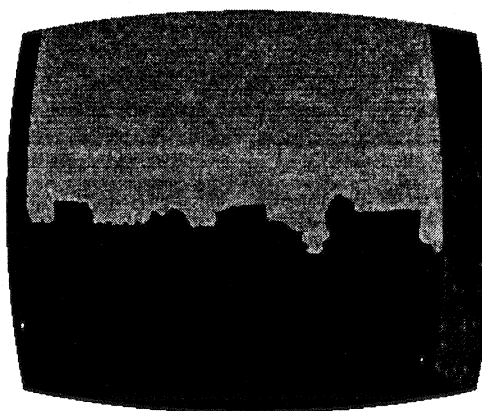
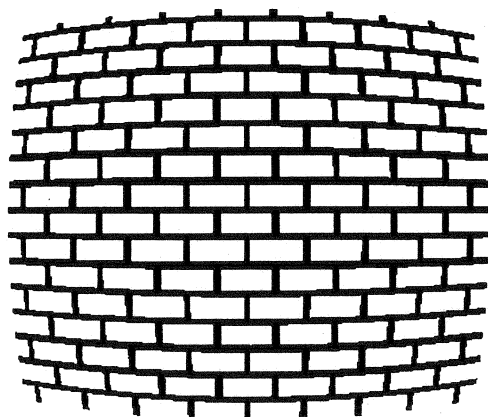


Figure 7: The effect of the radial distortion when $k=0.03$.