Yasuyoshi Yokokohji
Department of Mechanical Engineering
Kyoto University, Kyoto 606-8501, JAPAN
http://www.cs.cmu.edu/~msl/virtual_desc.html
yokokoji@mech.kyoto-u.ac.jp

Ralph L. Hollis
The Robotics Institute, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213

Takeo Kanade
The Robotics Institute, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213

WYSIWYF Display:

A Visual/Haptic Interface to Virtual Environment

Abstract

To build a VR training system for visuomotor skills, an image displayed by a visual interface should be correctly registered to a haptic interface so that the visual sensation and the haptic sensation are both spatially and temporally consistent. In other words, it is desirable that what you see is what you feel (WYSIWYF).

In this paper, we propose a method that can realize correct visual/haptic registration, namely WYSIWYF, by using a vision-based, object-tracking technique and a video-keying technique. Combining an encountered-type haptic device with a motion-command-type haptic rendering algorithm makes it possible to deal with two extreme cases (free motion and rigid constraint). This approach provides realistic haptic sensations, such as free-to-touch and move-and-collide. We describe a first prototype and illustrate its use with several demonstrations. The user encounters the haptic device exactly when his or her hand reaches a virtual object in the display. Although this prototype has some remaining technical problems to be solved, it serves well to show the validity of the proposed approach.

1 Introduction

Haptic interfaces have been recognized as important input/output channels to/from the virtual environment (Burdea, 1996). Usually a haptic interface is implemented with a visual display interface such as a head-mounted display (HMD) or a stereoscopic screen, and the total system is configured as a visual/haptic interface. Correct spatial registration of the visual interface with the haptic interface is not easy to achieve, however, and has not been seriously considered. For example, some systems have a graphics display simply located next to the haptic interface, resulting in a "feeling here but looking there" situation, as shown in Figure 1.

One of the most important potential applications of virtual reality (VR) systems is in training and simulation (Kozak, Hancock, Arthur, & Chrysler, 1993; NRC, 1995). Visual/haptic interfaces are expected to be useful for training of visuomotor skills such as medical operations, where visual stimuli and haptic stimuli are tightly coupled. Poor visual/haptic registration may cause an intersensory conflict leading to a wrong adaptation and a skewed sensory rearrangement (Rolland, Biocca, Barlow, & Kancherla, 1995; Groen & Werkhoven, 1998). The inconsistency between visual and haptic stimuli may render the training useless, or, in an even worse case, the training may negatively hurt the performance (negative skill transfer) in real situations.

Presence, Vol. 8, No. 4, August 1999, 412–434

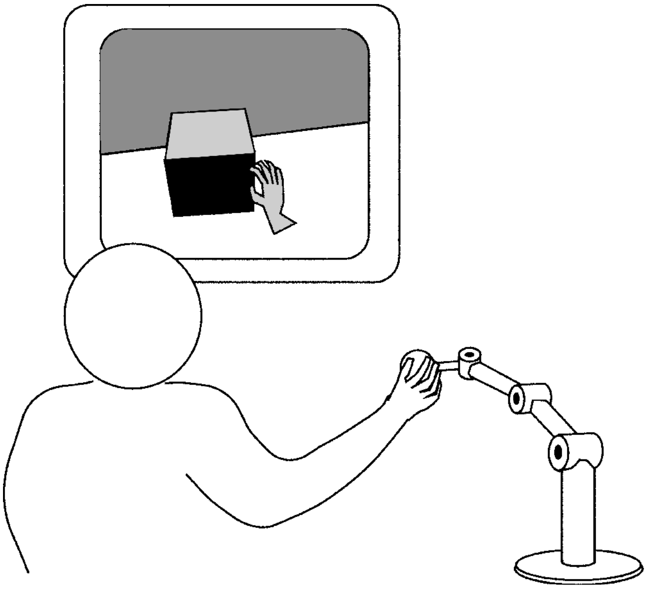


Figure 1. *"Feeling here but looking there" situation*

Ideally, a visual/haptic interface should be configured in such a way that "what you see is what you feel," as is the case in real life situations. Hereafter we refer to such an ideal situation as "WYSIWYF," in analogy with the term "WYSIWYG," or "what you see is what you get," commonly used in the context of document creation.

To date, several visual/haptic interfaces have been developed to realize the WYSIWYF, but some of them have forced the user to fix his or her head in a single position and others have had low spatial accuracy. This paper proposes a new method to realize WYSIWYF for visual/haptic interfaces, with a potential application area being the training of visuomotor skills. We refer to our first prototype as a "WYSIWYF display."

The authors believe that they first coined the term "WYSIWYF" (Yokokohji, Hollis, & Kanade, 1996a), but the concept of WYSIWYF itself is not novel. The contributions of this paper are to propose a reasonable way to realize WYSIWYF with sufficient accuracy to be useful, and to demonstrate the validity of the proposed method with the prototype system. Three key components of the proposed method are vision-based tracking, blending live video and computer graphics (CG) images by a chroma-keying technique, and introducing an encountered-type haptic interface. The encountered-type haptic interface controlled by a position-command-based haptic rendering algorithm can deal with two extreme cases: free motion and rigid constraint. The user's hand can "encounter" the haptic device exactly when/where his or her hand reaches the virtual object in the visual display.

The remainder of the paper is organized as follows. First the importance of WYSIWYF is discussed in Section 2. A reasonable method to realize WYSIWYF is presented in Section 3. A prototype WYSIWYF display and some demonstrations are shown in Section 4. Performance of the prototype system is evaluated in Section 5. Finally, conclusions are given in Section 6.

2 What is the Importance of WYSIWYF?

It would perhaps seem obvious that the ideal configuration of visual/haptic interfaces is WYSIWYF, matching the situation in real life. For example, if you reach out to manipulate an object on your desk, you feel the object in precisely the same place that you see it. In the case of a VR system, one would wish to "reach into" a graphical display with one's hand to manipulate an object in the same way. It is worth discussing, however, why WYSIWYF is actually important, especially when a WYSIWYF system might be expensive to realize. For example, if experience shows that the training performance for a given task on a non-WYSIWYF training system is transferred without difficulty to the target real-life situation, then WYSIWYF is evidently not an important requirement for that domain.

It is well known that even when the visual system is distorted by wearing a pair of prism glasses and becomes inconsistent with the body coordinate system, the visual system is still dominant (visual capture) (Kornheiser, 1976; Welch & Warren, 1980). This inconsistency causes a wrong adaptation that remains even after the distortion is removed (negative aftereffect) (Rolland et al., 1995; Groen & Werkhoven, 1998). Therefore, if the motion in the target task is closely related to the body coordinates (e.g., tasks such as reaching or catching an object), the training system should be WYSIWYF. Otherwise, the training effort might turn out to be useless.

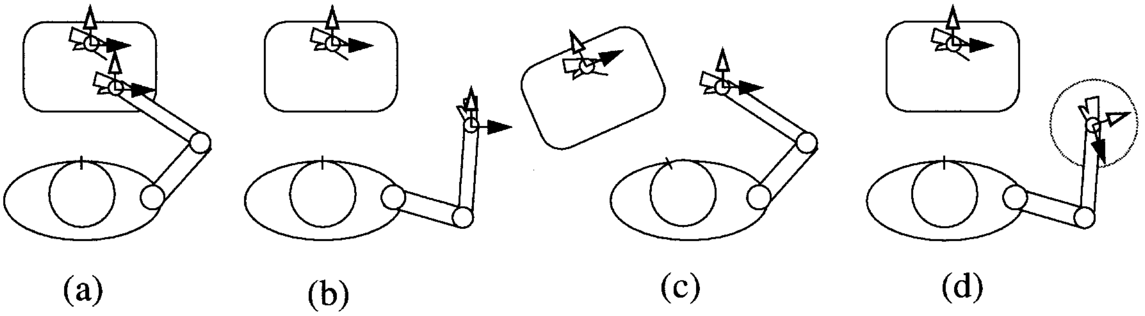


Figure 2. *Consistent and inconsistent layouts of visual/haptic display*

In the above examples, the importance of WYSIWYF depends on whether the motion is associated with the body coordinates or the task coordinates. In other cases, local motions with forearm and fingers tend to dominate, such as in medical operations and handicraft work. In these cases, it is not clear whether the motion is associated with the body coordinates or the task coordinates. To answer this question, Hammerton and Tickner (1964) showed an interesting experimental result. They investigated whether a finger motion that was trained with visual guidance is space oriented (i.e., relative to the CRT display) or body oriented (i.e., relative to the subject's forearm). Results showed the training was transferred better in a body-oriented situation than in a space-oriented one. Consequently, they concluded that the trained finger motion was associated with the body coordinates and not with the task coordinates.

Considering Hammerton and Tickner's result, we can consider the importance of WYSIWYF. As shown in Figure 2, suppose a trainee is manipulating a haptic device while watching a visual display. (For simplification, no haptic device is drawn in the figure.) Arrows indicate the corresponding directions between the visual display and the haptic display. The situation (*a*) is WYSIWYF and is considered ideal for training. If (*a*) cannot be realized for some reason, then (*c*) is better for training than (*b*), because (*c*) keeps the consistency with respect to the body coordinates whereas (*b*) does not. Even though (*d*) is the same configuration as (*b*), it is better than (*b*) for local motions within the circle, because, as long as the motion is within this circle, (*d*) is consistent to the ideal case. This agrees with our common-sense observation, e.g., that we can write letters more comfortably in situation (*d*) than we can in situation (*b*).

Related to WYSIWYF, Spragg, Finck, and Smith (1959) studied tracking performance as a function of the control-display movement relationships, such as vertical-horizontal and vertical-vertical. Bernotat (1970) investigated the influence of a relative rotation between the joystick reference system and the display reference system. Norris and Spragg (1953) compared the performance of a tracking task by two-hand coordination in various display/controller configurations. Pichler, Radermacher, Boeckmann, Rau, & Jakse (1997) compared the performance of endoscopic surgery between compatible and incompatible arrangements for eye-hand coordination. In all cases, "natural," "compatible," or "expected" arrangements gave better performance than other "unnatural," "incompatible," or "ambiguous" situations.

Groen and Werkhoven (1998) compared manipulation performance between the case in which a virtual hand is laterally shifted and the case using an aligned virtual hand. Subjects were asked to grasp virtual objects, to rotate them, and to place them to a certain position by using a virtual hand, which is laterally shifted or aligned, looking through a fixed HMD. They measured completion times and positioning accuracy in two cases, but did not find any significant difference in performance. This does not mean, however, that WYSIWYF is unimportant. First, as the authors pointed out, both cases had depth distortion (due to the HMD they used), and this distortion might have dominated the effects of lateral misalignments. Second, there was no haptic sensation provided in their experiment. Finally, equal performance does not directly mean that the effects on training are equal, a fact that is well known as the "learning versus performance" distinction (Schmidt, 1988).

In all likelihood, WYSIWYF is preferable for training, but just how important WYSIWYF is for the effectiveness of training may depend greatly on the nature of the target tasks.[1] Certainly, human beings have a high capability to adapt to different arrangements of the visual and haptic coordinate frames. It is questionable, however, that we should expect such a capability to kick in even in unexpected panic situations. (Imagine a non-WYSIWYF emergency medical operation.) In conclusion, the importance of WYSIWYF may depend on the target domain, and it should be examined experimentally. We leave the experimental study comparing the effectiveness of training between WYSIWYF and non-WYSIWYF environments as an important future topic.

3 How Can WYSIWYF Be Realized?

Figure 3 conceptually illustrates the proposed method to realize WYSIWYF. Three key components of the proposed method are vision-based tracking, blending live video and CG by the chroma-key technique, and introduction of the encountered-type haptic interface. Individual components are explained below.

3.1 Vision-Based Visual/Haptic Registration

As shown in Figure 3, the user wears a head-mounted display (HMD), on which a pair of stereo cameras is attached. A haptic device is placed at an appropriate location and some fiducial points are attached to the haptic device. The user's head motion is estimated by tracking the fiducial points on the haptic device. Knowing the precise position/orientation of the user's head makes it possible to overlay a virtual-object image on the real image of the haptic device, regardless of the position of the user's head. This is the same idea used in augmented-reality applications, in which a supplemental image is overlaid on the real-object image. In the case of the WYSIWYF display, however, no real-world image other than the image of the user's hand is displayed to the user.

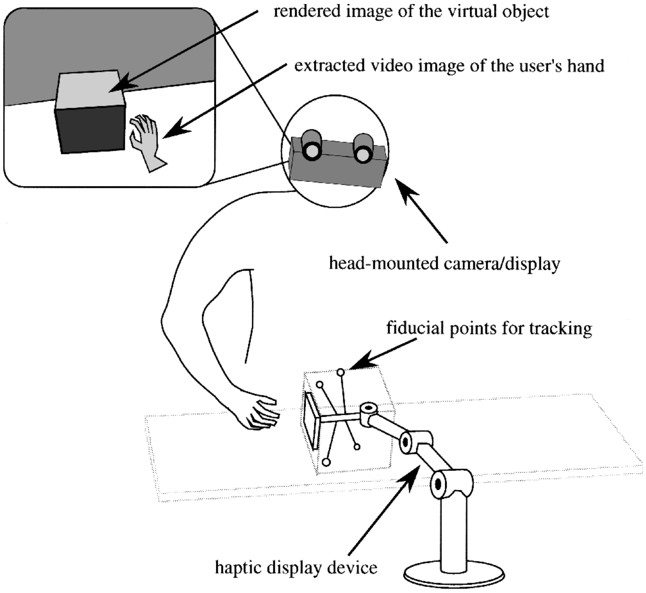

rendered image of the virtual object
extracted video image of the user's hand
head-mounted camera/display
fiducial points for tracking
haptic display device

Figure 3. *WYSIWYF display*

The most popular way to measure head motion is to use magnetic trackers, but a vision- or optical-based approach is another option (Bajura & Neumann, 1995; Bishop, 1984; Uenohara & Kanade, 1995; Ward, Azuma, Bennett, Gottschalk & Fuchs, 1992). Unlike magnetic-based sensors, whose accuracy is affected by nearby ferromagnetic materials and other magnetic field sources and depends on the distance from the emitter, a vision-based approach can be spatially homogeneous by appropriately placing the set of fiducial points. Vision-based sensing is generally more accurate than magnetic-based sensing (State, Hirota, Chen, Garrett, & Livingston, 1996). Its accuracy can be up to the level of a pixel in the display coordinates. Although the registration accuracy in the depth direction is worse than in other directions, the depth error does not contribute much to the final alignment error in the image. Therefore, it seems reasonable to use the target image, on which the

[1] In some cases, non-WYSIWYF training might be more effective than WYSIWYF, especially at the initial stage of training. However, this observation is beyond the scope of this paper, and we assume that WYSIWYF is generally better than non-WYSIWYF.

supplemental image is overlaid, for vision-based tracking. Whenever 2-D alignment is discussed, registration generally refers to the degree of overlay in the image plane. A certain depth accuracy is necessary, however, to enable the user to reach the virtual object without too much adjustment. Precise alignments using stereo cameras will improve the depth accuracy and make 2-D alignment correspond more closely to 3-D registration.

Unfortunately, the vision-based approach also has several drawbacks. First, the fiducial points must be visible from the camera, otherwise occlusions may result in an incorrect estimation. When an iterative estimation assuming temporal coherence is used, a quick motion may cause the tracker to miss all of the fiducial points and lose itself completely. Therefore, for accurate and robust tracking, complementary use of other sensors, such as magnetic trackers, will be required (State et al., 1996).

To recover the 3-D pose from a single camera image, at least three fiducial points are necessary. (The actual number of fiducial points should be more than three to cope with occlusions.) The fiducial points can be put either on the endpoint of the haptic device or on the base of the working environment. In Figure 3, the fiducial points are put at the endpoint of the device, because it is likely that the camera on the HMD will always capture the haptic device, whereas fiducial points on the fixed base may be out of camera sight or may be occluded by the haptic device. An iterative pose estimation with least-squares minimization (Lowe, 1992) is a simple method to use, but it is sensitive to noise. In our prototype, which will be introduced in Section 4, a simple least-squares minimization yielded some jitters in the overlaid image. To smooth out the tracking noise, we implemented the extended Kalman filter (Gennery, 1992).

3.2 Extracting the User's Hand Image and Blending with CG

The camera image captured for tracking the fiducial points should also include the user's hand. If one can extract the portion of the user's hand from the captured image, this extracted image can be superimposed on the CG image of the virtual environment. Displaying the user's actual hand image in the CG scene provides a marked improvement over the usual practice of rendering a synthetic polygonal hand image. The easiest way to do so is by "chroma-key," which enables us to extract images from a uniformly colored (usually blue) background.

Video keying has been used to include a virtual image in a real image, such as the "luna-key"-based superimposition of the medical ultrasound image on the patient body by Bajura, Fuchs, and Ohbuchi (1992). Metzger (1993) proposed to use chroma-key for mixed reality, i.e., not only "virtual in real" but also "real in virtual." The keying method in this paper corresponds to "real in virtual" and is likely the first one to be applied to visual/haptic interfaces.

Using the live user's hand image allows us to eliminate a sensing device, like a data glove. Wearing a data glove would be a burden for the user, and adding a sensing device would make the system more complex. Especially, synchronizing the information from multiple sensors is not easy (Jacobs, Livingston, & State, 1997). With the proposed method using chroma-key, on the other hand, there is no synchronization problem because a single camera image is used both for tracking the fiducial points and for extracting the user's hand image.

One difficulty of the chroma-key method is that it cannot a priori realize the correct spatial relationship between the user's hand and the virtual object. That is, even when the user's hand is behind a virtual object, the hand image is simply "pasted" on top of the virtual scene as long as the camera captures it in the real scene.[2] To solve this problem, keying should really be based on depth information instead of color information. Kanade, Kano, Kimura, Yoshida, and Oda (1995) developed a real-time stereo machine and applied it to perform "Z-keying" (Kanade, Yoshida, Oda, Kano, & Tanaka, 1996), merging a real scene into a virtual scene in a spatially consistent manner. If one can get an accurate depth map in real-time in the future, chroma-key will be replaced by this Z-key technique.

[2] As we will see later, sometimes the user's hand is consistently occluded in the final blended image, when the hand is actually occluded by a real object that has the same shape as the virtual object.

3.3 Encountered-Type Haptic Display

Most of haptic devices developed so far are either worn, such as exoskeleton masters (Bergamasco, Allota, Bosio, Ferretti, Parrini, Prisco, Salsedo, & Sartini, 1994) or held, such as universal hand controllers (Iwata, 1990). In both types, physical contact between the device and the user's hand is maintained all the time. With the encountered-type approach, which was proposed first by McNeely (1993), the device stays at the location of the virtual object and waits for the user to "encounter" it. Tachi, Maeda, Hirata & Hoshino (1994) independently proposed a similar idea. Figure 3, for example, shows the user about to touch a face of the virtual cube, and the device is already positioned at the location of that face. To do this, the system must predict where the user is going to reach, beforehand. With this method, the user need not wear any haptic devices. When the user's hand is free in the virtual environment, it is completely unencumbered.

Another method that allows the user's hand to be free is to measure the distance between the device and the user's finger in a noncontact manner and let the device follow the user's motion (Hirota & Hirose, 1993; Luecke & Winkler, 1994; Yoshikawa & Nagura, 1997). Let us refer to this approach as a "noncontact tracking type." With the noncontact tracking type, the haptic device is used to track the user's hand, so it is unlikely that the device will collide with the user unexpectedly. In the case of an encountered type, however, one has to implement a tracking function separately, and the device can potentially collide with the user. A drawback of the noncontact tracking type is that the device has to keep tracking even when the user's hand happens to be far from the virtual object, and the user cannot move his or her hand beyond the working volume of the device.

A serious problem of the encountered-type approach is that the device cannot display an arbitrary shaped surface with the currently available technologies. Tachi, Maeda, Hirata, and Hoshino (1994, 1995) tried to display arbitrary surfaces by the shape-approximation technique, but currently the user is allowed to touch the surface with only a point contact. Hirota and Hirose (1995) developed a surface display with a distributed array of prismatic actuators, but its resolution is not enough. Another criticism against this approach is that it cannot display a large smooth surface such as a tabletop. To display such large areas, one could prepare just a small patch of the surface and attach it to the display endpoint. By tracking the user's motion, it would be possible to always locate the patch beneath the user's hand while reacting to the force exerted by the user in the surface normal direction. This would be a kind of hybrid tracking/rendering control (Tachi et al., 1995). Although the user cannot get any sense of slip with this approach, it is difficult for any other haptic devices to simulate such a situation. To add realism, a small vibration could be used for rendering the surface texture (Minsky, Ohu-young, Steele, Brooks & Behensky, 1990). At present, we would judge that the encountered-type approach is probably most appropriate when the size and shape of the objects are limited (e.g., a few simple tools, or switches and knobs such as those found on control panels).

For our encountered-type haptic interface, we chose to implement a physically based dynamic simulation algorithm for nonpenetrating rigid bodies (Baraff, 1994). Appendix A shows the basic idea of Baraff's algorithm, which can calculate constraint forces and collision impulses between rigid bodies without any penetrations, making it suitable for realistic haptic rendering of rigid contacts.

As shown in Appendix B, haptic rendering algorithms can be classified as force-command types (or impedance approaches) and motion-command types (or admittance approaches) (Yoshikawa, Yokokohji, Matsumoto, & Zheng, 1995). Appendix B shows that neither type can deal with either of two extreme situations: free motion and rigid constraint. Since Baraff's approach is based on solving ordinary differential equations (ODEs), it is of the motion-command type and cannot deal with free motion. Nevertheless, this problem can be solved by introducing the encountered-type haptic approach. With the encountered-type approach, the user is unconnected with the haptic device when in free space in the virtual environment; therefore, a completely free situation is already realized, and the device itself need not display such a situation. Before the user encounters the device, the device has to stay at a certain location, and its control mode should be of the position-command type. If we adopt the force-command type for haptic rendering, we have to change the control mode after the user encounters the device. The motion-command type is also advantageous in this sense, because it makes the device control seamless before and after the encounter.

3.4 Other Visual/Haptic Interface Systems

As mentioned in Section 1, the concept of WYSIWYF is itself not novel, and there are some systems aiming at this concept. For example, Iwata (1990) developed a desktop-type haptic device. They put a mirror between the haptic device and the user in such a way that the CG image is registered with the haptic device. Sato, Hirata and Kawarada (1992) developed a string-driven haptic device combining with a stereoscopic display. Since the haptic device is string based, the graphic display can be placed simply behind the device so that a 3-D image pops up at the location of the device. In those cases, a careful setup is required, and the user must fix his or her head position to see a well-registered image. Deering (1992) introduced several compensation techniques in order to realize correct disparity and motion parallax with a stereoscopic screen. He demonstrated that a virtual tool could be accurately registered to a real 3-D mouse; however, no haptic device was used.

The virtual control panel by Gruenbaum, Overman, Knutson, McNeely, and Sowizral (1995) and Gruenbaum, McNeely, Sowizral, Overman, and Knutson (1997) and the virtual haptic space system by Tachi et al. (1994, 1995) adopted the encountered-type approach, and they used an HMD to realize WYSIWYF. Both systems used a magnetic sensor for head tracking, and the accuracy was insufficient for correct WYSIWYF. Both systems used a synthetic polygonal hand image.

For correct WYSIWYF, the visual/haptic registration must be spatially and temporally accurate, but with the approaches cited it is difficult to get the accuracy required. The method proposed in this paper is a combination of vision-based tracking, chroma-key, and an encountered-type haptic interface. Although each of these elements by themselves are existing techniques, we propose that, through a novel and serendipitous combination of these techniques, correct WYSIWYF is achievable.

We summarize the contributions of the paper as follows.

- For correct WYSIWYF operation, accurate visual/haptic registration must be realized both spatially and temporally. We show that vision-based tracking is a good way to get the required accuracy of 3-D registration as well as 2-D alignment.
- We show that the real user's hand image extracted by chroma-key can replace the traditional polygonal hand image, which requires a sensing device (like a data glove). Since the hand image is extracted from the same camera image used for the fiducial point tracking, there is no synchronization problem.
- We show that the combination of the encountered-type haptic device with the motion-command type haptic rendering algorithm can deal properly with the two extreme situations: free motion and rigid constraint.

Figure 4 shows the overall registration/blending process of the proposed method.

4 Prototype WYSIWYF Display

4.1 System Configuration

Figure 5 illustrates the prototype system configuration. Figure 6 shows a system overview in use. In the following sections, we explain the details of the prototype system. Configuring the prototype system, we decided to use some existing devices, such as an LCD panel and a PUMA robot. Note that this configuration is not the optimal solution but just a first step in examining the validity of the proposed approach.

4.1.1 Visual Display Part. Although a head-mounted camera/display would be ideal for WYSIWYF as shown in Figure 3, we decided to use an existing LCD panel (10 in. TFT color) for the first prototype. A color CCD camera (lens focal length 6 mm, FOV 56.14 deg. × 43.60 deg.) was attached to the back plane of the LCD panel. The LCD/camera system is mounted on a

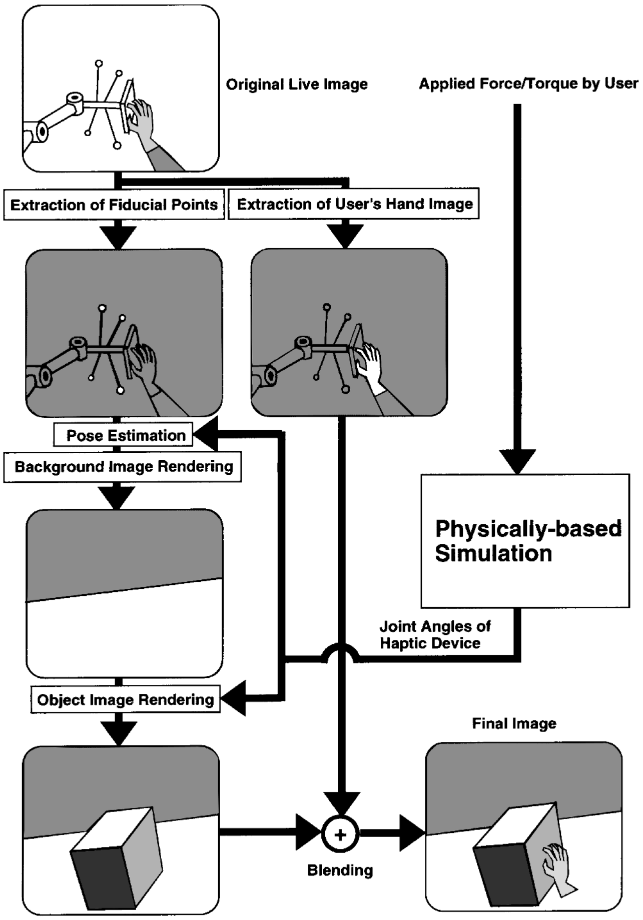

Original Live Image
Applied Force/Torque by User
Extraction of Fiducial Points
Extraction of User's Hand Image
Pose Estimation
Background Image Rendering
Physically-based Simulation
Joint Angles of Haptic Device
Object Image Rendering
Final Image
+
Blending

Figure 4. *Flow of the registration/blending process*

movable stand so that the user can move it around to change his or her viewpoint. In this configuration, the LCD panel becomes a virtual window through which the user can see a virtual world.[3]

There are several ways to represent the orientation of the camera, such as Euler angles and roll-pitch-yaw angles. However, these representations have singularities where the orientation cannot be expressed uniquely. Since the camera orientation may change in a wide range, we introduced quaternions to represent the orientation. This is a redundant representation but is singularity free. To recover the camera pose from the fiducial points in image coordinates, we used the extended Kalman filter by Gennery (1992), where the quaternion representation is used.

The virtual environment is rendered by an SGI PowerOnyx (MIPS R8000 × 2) with OpenGL application programming interface (API). An optional SIRIUS video board, which has built-in video-keying circuitry, is installed in the PowerOnyx. In the first prototype, the vision-based tracking function was implemented on the PowerOnyx (Yokokohji, Hollis, & Kanade, 1996a). We first tried to input the video image into the main memory through the SIRIUS circuitry. A somewhat disappointing design specification of the SIRIUS board, however, prevents us from using the video-keying circuitry while the video-to-memory path is being used. We gave up using the hardware keying circuitry and implemented a software chroma-key instead, but it ended up with a very low frame rate (3 Hz) and a large latency (0.9 sec). To avoid this annoying low frame rate and large latency, we reluctantly introduced "camera-fixed mode," in which the vision-based tracking feature is used only for the initial registration. Once the initial registration is completed, the vision-based tracking is disabled so that the built-in circuitry can be used for chroma-key. More details about the performance evaluation of the prototype system will be discussed in Section 4.4.

4.1.2 Haptic Display Part.

A Unimation PUMA 560 6-DOF industrial robot is used for the haptic device. A JR^3 six-axis force/torque sensor is attached to the tool flange of the PUMA. We then attached an aluminum plate with four fiducial points—small incandescent lamps covered by translucent lenses. The working environment including the PUMA was covered by blue cloth.

The physically based simulation runs on a VME-bus CPU board (25 MHz Motorola MVME162-23 MC68040) under the VxWorks real-time operating system. RCCL/RCI (Lloyd & Hayward, 1992), real-time C libraries for controlling PUMAs, was installed on the VxWorks system. VAL, the original software system of the Unimation controller, was replaced by "moper," a special control/communication program for RCCL. The Unimation controller receives the joint trajectories

[3] It is not a perfect virtual window in a sense that once the camera/display system is fixed, the view direction of the virtual image does not change, even if the user changes his or her head location.

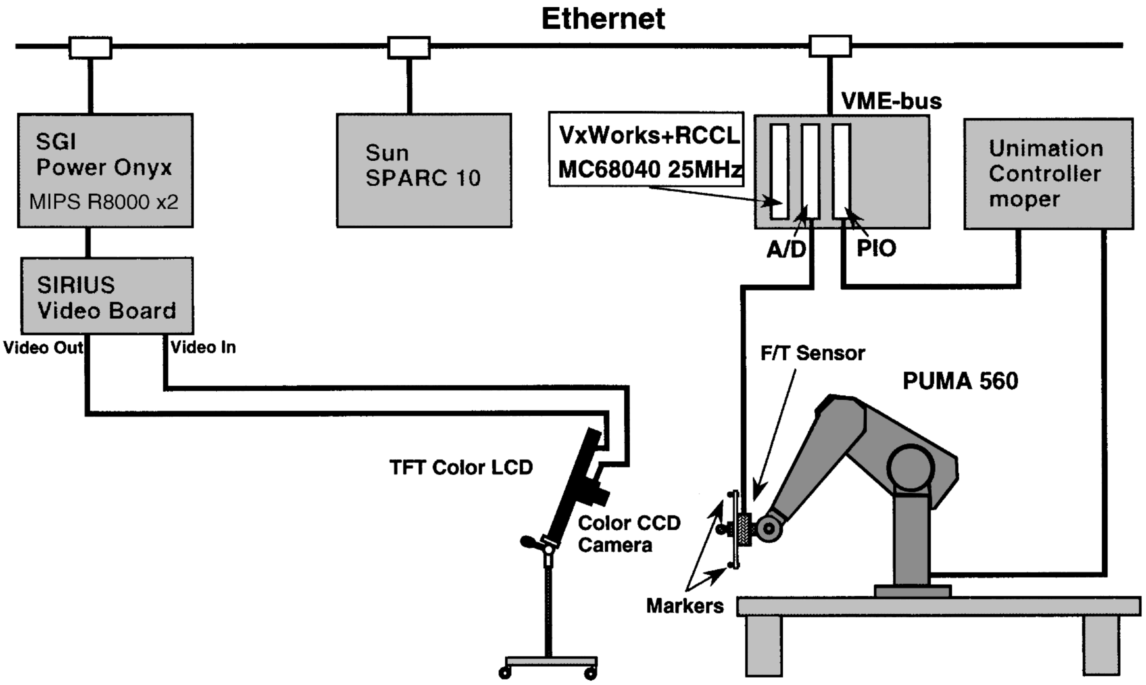


Figure 5. *Prototype system configuration*

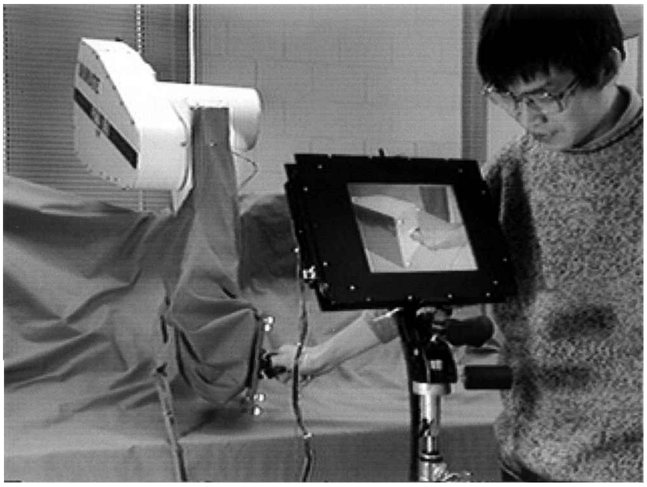


Figure 6. *System overview in use*

from the RCCL routine running on the VxWorks system through a parallel I/O board, while six independent servo modules in the controller handle the position servo of each joint.

Basically, programs in the force display part and the visual display part are running asynchronously. The control cycle of the haptic display is 20 msec. In each control cycle, the physically based simulation routinely checks static contacts and dynamic collisions between the objects (details are shown in Appendix A.3) and obtains a desired position/orientation of the device endpoint. The endpoint data is translated into joint angle data and sent to the Unimation controller. Moper then interpolates the received joint data and each servo module controls each joint at 1,000 Hz.

To render the virtual environment, the PowerOnyx must know the haptic device endpoint position/orientation. In the prototype system, a CG-rendering routine in the PowerOnyx sends a request for the current endpoint information to the VxWorks system via ethernet by a socket communication protocol. In the VxWorks system, a high-priority process receives this request and sends the data back to the PowerOnyx.

4.2 Some Demonstrations

Several demonstrations of our developed system are shown in this section. The demonstrations were all conducted in the camera-fixed mode.

4.2.1 Cube.

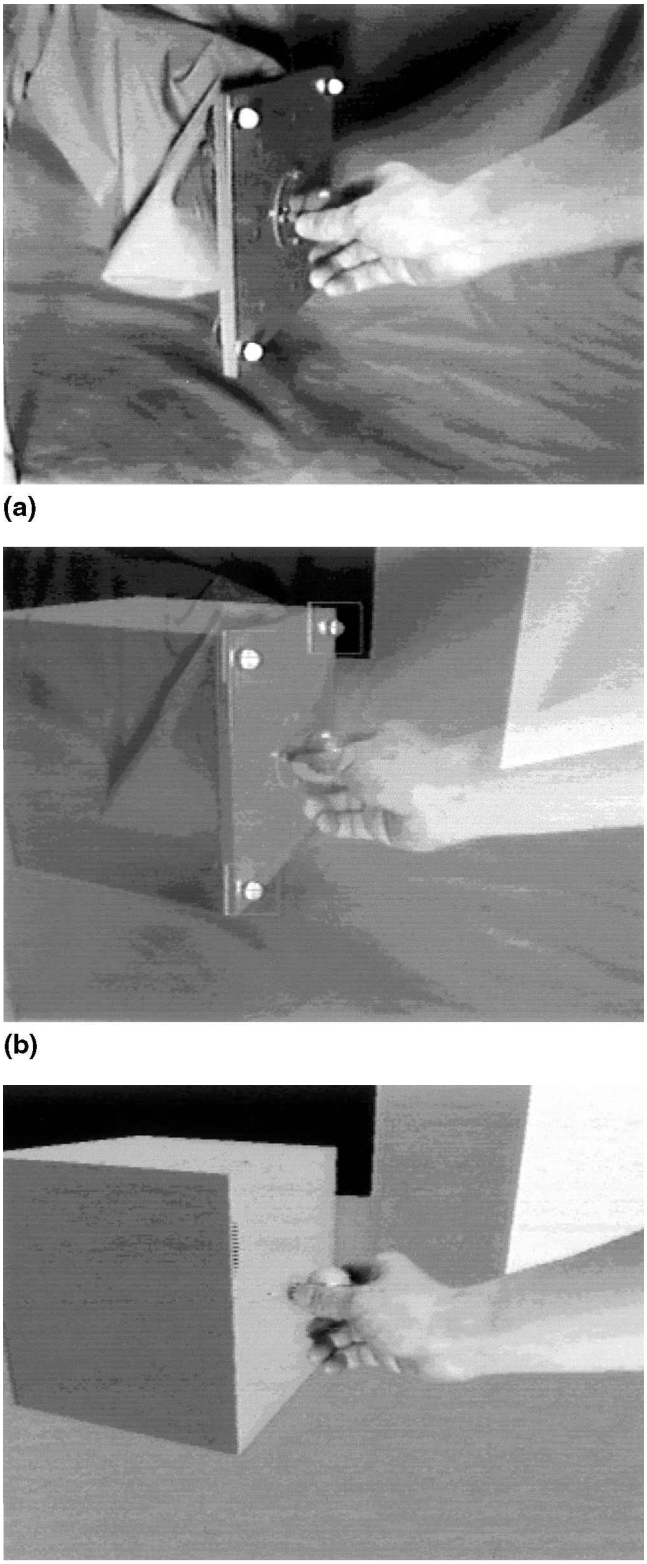
A simple frictionless virtual environment was built, in which a cube with edges of 20 cm is placed on top of a flat table. The user can manipulate the cube in space and bring it into contact with the table. Figure 7 shows the video-blending process. The overlaid image in Figure 7(b), which is not actually shown to the user, demonstrates how well the virtual cube is registered to the real fiducial plate. Figure 8 shows a sequence of manipulating the cube. Note that these are not static scenes (the user moves the cube around at approximately 30 cm/s), but the synthetic image (cube) is well registered to the real image (user's hand).

In this example, a knob at the device endpoint is the only part that the user is allowed to touch. (see Figure 7(a).) In the virtual environment, this part corresponds to a virtual knob attached to the virtual cube. (Figure 7(c).) In Figure 7(c) and Figure 8, the user's hand is consistently occluded by the knob in the virtual environment, because it is occluded by the real knob.[4] The mass of the virtual cube was set at 10 kg, and gravity was 0.01 that of normal. If we try to simulate an object lighter than 10 kg, the system response becomes oscillatory and one cannot continue the operation stably. The oscillation occurs when we try to cancel too much of the PUMA's inertia with a relatively slow sampling period (20 ms). With the above stable parameter setting, however, the user can get a convincing haptic sensation. Although it is quite subjective, the user can clearly distinguish contact states between the cube and the table, such as vertex contact, edge contact, and face contact—even without visual information.

As long as we are in camera-fixed mode, the prototype system provides a situation that is quite realistic. For example, the user can touch the haptic device exactly when his or her hand reaches the virtual cube in the display, and can feel the reaction forces exactly when the cube hits the table in the display.

4.2.2 Virtual Tennis.

Figure 9 illustrates "virtual tennis." Here, a virtual ball hangs by a virtual string. The ball diameter is 7 cm, and the string length is 20 cm. The mass of the ball was set to 3 kg, and the gravity


(a)

(b)

(c)

Figure 7. *Results of registration and blending: (a) original video scene; (b) overlaid image; (c) final blended image*

[4] This is an example of correctly rendered occlusion discussed in the previous footnote.

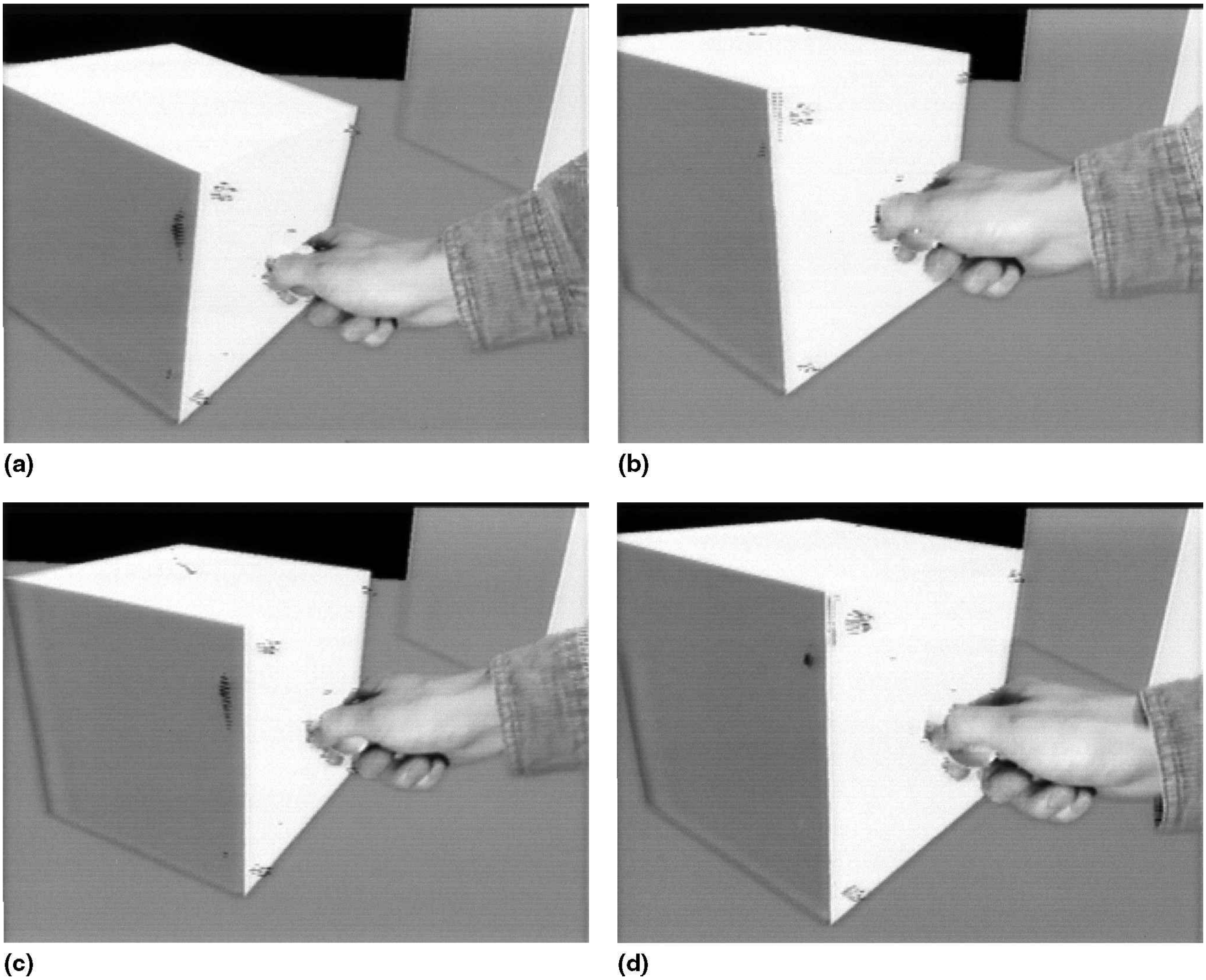


Figure 8. *A sequence of cube manipulation*

is 0.05 that of normal. Note that, in this case, the ball is a virtual image, but the racket is a real image. This example demonstrates that the user can interact with virtual objects not only with his or her own hand but also by using real tools.

4.2.3 Training. A potential application of this system is the training of visuomotor skills. The basic idea of skill training is referred to as a ‘‘record-and-replay’’ strategy (Yokokohji, Hollis, Kanade, Henmi, & Yoshikawa, 1996c). First, an expert demonstrates his or her skill with the WYSIWYF display, and all available data is recorded. Then a trainee subsequently learns the skill by replaying the data with the WYSIWYF display.

Figure 10 shows a simple example of training. The user is trying to follow a prerecorded motion demonstrated by an expert, which is shown by a translucent cube. A position servo can guide the user to the reference motion. Unlike just watching a video, the trainee can feel the reaction forces from the virtual environ-ment while trying to follow the reference motion. The servo gain can be adjusted according to the trainee’s progress; for example, starting from a high gain and adjusting it to a lower gain as the trainee’s performance improves. For more discussion about training, see (Yoshikawa & Henmi, 1996; Yokokohji et al.,

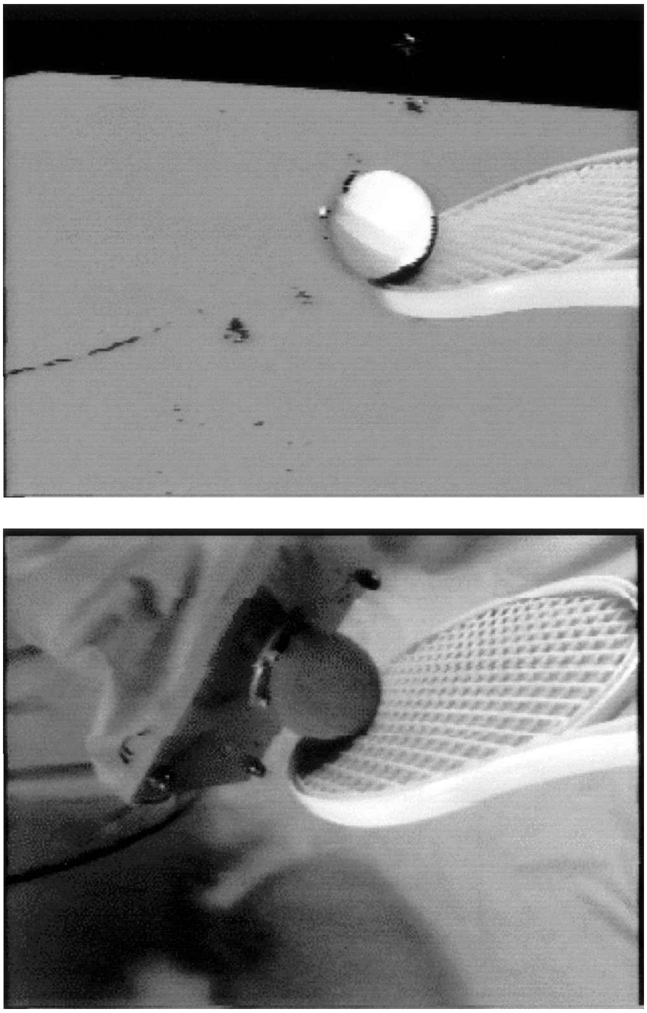


Figure 9. *Virtual tennis: (top) what the user can see; (bottom) what the user is actually doing.*

1996c; Henmi & Yoshikawa, 1998; Yokokohji, 1998).

4.2.4 Handling Multiple Tools. As discussed in Section 3.3, our WYSIWYF display adopts the encountered-type haptic display approach. In the previous examples, the user manipulated only one virtual object (a cube or a ball). In such cases, the haptic device can simply stay at the location of the virtual object. Although the user can get a realistic touch feeling when he or she encounters the object, there is no difference between the encountered type and the held type as long as the user keeps holding the haptic device.

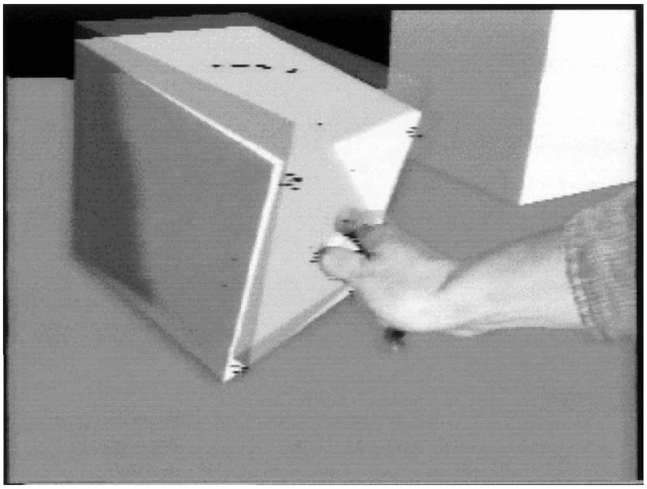


Figure 10. *An example of skill training*

To demonstrate the encountered-type approach effectively, we need a situation in which the user repeats his or her encounter with the object, like the virtual tennis example, or handles multiple tools and frequently exchanges one with another. Suppose that there are more than two virtual objects to be encountered. In such a case, the system must track the user's hand motion in some way, predict which object the user's hand is going to reach for, and quickly move the device to the predicted object location before it is contacted.

Figure 11 shows an example of such cases. Here, there are two virtual tools sticking in a piece of "virtual cheese." When the user decides to change the tool, the haptic device changes its location so that he or she can encounter the selected one. In this example, the user manually selects the tool with a toggle switch using his or her left hand, and no tracking/prediction mechanism is implemented. When the selected tool is ready to be encountered, its color changes from red (dark shading in the figures) to green (light shading). Of course, a tracking/prediction mechanism needs to be implemented to claim the advantage of the encountered-type approach. Therefore, this example merely illustrates the potential of simulating multiple objects with a single haptic device. A possible application would be to simulate a task having several tools with the same grip shape, which would be a first step toward training for medical procedures.

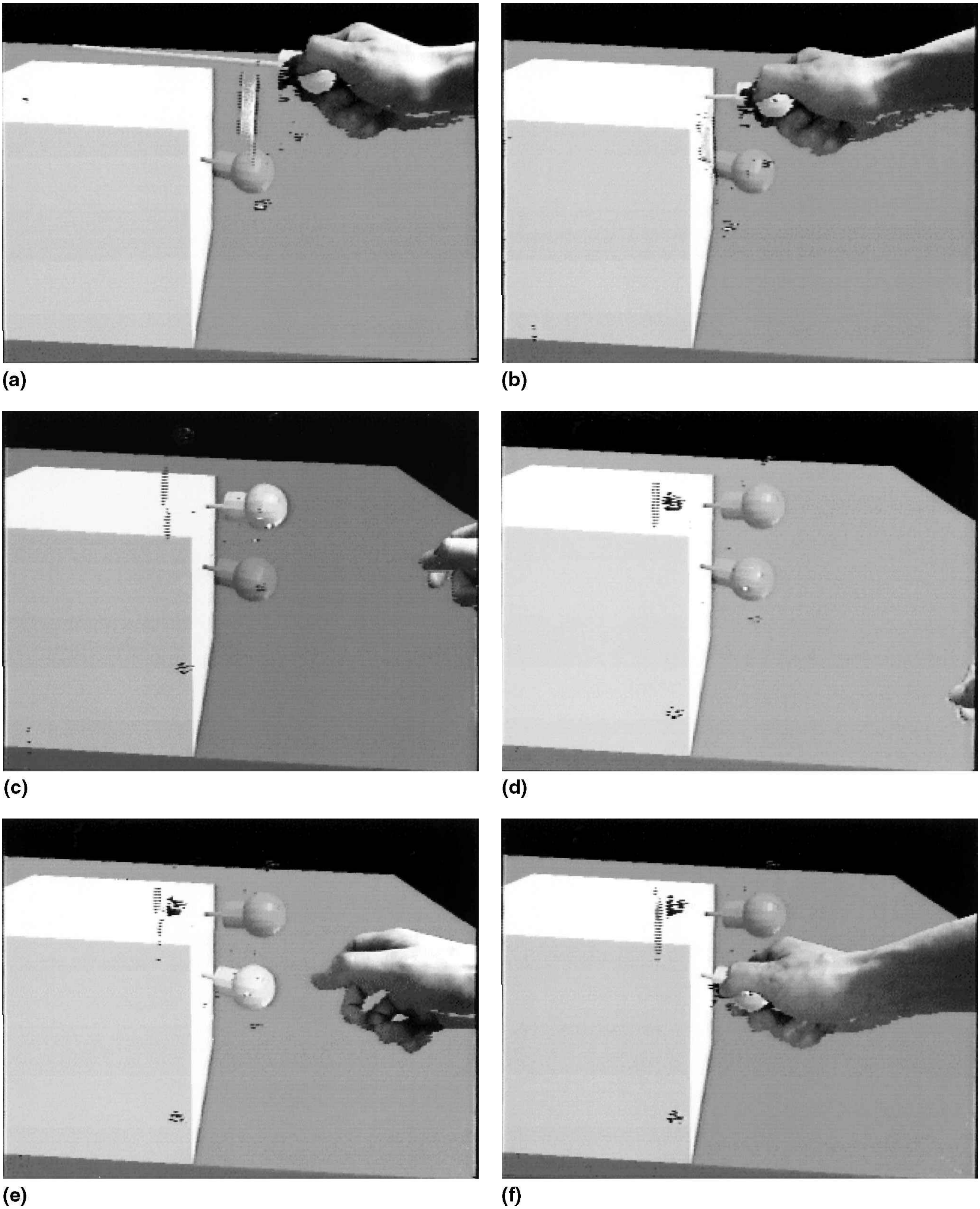


Figure 11. *Handling multiple tools: (a) User grasps virtual tool A, physically simulated by the haptic device; (b) User cuts "virtual cheese"; (c) User releases tool A; (d) User toggles the active tool, causing the haptic device to shift from the tool A location to the tool B location; (e) User grasps virtual tool B, physically simulated by the haptic device; (f) User continues interactions using tool B.*

There are several ways to track the user's hand motion, such as tracking an infrared LED attached to the finger by two cameras (Gruenbaum et al., 1995, 1997) and using a passive link mechanism (Tachi et al., 1994; Tachi et al., 1995). Using color information and a statistical model of hand motion may make the tracking more robust and reliable (Wren, Azarbayejani, Darrell, & Pentland, 1997). In some applications, like the virtual control panel by Gruenbaum et al. (1995, 1997), it is enough to track one reference point of the user's hand in order to determine the location of the device. In addition, since the hand tracking is used just for predicting the location the user is going to reach for, the tracking accuracy and delay may not affect the system performance directly. Once the reaching location is predicted, the device can use its own accurate joint sensors to position itself to that location. Consequently, hand tracking for the encountered-type haptic display approach can be much simpler than rendering a polygonal hand, for which one needs to know finger joint angles. Therefore, it is still reasonable to use the chroma-key technique even when the user's hand motion must be tracked.

4.3 Some Attempts to Improve the Performance

4.3.1 Implementing a Fast Video Tracker.

As already mentioned, we first implemented the vision-based tracking function in the PowerOnyx. Due to the limited performance of the SIRIUS video board, however, we reluctantly introduced the camera-fixed mode. To allow the user to move the camera/display system at any time during the interaction, we must implement a tracking function that does not depend on the SIRIUS board. For this purpose, we introduced Tracking Vision, a motion tracking system made by Fujitsu Co., Ltd. Tracking Vision (TRV) has a Motion Estimation Processor (MEP) that can track more than 100 feature points by template matching at video rate (30 Hz).

Figure 12 shows a demonstration using the Tracking Vision. This figure shows an instant when the user moves the cube downward. Since we are using the hardware-based chroma-keying circuitry, the video image of the user's hand and the fiducial points are displayed with almost no delay. The cube image, on the other hand, is displayed with a delay of three to five frames, resulting in a noticeable misalignment. The end-to-end system delay was estimated to be 150 ms. The bottom line is that, even after implementing the Tracking Vision, the system performance (150 ms delay) was unsatisfactory. The performance evaluation and factors for this delay are discussed in Section 5.

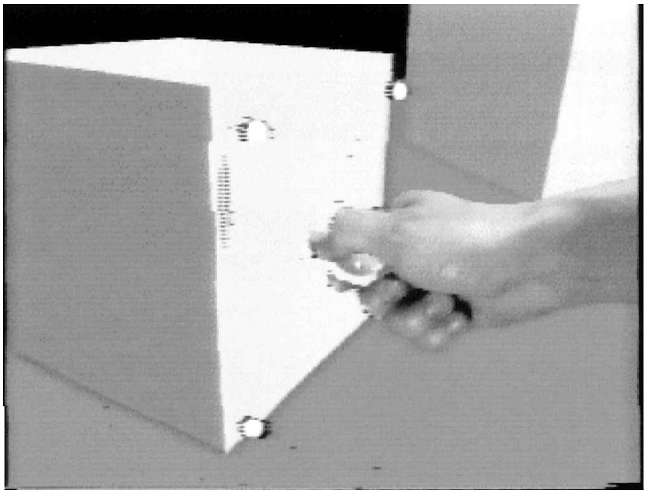


Figure 12. *Tracking mode using a fast tracker*

4.3.2 Background Swinging Problem.

Fiducial points at the endpoint of the haptic device are less likely to be occluded by the device or be out of the camera view than fiducial points on the base. Putting fiducial points at the endpoint of the haptic device, however, makes the tracking difficult, because the system has to track moving fiducial points from a moving camera. To render the virtual object and the background, we need ${}^{cam}T_{dev}$ and ${}^{cam}T_{base}$, respectively, as shown in Figure 13. Here, ${}^{A}T_{B}$ denotes a 4 × 4 transformation matrix from a coordinate system Σ_A to another one Σ_B. Since the camera motion estimator does not know the absolute location of the fiducial points, what it can estimate is the camera-to-device pose, i.e., ${}^{cam}T_{dev}$. Next, we obtain the camera-to-base pose, ${}^{cam}T_{base}$, by

$$ {}^{cam}T_{base} = {}^{cam}T_{dev}({}^{base}T_{dev})^{-1}, $$

where ${}^{base}T_{dev}$ is obtained from the joint sensor information of the haptic device.

Unfortunately, as it turned out, with this method the device motion causes the background image to swing

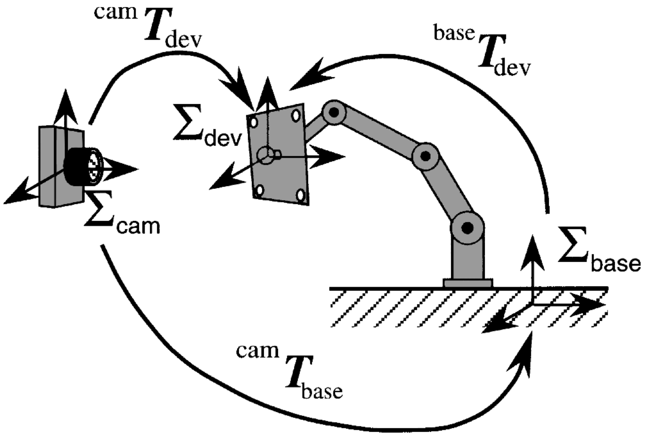


Figure 13. *Coordinate frames and transformations*

around even when the camera/display system remains stationary. This is because the estimation of the camera-to-device pose is delayed a few frames, while the device joint information is available almost immediately.

To solve this problem, we excluded the influence of the device motion from the camera motion estimation. Since ${}^{dev}T_{base}$ is available immediately, we can change the problem from "tracking moving fiducial points from a moving camera" to "tracking fixed fiducial points from a moving camera," by updating the locations of the fiducial points as though they are fixed on the base. In this case, ${}^{cam}T_{base}$ is first estimated, and then ${}^{cam}T_{dev}$ is obtained by

$${}^{cam}T_{dev} = {}^{cam}T_{base}\,{}^{base}T_{dev}.$$

After this modification, there is no longer interference of the background image by the device motion. Of course, this is not a fundamental solution, because, even with this method, the background image swings around when the camera is moved and stopped quickly. To solve this swinging problem completely, we must make the end-to-end system delay negligibly small. For more details, see Yokokohji, Hollis, & Kanade 1996b.

5 Performance Evaluation of the Prototype System

In this section, we evaluate the performance of the prototype WYSIWYF display. Since the prototype system includes some existing devices and is not an optimal configuration, it has many technical problems that need to be solved. Nevertheless, several important lessons were learned throughout the experimental studies.

Table 1. *Estimated Performance of the Visual Display Component*

Mode	Frame rate	Latency	Alignment error
Camera fixed	50 Hz	Negligible ($\leq$10 ms)	4 pixels*
Tracking by SIRIUS	3 Hz	0.9 sec	2 pixels
Tracking by TRV	20 Hz	150 ms	50 pixels

*As long as the motion is within 20 cm from the initial position.

5.1 Visual Display Part

The most important performance of the visual display part is the accuracy of visual/haptic registration. Registration errors can be classified as static and dynamic (Azuma & Bishop, 1994; Azuma, 1997). Table 1 shows estimated values of frame rate, end-to-end latency, and alignment errors in the prototype system. The static registration error causes an alignment error of approximately two pixels, which corresponds to roughly 1 mm in real space. We did not precisely estimate the depth error of the registration (which might be somewhat larger than 1 mm) because we used a single camera for registration. The depth error mostly affects the difficulty in reaching the virtual object. Since the visual display of the prototype system is not stereo, it is difficult anyway for the user to get a depth cue. One generally finds it necessary to adjust his or her hand position a bit closer or farther when reaching the virtual object.

Even with the one-shot registration at the initial position, the virtual object is well aligned to the real haptic device while moving the virtual object around in the camera-fixed mode. (See Figure 8.) The alignment error does not exceed four pixels as long as the haptic device is moved close to its initial position (within approximately 20 cm), indicating that the registration accuracy is acceptable. Because we used the camera parameters given by the manufacturer and did not conduct any camera calibration, the alignment error becomes large when the haptic device is moved very close to the camera. This kind of error can be reduced by careful camera calibration or online registration in tracking mode.

The end-to-end system latency is the main source of the dynamic registration error. We have to consider two factors: delay from the camera motion, and delay from the haptic device motion. In camera-fixed mode, only the second factor is effective. We have not precisely measured the total delay from the device motion to the virtual object motion in the display, but this delay was so small that the user could not notice it. From Figure 8, one can judge that the misalignment around the knob is always within a few pixels and a rough estimation gives an end-to-end delay of less than about 10 ms. Satoh, Tomono, and Kishino (1991) studied the influence of delay time on images with motion parallax and estimated that the acceptable upper limit is approximately 100 ms. We are uncertain if we can evaluate our case by this measure, but our rough estimation of the delay is less than this limit.

As shown in Table 1, frame rate and latency in camera-tracking mode with the SIRIUS video board were far beyond acceptable levels. When Tracking Vision was used, the frame rate was recovered to 20 Hz. The difference of frame rates between this mode and the camera-fixed mode (50 Hz) comes from the computational overhead needed for the Kalman filtering. Since the user's hand image is displayed through the chroma-key circuitry with almost no delay, the delay of the displayed images becomes noticeable for the user. The estimated delay was approximately 150 ms, which results in 50 pixels misalignment when the user moves the device at approximately 30 cm/s. Note that the misalignment error in SIRIUS tracking mode, on the other hand, was kept at two pixels, because the same video image was used for the registration and chroma-key. It shows that introducing another image input channel causes a synchronization problem.

There are several factors of delay from the camera motion. The fiducial point coordinates are obtained from the image in the previous frame (delayed by at least 33 ms). The Kalman filter is equivalent to a second-order system in the steady state (Higgins, 1975; Gennery, 1990) and causes some amount of delay in the high frequency range.

In summary, vision-based tracking can provide good enough accuracy for the static registration, and, in the camera-fixed mode, the system performance was satisfactory. However, the end-to-end system delay is still large even after the fast video tracker is used. Dynamic registration errors due to the end-to-end delay is a fundamental problem for head tracking in VR applications (Azuma, 1997). To compensate for the end-to-end delay, accurate prediction is required and additional sensors such as gyros and accelerometers will be necessary (Azuma & Bishop, 1994, 1995). We are also working on reducing dynamic registration errors by using accelerometers in conjunction with the vision-based tracking (Yokokohji, Sugawara, & Yoshikawa, 1998).

5.2 Haptic Display Part

Computation for constrained forces and collision impulses needs 20 ms to complete, which is a relatively large sampling period for force feedback. To send the haptic device endpoint information from the VxWorks system to the PowerOnyx, we used simple asynchronous socket communication. Even with such a slow computation cycle and an asynchronous communication, the user could not notice any lag between the force feedback and the displayed image. Miyasato and Nakatsu (1997) estimated that the acceptable upper limit of the delay time between visual sensation and haptic sensation is 100 ms, which is comparable with the motion parallax limit (Satoh et al., 1991). Although we have not precisely measured the time lag between the virtual object contact in the display and the response of the haptic device, we believe that the total delay is less than this limit of 100 ms. The apparent stiffness that the user feels when the rigid contact occurs is the sum of the structural stiffness of the haptic device and the servo stiffness of the joint control module. The PUMA has high structural stiffness and high servo stiffness like most industrial robots, and our prototype system could provide the user a realistic (or crisp) rigid contact feeling. In summary, our prototype demonstrated that the combination of the encountered-type haptic device and the motion-command type of haptic rendering algorithm is effective, and that an industrial robot with a high reduction-gear ratio is a reasonable choice for the haptic device.

We must mention some drawbacks of the industrial robot. First of all, safety is an important issue. Industrial robots are not designed to be used as haptic devices that will be touched and grasped by humans. Especially the encountered-type approach can potentially cause an accidental collision with the user through careless path planning. A careful path-planning algorithm to avoid unwanted collisions should be developed in the future. Structural singularities are also a problem. The PUMA is easy to fall into the "wrist-singular posture," when the fifth joint stretches out. A robot in a singular posture cannot move in arbitrary directions. If we try to resolve the endpoint motion by each joint motion while at a singular posture, some of the joint rates become infinitely large. RCCL is well designed for safety and shuts the power down when the joint velocity exceeds a certain limit. This is not very convenient, however, as operation is discontinued every time the robot gets close to the singular posture. We can introduce an artificial potential field in configuration space to avoid the singular posture, but the additional potential field would give the user a force that has no counterpart in the virtual environment, thereby giving the user false information.

Another problem is the mass of the haptic device. Theoretically speaking, any object with any mass can be simulated. Practically speaking, however, there is a lower limit of mass that the haptic device can simulate. If the mass to be simulated is below this limit, the haptic device becomes oscillatory and unstable. A fundamental solution to those problems would be to design a new lightweight mechanism that has no singular points in the important working volume.

In this section, we could give very little qualitative evaluations for visual and haptic components and the overall system. More qualitative evaluations should be conducted; for example, what is the allowable delay between vision and haptic stimuli, and what are the allowable static/dynamic registration errors? With the first prototype, however, the user can manipulate a virtual object quite realistically with his or her real hand image that is well aligned to the virtual object and with crisp force feedback by an industrial robot. Therefore, we feel that this first prototype WYSIWYF system shows the validity of the proposed approach.

6 Conclusions

This paper proposed a reasonable and workable method to realize correct visual/haptic registration or WYSIWYF (what you see is what you feel). The proposed method can be summarized as follows:

- vision-based tracking to realize correct visual/haptic registration
- chroma-key to extract the user's hand image from the same video source used for the tracking, which eliminates the need for a sensing device like a data glove
- combination of the encountered-type haptic device with the motion-command type haptic rendering algorithm, which can deal with two extreme cases: free motion and rigid constraint

Based on the proposed method, we built a prototype WYSIWYF display and have shown some demonstrations. The encountered-type approach provided realistic haptic sensations, such as free-to-touch and nonpenetrating contact. The haptic rendering algorithm controlling an industrial robot provided crisp move-and-collide sensations as well as rigid constraints. Although the prototype system has many unsolved problems, it showed a satisfactory level of performance in camera-fixed mode. The user can manipulate a virtual object realistically with precise alignment between the synthetic images and the real hand image together with the crisp haptic feedback.

Future topics in order to improve the performance of the prototype system are summarized as follows:

- using complementary sensors to make tracking more robust
- precise prediction to compensate for the end-to-end delay of tracking
- introducing an HMD
- replacing the PUMA with a new device designed expressly for haptics

- tracking the user's hand when displaying multiple virtual tools

In this paper, we also discussed briefly the importance of WYSIWYF. However, the importance of WYSIWYF can be established only through thoughtful experimentation with subjects performing well-defined tasks. Comparison between WYSIWYF and non-WYSIWYF situations as well as more detailed quantitative evaluations will be necessary in the future.

Acknowledgment

The authors would like to express their grateful thanks to Mr. Michihiro Uenohara, Visiting Research Scientist of CMU, for his comments on the vision-based tracking and for his help in building the LCD/camera system. Dr. David Baraff, Assistant Professor of CMU, gave many valuable comments on the physically based simulation. Dr. John Lloyd provided many suggestions for solving the RCCL/RCI installation problems. Mr. Mike Blackwell, Senior Research Engineer of CMU, kindly maintained the SGI PowerOnyx and installed the SIRIUS video. Dr. Alfred Rizzi, Postdoctoral Research Associate of CMU, helped to solve the hardware problem of the PUMA. Finally, the authors express their appreciation to SHARP Corporation for providing the LCD panel and to FUJITSU Laboratories Ltd. for their technical support of the Tracking Vision system.

This paper was presented in part at the IEEE 1996 Virtual Reality Annual International Symposium (VRAIS'96), Santa Clara, California, March 30–April 3, 1996

References

Azuma, R. T. (1997). A Survey of augmented reality. *Presence: Teleoperators and Virtual Environments, 6*(4), 355–385.

Azuma, R., & Bishop, G. (1994). Improving static and dynamic registration in an optical see-through HMD. *Proceedings of SIGGRAPH'94,* 197–204.

———. (1995). A frequency-domain analysis of head-motion prediction. *Proceedings of SIGGRAPH'95,* 401–408.

Bajura, M., Fuchs, H., & Ohbuchi, R. (1992). Merging virtual objects with the real world. *Proceedings of SIGGRAPH'92,* 203–210.

Bajura, M., & Neumann, U. (1995). Dynamic registration correction in augmented-reality systems. *Proceedings of 1995 IEEE Virtual Reality Annual International Symposium (VRAIS'95),* 189–196.

Baraff, D. (1994). Fast contact force computation for nonpenetrating rigid bodies. *Proceedings of SIGGRAPH'94,* 23–34.

Bergamasco, M., Allotta, B., Bosio, L., Ferretti, L., Parrini, G., Prisco, G. M., Salsedo, F., & Sartini, G. (1994). An arm exoskeleton system for teleoperation and virtual environments applications. *Proceedings of 1994 IEEE International Conference on Robotics and Automation,* 1449–1454.

Bernotat, R. K. (1970). Rotation of visual reference systems and its influence on control quality. *IEEE Trans. on Man-Machine Systems,* MMS-11(2), 129–131.

Bishop, G. (1984). Self-tracker: A smart optical sensor on silicon. Ph.D. Thesis. Univ. of North Carolina at Chapel Hill.

Burdea, G. C. (1996). *Force and Touch Feedback for Virtual Reality.* John Wiley & Sons.

Deering, M. (1992). High resolution virtual reality. *Proceedings of SIGGRAPH'92,* 195–202.

Gennery, D. B. (1990). Properties of a random-acceleration recursive filter. *JPL Internal Report D-8580.*

———. (1992). Visual tracking of known three-dimensional objects. *International Journal of Computer Vision, 7*(3), 243–270.

Groen, J., & Werkhoven, P. J. (1998). Visuomotor adaptation to virtual hand positioning in interactive virtual environment, *Presence: Teleoperators and Virtual Environments, 7*(5), 429–446.

Gruenbaum, P. E., Overman, T. L., Knutson, B. W., McNeely, W. A., & Sowizral, H. A. (1995). Implementation of robotic graphics for a virtual control panel. *VRAIS'95 Video Proceedings.*

Gruenbaum, P. E., McNeely, W. A., Sowizral, H. A., Overman, T. L., & Knutson, B. W. (1997). Implementation of dynamic robotic graphics for a virtual control panel. *Presence: Teleoperators and Virtual Environments, 6*(1), 118–126.

Hammerton, M., & Tickner, A. H. (1964). Transfer of training between space-oriented and body-oriented control situations. *British Journal of Psychology, 55*(4), 433–437.

Henmi, K., & Yoshikawa, T. (1998). Virtual lesson and its application to virtual calligraphy system. *Proceedings of 1998 IEEE International Conference on Robotics and Automation,* 1275–1280.

Higgins, Jr., W. T. (1975). A comparison of complementary and Kalman filtering. *IEEE Trans. on Aerospace and Electronic Systems,* AES-11(3), 321–325.

Hirota, K., & Hirose, M. (1993). Development of surface display. *Proceedings of 1993 IEEE Virtual Reality Annual International Symposium (VRAIS'93),* 256–262.

———. (1995). Simulation and presentation of curved surface in virtual reality environment through surface display. *Proceedings of 1995 IEEE Virtual Reality Annual International Symposium (VRAIS'95),* 211–216.

Iwata, H. (1990). Artificial reality with force-feedback: Development of desktop virtual space with compact master manipulator. *Proceedings of SIGGRAPH'90,* 165–170.

Jacobs, M. C., Livingston, M. A., & State, A. (1997). Managing latency in complex augmented reality systems. *Proceedings of 1997 Symposium on Interactive 3D Graphics,* 49–54.

Kanade, T., Kano, H., Kimura, S., Yoshida, A., & Oda, K. (1995). Development of a video-rate stereo machine. *Proceedings of International Conference on Intelligent Robots and Systems (IROS'95), 3,* 95–100.

Kanade, T., Yoshida, A., Oda, K., Kano, H., & Tanaka, M. (1996). A stereo machine for video-rate dense depth mapping and its new applications. *Proceedings of 15th Computer Vision and Pattern Recognition Conference (CVPR),* 196–202.

Kornheiser, A. S. (1976). Adaptation to laterally displaced vision: A review. *Psychological Bulletin, 83*(5), 783–816.

Kozak, J. J., Hancock, P. A., Arthur, E. J., & Chrysler, S. T. (1993). Transfer of training from virtual reality. *Ergonomics, 36*(7), 777–784.

Lloyd, J., & Hayward, V. (1992). *Multi-RCCL User's Guide.* McGill University.

Lowe, D. G. (1992). Robust model-based motion tracking through the integration of search and estimation. *International Journal of Computer Vision, 8*(2), 113–122.

Luecke, G. R., & Winkler, J. (1994). A magnetic interface for robot-applied virtual forces. *Proceedings of 1994 ASME WAM,* DSC-55(1), 271–276.

McNeely, W. A. (1993). Robotic graphics: A new approach to force feedback for virtual reality. *Proceedings of 1993 IEEE Virtual Reality Annual International Symposium (VRAIS'93),* 336–341.

Metzger, P. J. (1993). Adding reality to the virtual. *Proceedings of 1993 IEEE Virtual Reality Annual International Symposium (VRAIS'93),* 7–13.

Minsky, M., Ohu-young, M., Steele, O., Brooks, Jr., F. P., & Behensky, M. (1990). Feeling and seeing: Issues in force display. *Computer Graphics, 24*(2), 253–243.

Miyasato, T., & Nakatsu, R. (1997). Allowable delay between images and tactile information. *Proc. of International Conference on Virtual Systems and Multimedia (VSMM'97),* 84–89.

Norris, E. B., & Spragg, S. D. S. (1953). Performance on a following tracking task (modified SAM two-hand coordination test) as a function of the relations between direction of rotation of controls and direction of movement of display. *Journal of Psychology, 35,* 119–129.

National Research Council (1995). *Virtual Reality: Scientific and Technological Challenges.* National Academy Press.

Pichler, C., Radermacher, K., Boeckmann, W., Rau, G., & Jakse, G. (1997). Stereoscopic visualization in endoscopic surgery: Problems, benefits, and potentials, *Presence: Teleoperators and Virtual Environments, 6*(2), 198–217.

Rolland, J. P., Biocca, F. A., Barlow, T., & Kancherla, A. (1995). Quantification of adaptation to virtual-eye location in see-thru head-mounted displays. *Proceedings of 1995 IEEE Virtual Reality Annual International Symposium (VRAIS'95),* 56–66.

Sato, M., Hirata, Y., & Kawarada, H. (1992). Space interface device for artificial reality—SPIDAR. *Systems and Computers in Japan, 23*(12), 44–54.

Satoh, T., Tomono, A., & Kishino, F. (1991). Allowable delay time of images with motion parallax, and high speed image generation. *Proceedings of Visual Communications and Image Processing '91: Image Processing (SPIE Vol. 1606),* 1014–1021.

Schmidt, R. A. (1988). Motor control and learning: A Behavioral Emphasis (2nd ed.). Human Kinetic Publishers, Inc.

Spragg, S. D. S., Finck, A., & Smith, S. (1959). Performance on a two-dimensional following tracking task with miniature stick control, as a function of control-display movement relationships. *Journal of Psychology, 48,* 247–254.

State, A., Hirota, G., Chen, D. T., Garrett, W. F., & Livingston, M. A. (1996). Superior augmented reality registration by integrating landmark tracking and magnetic tracking. *Proceedings of SIGGRAPH'96,* 429–438.

Tachi, S., Maeda, T., Hirata, R., & Hoshino, H. (1994). A construction method of virtual haptic space. *Proc. of Int. Conf. on Artificial Reality and Telepresence (ICAT'94),* 131–138.

———. (1995). A machine that generates virtual haptic space. *VRAIS'95 Video Proceedings.*

Uenohara, M., & Kanade, T. (1995). Vision-based object registration for real-time image overlay. *Computers in Biology and Medicine, 25*(2), 249–260.

Yokokohji, Y. (1998). A visual/haptic interface to virtual environment (WYSIWYF display) and its application. *Proceedings of 1998 IEEE and ATR Workshop on Computer Vision for Virtual Reality Based Human Communications (CVVRHC'98),* 99–104.

Yokokohji, Y., & Yoshikawa, T. (1992). Bilateral control of master-slave manipulators for ideal kinesthetic coupling: Formulation and experiment. *Proceedings of 1992 IEEE International Conference on Robotics and Automation,* 849–858.

Yokokohji, Y., Hollis, R. L., & Kanade, T. (1996a). What you can see is what you can feel: Development of a visual/haptic interface to virtual environment. *Proceedings of 1996 IEEE Virtual Reality Annual International Symposium (VRAIS'96),* 46–53.

———. (1996b) Vision-based visual/haptic registration for WYSIWYF display. *Proceedings of 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'96),* 1386–1393.

Yokokohji, Y., Hollis, R. L., Kanade, T., Henmi, K., & Yoshikawa, T. (1996c). Toward machine mediated training of motor skill. *Proceedings of 1993 IEEE International Symposium on Robot and Human Communication (RO-MAN'96),* 32–37.

Yokokohji, Y., Sugawara, Y., & Yoshikawa, T. (1998). Vision-based head tracking for image overlay and reducing dynamic registration error with acceleration measurement, *Proceedings of 1998 Japan-USA Symposium on Flexible Automation,* 1293–1296.

Yoshikawa, T., Yokokohji, Y., Matsumoto, T., & Zheng, X.-Z. (1995). Display of feel for the manipulation of dynamic virtual objects. *ASME J. Dynamic Systems, Measurement, and Control, 117*(4), 554–558.

Yoshikawa, T., & Henmi, K. (1996). Availability of virtual lesson and construction of virtual syuuji system. *Proceedings of the Virtual Reality Society of Japan Annual Conference,* 89–90. (In Japanese.)

Yoshikawa, T., & Nagura, A. (1997). A touch and force display system for haptic interface. *Proceedings of 1997 IEEE International Conference on Robotics and Automation,* 3018–3024.

Ward, M., Azuma, R., Bennett, R., Gottschalk, S., & Fuchs, H. (1992). A demonstrated optical tracker with scalable work area for head-mounted display system. *Proceedings of 1992 Symposium on Interactive 3D Graphics,* 43–52.

Welch, R. B., & Warren, D. H. (1980). Immediate perceptual response to intersensory discrepancy. *Psychological Bulletin, 88*(3), 638–667.

Wren, C., Azarbayejani, A., Darrell, T., & Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *IEEE Trans. on Pattern Analysis and Machine Intelligence, 19*(7), 780–785.

Witkin, A., Baraff, D., & Kass, M. (1994). An introduction to physically based modeling. *SIGGRAPH'94 Course Notes.*

A Baraff's algorithm

A.1 Static Constrained Forces

Suppose a rigid body is resting on another object with n contact points. For simplicity, a frictionless case is considered. At the i-th contact point, a unit surface normal vector is defined such that the vector is directed outward from the surface. The i-th contact point acceleration, $\ddot{d}_i$, which is the normal component of the translational acceleration of the object at the i-th contact point, can be expressed by the following equation:

$$\ddot{d}_i = a_{i1}f_1 + a_{i2}f_2 + \cdots + a_{in}f_n + b_i, \qquad \text{(A.1)}$$

where f_j denotes the magnitude of the j-th contact force; a_{ij} is the coefficient representing the contribution of the j-th contact force to the i-th contact acceleration; and b_i is the term containing Coriolis and centrifugal forces and the external force.

To realize nonpenetrating rigid-body motion, the following conditions should be satisfied:

$$\ddot{d}_i \geq 0, f_i \geq 0 \text{ and } f_i \cdot \ddot{d}_i = 0. \qquad \text{(A.2)}$$

Getting equations (A.1) and (A.2) for all n contact points together, we get

$$Af + b \geq 0, \qquad \text{(A.3)}$$

$$f \geq 0 \text{ and } f^T(Af + b) = 0. \qquad \text{(A.4)}$$

The problem is to find f_is which satisfy equations (A.3) and (A.4). This problem can be regarded as an optimization problem such as linear complementarity programming or quadratic programming. But solving such an optimization problem requires much computational effort and might not be adequate for the purpose of interactive simulation. Baraff (1994) has proposed a fast algorithm to compute the contact forces; it is a kind of iterative method by pivoting matrix A. In the frictionless case, his algorithm is guaranteed to converge to the correct solution. In the friction case, his algorithm also works well in practice.

A.2 Colliding Impulses

Suppose that a rigid-body object is colliding with another rigid object with m colliding points. Let v_i^+ and v_i^- denote normal components of the velocities after and before the collision at the i-th colliding point, respectively. Here v_i^+ can be expressed by the following equation:

$$v_i^+ = v_i^- + a_{i1}j_1 + a_{i2}j_2 + \cdots + a_{im}j_m, \quad (A.5)$$

where j_i denotes the impulse at the i-th colliding point; and a_{ij} is the coefficient representing the contribution of the j-th impulse to the i-th postcollision velocity.

Newton's law of restitution says

$$v_i^+ + \varepsilon_i v_i^- \geq 0, \quad (A.6)$$

where ε_i denotes the coefficient of restitution at the i-th colliding point. The reason why we use "≥" in equation (A.6) instead of "=" is that there might be no impulse at the i-th colliding point, but the object may be pushed away by the impulses at other colliding points.

For nonpenetrating rigid-body collisions, the following conditions should be satisfied:

$$j_i \geq 0 \text{ and } j_i \cdot (v_i^+ + \varepsilon_i v_i^-) = 0. \quad (A.7)$$

Substituting equation (A.5) to (A.6), we get

$$a_{i1}j_1 + a_{i2}j_2 + \cdots + a_{im}j_m + v_i^- + \varepsilon_i v_i^- \geq 0. \quad (A.8)$$

Getting equations (A.8) and (A.7) for all m colliding points together, we get

$$Aj + c \geq 0, \quad (A.9)$$

$$j \geq 0 \text{ and } j^T(Aj + c) = 0. \quad (A.10)$$

The problem is to find j_is which satisfy equations (A.9) and (A.10). Note that equations (A.9) and (A.10) have the same forms as equations (A.3) and (A.4). Therefore, we can use the same algorithm used for contact forces to find these impulses. Once we have obtained j_is, we can get the object velocities after the collision, reset the state variables, and restart to solve the ODEs. For more details of the algorithm, see Baraff (1994) and Witkin, Baraff, & Kass (1994).

A.3 Simulation Algorithm

An actual computation flow at every simulation cycle is as follows:

STEP 1: Applied force/torque by the user is measured by the force/torque sensor attached to the endpoint of the haptic device.

STEP 2: If any resting contact points were found in step 4 in the previous simulation cycle, compute constraint forces that satisfy equations (A.3) and (A.4).

STEP 3: Solve Newton/Euler equations with the measured force/torque in step 1 and the constrained forces obtained in step 2. Integrate the resultant acceleration and update the state variables.

STEP 4: Check for collisions with other objects and find the colliding contact points and the resting contact points for the new state variables.

STEP 5: If any colliding contact points were found in step 4, compute impulses that satisfy equations (A.9) and (A.10). Otherwise, go to step 7.

STEP 6: Compute the object velocity after the collision and reset the state variables.

STEP 7: Send the motion command to the haptic display. The motion command could be given by position, velocity, or acceleration, depending on the device controller type.

STEP 8: Increment time step and go to step 1.

B Basic Formulation of Haptic Rendering Algorithm

Two approaches for haptic rendering are introduced, and it is shown that both cannot deal with either of two extreme cases: free motion and rigid constraint.

B.1 Two Approaches of Haptic Rendering

First of all, the dynamics of a simple one-DOF haptic device, which is shown in Figure 14, is modeled. We suppose that the user keeps holding the device tightly and never releases it.

$$f_m + \tau = M\ddot{x} + B\dot{x}, \tag{B.1}$$

where f_m denotes the force applied by the operator, and τ is the force generated by the actuator of the device. The mass of the device is denoted by M, while B is the coefficient of viscosity. The displacement of the device is denoted by x.

Suppose that a virtual environment has the following impedance character:

$$f_m = m_w\ddot{x} + b_w\dot{x} + k_w x, \tag{B.2}$$

where m_w, b_w, and c_w are mass, viscous coefficient, and stiffness of the virtual environment, respectively. Equation (B.2) specifies the goal of the haptic device behavior.

Hereafter we consider two approaches of haptic rendering to realize the relationship of equation (B.2). To simplify the problem, some ideal sensors are supposed: the force sensor that can measure f_m, the position, velocity sensors that can measure x and $\dot{x}$, and the accelerometer that can get $\ddot{x}$.

Equation (B.2) can be expressed in more general form as

$$f_m = F_f(\ddot{x}; \dot{x}, x). \tag{B.3}$$

We can extract the acceleration term and rewrite equation (B.3) to give

$$\ddot{x} = F_\alpha(f_m; \dot{x}, x), \tag{B.4}$$

which is the closed form of direct dynamics of the virtual environment. Yoshikawa et al. (1995) showed the two basic methods for displaying the operating feel.

1. Based on the measurement of device motion $\ddot{x}$ and the current state $(\dot{x}, x)$, obtain the necessary resultant force f_m (with equation (B.3)) and control the device to realize this force.
2. Based on the measurement of force f_m and the current state $(\dot{x}, x)$, obtain the corresponding acceleration $\ddot{x}$ (with equation (B.4)) and control the device to realize this acceleration or its integrals (velocity or position).

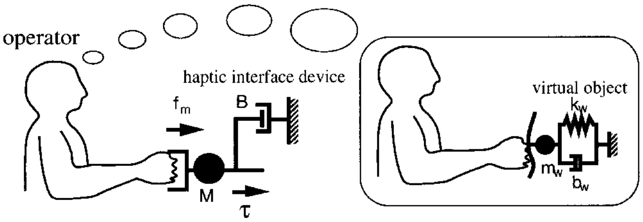


Figure 14. *One-DOF model*

The former method is called "measuring force and displaying motion" or "motion-command type," and the latter is called "measuring motion and displaying force" or "force-command type." We formulate two approaches by using the most simple case: equation (B.2).

B.2 Force-Command Type

Assuming that we can get the information of $\ddot{x}$, $\dot{x}$ and x, we can specify the following actuator force:

$$\tau = \tau_f \triangleq \hat{M}\ddot{x} + \hat{B}\dot{x} - f_{cmd}(x, \dot{x}, \ddot{x}), \tag{B.5}$$

where $\hat{M}$ and $\hat{B}$ are estimate values of M and B, respectively.

Substituting the above equation into equation (B.1), and assuming that the estimated values are precise ($\hat{M} = M$, $\hat{B} = B$), we get

$$f_m = f_{cmd}, \tag{B.6}$$

and the force command is actually realized. The force command can be given by

$$f_{cmd} = m_w\ddot{x} + b_w\dot{x} + k_w x, \tag{B.7}$$

then we get the final goal:

$$f_m = m_w\ddot{x} + b_w\dot{x} + k_w x. \tag{B.8}$$

B.3 Motion-Command Type

Motion-command type specifies acceleration a_{cmd} to be realized instead of force. The actuator force is given as follows:

$$\tau = \tau_a \triangleq -f_m + \hat{B}\dot{x} + \hat{M}a_{cmd}(x, \dot{x}, f_m). \qquad (B.9)$$

Again, assuming that the estimated parameters are precise enough ($\hat{M} = M$, $\hat{B} = B$), we get

$$\ddot{x} = a_{cmd}, \qquad (B.10)$$

and the acceleration command can be specified as

$$a_{cmd} = (f_m - k_w x - b_w \dot{x})/m_w, \qquad (B.11)$$

which is equivalent to our final goal:

$$f_m = m_w \ddot{x} + b_w \dot{x} + k_w x. \qquad (B.12)$$

Note that, when we calculate a_{cmd}, we assume that $m_w \neq 0$.

B.4 Two Extreme Cases

Performance of haptic device used for teleoperation and VR should be evaluated in two extreme cases: completely free motion and rigid constraint (Yokokohji & Yoshikawa, 1992). An ideal device must behave in such a way that the user does not feel its existence in free motion (transparency), while it does not move against any exerted force when simulating a rigid wall. Practically it is difficult to realize such an ideal situation. We can theoretically show that both motion command-type and force-command type cannot deal with either of two extreme cases.

B.4.1 Free Motion. When the operator's hand is free in the virtual space, all of m_w, b_w, and k_w become zero.

As a result, we cannot specify the motion command by equation (B.11). Consequently, we cannot use the motion-command type in the case of free motion.

The force command in equation (B.7) simply becomes

$$f_{cmd} = 0, \qquad (B.13)$$

and

$$f_m = 0 \qquad (B.14)$$

is realized.

Of course, the above discussion is based on some ideal assumptions, and, in a practical sense, realization of equation (B.14) is almost impossible.

B.4.2 Rigid Wall. When the virtual environment is rigid wall, either m_w, b_w, or k_w should be infinitely large, which means that we cannot specify the force command by equation (B.7). Therefore, we cannot use the force-command type for displaying a rigid-wall-like virtual environment.

In the case of motion-command type, we simply specify the motion command as

$$a_{cmd} = 0, \qquad (B.15)$$

assuming that $m_w = \infty$, and we get

$$\ddot{x} = 0. \qquad (B.16)$$

To prevent the effect of "drifting," we can add a feedback component,

$$a_{cmd} = -k_1\dot{x} - k_2 x. \qquad (B.17)$$

The feedback gains k_1 and k_2 is just for preventing the drift and need not be large, so it is no problem to add it to the general case:

$$a_{cmd} = (f_m - k_w\dot{x} - k_w x)/m_w - k_1\dot{x} - k_2 x, \qquad (B.18)$$

and we get

$$f_m = m_w\ddot{x} + (b_w + m_w k_1)\dot{x} + (k_w + m_w k_2)x. \qquad (B.19)$$