"What You can See Is What You can Feel."

-Development of a Visual/Haptic Interface to Virtual Environment-

Yasuyoshi Yokokohji, Ralph L. Hollis, and Takeo Kanade
The Robotics Institute
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213, U.S.A.
{yokokoji | rhollis | tk}@cs.cmu.edu
http://www.cs.cmu.edu/~msl

Abstract

In this paper, we propose a new concept of visual/haptic interfaces called WYSIWYF display. The proposed concept provides correct visual/haptic registration using a vision-based object tracking technique and a video keying technique so that what the user can see via a visual interface is consistent with what he/she can feel through a haptic interface. Using Chroma Keying, a live video image of the user's hand is extracted and blended with the graphic scene of the virtual environment. The user's hand "encounters" the haptic device exactly when his/her hand touches a virtual object in the blended scene. The first prototype has been built and the proposed concept was demonstrated.

1. Introduction

Haptic interfaces have been recognized as important input/output channels to/from the virtual environment [10][12][17]. Usually a haptic interface is implemented with a visual display interface such as a head-mounted display or a stereoscopic display screen. Correct registration of visual and haptic interfaces, however, is not easy to achieve and has not been seriously considered. For example, some systems have a graphics display simply beside the haptic interface resulting in a "feeling here but looking there" situation. Poor visual/haptic registration could result in inconsistent situations, where the user feels the reaction forces from the haptic device before the user's hand reaches the virtual object in the visual display, or vice versa.

One of the most important potential applications of VR systems is training and simulation. For training visual-motor skills (e.g. pick-and-place), correct visual/haptic registration is important because a visual-motor skill is composed of tightly coupling visual stimuli (associated with task coordinates) and kinesthetic stimuli (associated with body coordinates). If there is an inconsistency between the two kinds of stimuli, there would be no significant skill transfer[8], or in an even worse case, the training might negatively hurt performance in real situations (negative skill transfer).

In this paper, we propose a new concept of visual/haptic interfaces called a WYSIWYF (What You can See Is What You can Feel) display. The proposed concept ensures correct visual/haptic registration so that what the user can see from the visual interface is consistent with exactly what he/she can feel through the haptic device. In other words, the user's hand can "encounter" the haptic device exactly when his/her hand touches an object in the virtual environment. A vision-based object tracking technique and a video-keying technique are used to get correct visual/haptic registration. The first prototype was built using a color liquid crystal display (LCD) panel and a CCD camera for the visual interface component and a PUMA 560 robot for the haptic interface component.

2. WYSIWYF display concept

Figure 1 illustrates the proposed concept of the visual/haptic interface. In this section, we first discuss requirements of correct visual/haptic registration and then introduce our proposed method.

2.1. Requirements of WYSIWYF

In this subsection, we will discuss how to realize correct visual/haptic registration, namely the WYSIWYF situation. Figure 2 shows several coordinate frames in the visual/haptic interface. Σ_{eye} is attached to the user's head representing his/her head position and viewing direction. Σ_{hand} is attached to the user's actual hand while Σ_{v_hand} is connected to the synthetic hand in the virtual environment. Σ_{dev} is

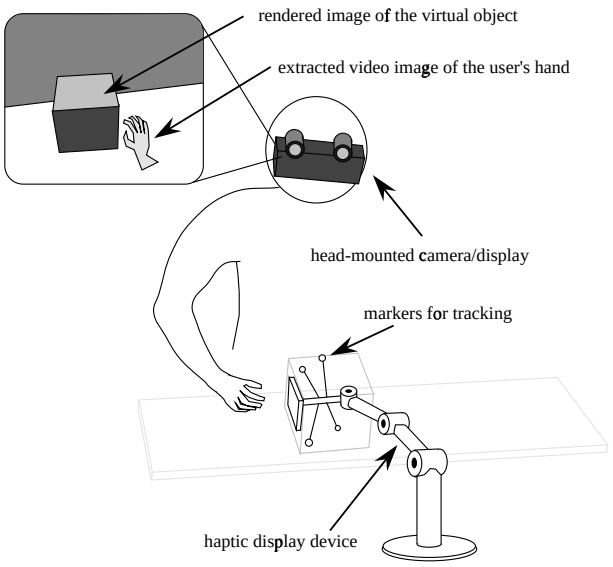


Figure 1. WYSIWYF display

attached to the tip of the haptic device. $\boldsymbol{\Sigma}_{v\_obj}$ is attached to the virtual object. $\boldsymbol{\Sigma}_{base}$ is the base coordinate frame of the real environment while $\boldsymbol{\Sigma}_{v\_env}$ is the base coordinate frame of the virtual environment. Hereafter let ${}^{A}\boldsymbol{T}_{B}$ denote a 4×4 homogeneous transformation matrix from $\boldsymbol{\Sigma}_B$ to $\boldsymbol{\Sigma}_A$.

First, the user's head should be tracked correctly, or the following relationship should be satisfied:

$$ {}^{eye}\boldsymbol{T}_{v\_env} = {}^{eye}\boldsymbol{T}_{base} \tag{1} $$

Assuming the above correct head tracking, WYSIWYF can be realized if the following two equations hold:

$$ {}^{eye}\boldsymbol{T}_{v\_hand} = {}^{eye}\boldsymbol{T}_{hand} \tag{2} $$

$$ {}^{eye}\boldsymbol{T}_{v\_obj} = {}^{eye}\boldsymbol{T}_{dev} \tag{3} $$

Eq.(2) means that the user's synthetic hand in the display is correctly registered to his/her actual hand with respect to his/her eye coordinates. Eq.(3) means that the virtual object is correctly registered to the haptic device.

2.2. Realizing WYSIWYF: Vision-based visual/haptic registration

To realize WYSIWYF, we will introduce a vision-based object tracking/registration technique and a video blending technique. Figure 3 shows the flow of the process. Each subprocess is explained below.

STEP 1: Vision based head-tracking and virtual environment rendering

Using the vision-based tracking technique, the user's head pose can be estimated and tracked with respect to the target object in the camera view. The target object could be either the haptic device or a fixed object in the working environment such as a table.

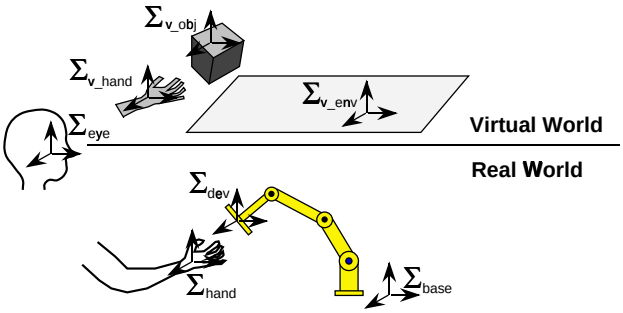


Figure 2. Coordinate frames of the visual/haptic interface

If the haptic device is used for the target object, the relative pose between the haptic device and the user's eye, i.e. ${}^{eye}\boldsymbol{T}_{dev}$, is estimated (assuming that a constant transformation matrix from the user's eye to the camera is calibrated), and the virtual object will be rendered based on this estimation, that is

$$ {}^{eye}\boldsymbol{T}_{v\_obj} = {}^{eye}\boldsymbol{T}_{dev} \tag{4} $$

Since the haptic device has joint sensors, we can get ${}^{dev}\boldsymbol{T}_{base}$, and the user's head pose with respect to the base coordinates is obtained by

$$ {}^{eye}\boldsymbol{T}_{base} = {}^{eye}\boldsymbol{T}_{dev}\,{}^{dev}\boldsymbol{T}_{base} \tag{5} $$

The background image will be rendered based on this, that is

$$ {}^{eye}\boldsymbol{T}_{v\_env} = {}^{eye}\boldsymbol{T}_{base} \tag{6} $$

Eqs.(4) and (6) are equivalent to eqs.(1) and (3). In case the target object is a fixed one, ${}^{eye}\boldsymbol{T}_{base}$ is estimated first and ${}^{eye}\boldsymbol{T}_{dev}$ is obtained using joint sensor information from the haptic device.

In the computer vision field, several techniques have been developed for tracking the object in the video image[20]. If we put special markers on the target object as shown in Fig.1, the tracking problem becomes relatively easy with a simple image processing effort. At least three markers are necessary to estimate the pose of the target object (position/orientation in three dimensional space).

STEP 2: Displaying the user's hand image

The CCD camera for marker tracking can also capture the working scene including the user's hand. If we could extract the portion of the user's hand image, we could superimpose this image onto the graphical image of the virtual environment instead of rendering a synthetic hand image. Using

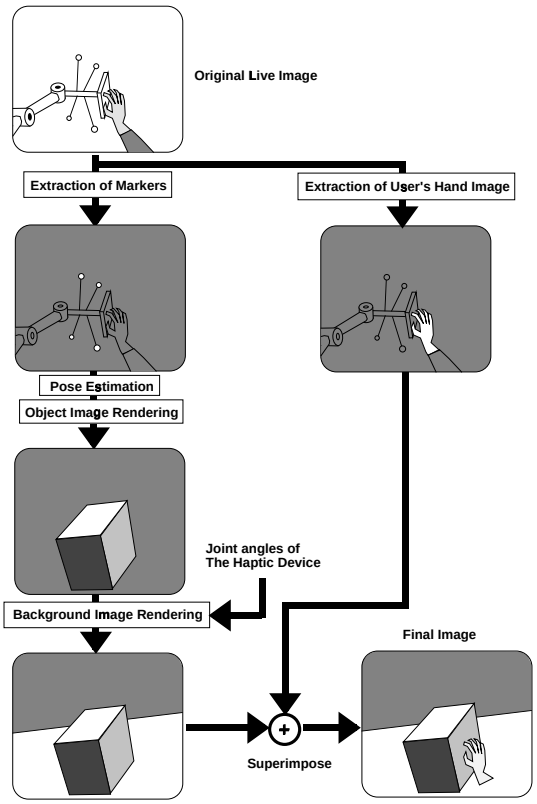


Figure 3. Flow of the blending process

a real hand image has the following advantages: first, one of the WYSIWYF requirements eq.(2) is automatically established, assuming that the camera's lens parameters are identical to those of the user's eye and the camera is located very close to the user's eye; second, the user does not need to wear any glove-like sensors to measure hand location and finger angles.

The easiest way to extract the user's hand image is to use the "Chroma Keying" technique by painting a uniform color (usually blue) on everything in the actual working environment (even the haptic device) except the user's hand. Metzger[15] has already proposed using Chroma Keying to mix virtual and real scenes. He showed two possible cases: overlaying images of the real world on top of the virtual world scene and overlaying virtual imagery onto a real world scene. Our approach corresponds to the former case, adding the reality to the virtual, like well known weather forecast TV programs. Bajura et al.[1] applied "Luna Keying" for merging medical ultrasound images with the real image of the patient body.

2.3. Encountered type haptic display

McNeely[14] classified haptic displays into the following three types: (i) worn-type (e.g., exoskeleton master), (ii) held-type (e.g., universal hand controller), (iii) encountered-type. A worn-type or held-type haptic interface also works as an input device to measure the user's motion, requiring that the device always be physically connected to some part of the user's body (typically arm, hand or fingers) during the operation. This requirement limits the user's workspace and obstructs the user's free motion.

With the encountered-type, on the other hand, the user need not keep holding the haptic device all the time. Instead, the system tracks the motion of the user's hand and places the haptic device in the appropriate location, waiting for the user to "encounter" it (*surface display mode*). Once the user encounters the device, it responds to the forces exerted by the user, based on the virtual object model (*admittance display mode*). McNeely[14][7], Hirota and Hirose[9], and Tachi et al.[19] have already proposed and implemented the encountered type haptic interface, which they called *robotic graphics*, *surface display*, and *haptic space* respectively.

Our WYSIWYF concept adopts the encountered-type approach for the haptic interface. Two display modes in this approach are discussed as follows.

Surface display mode

The device must display the appropriate part of the surface of the virtual object before the user touches the device. The location of the device is determined according to the current position of the user's hand which is obtained by an appropriate tracking sensor. The tracking sensor could be a video camera that tracks an infrared LED attached at the user's fingertip[7], a magnetic sensor, or a passive link mechanism[19]. As shown in Fig. 4, the device is placed at the position of the virtual object closest from the user.

It should be noted that the accuracy of the tracking sensor does not directly affect the accuracy of the haptic rendering, because the tracking data is used only for determining which part of the surface should be displayed. Once the displaying surface area is selected, the device is positioned with accurate joint sensors implemented in the haptic device.

Of course, the device could not cope with the user's quick or tricky motion (the device might have to jump from one object to another), and a careless path planning may potentially result in dangerous collisions with the user. Another difficulty is that the device cannot display an arbitrary surface shape unless ideal robotic graphics[14] are realized in the future. Currently the user would be restricted to access some limited part of the object such as buttons and knobs[7] or limited kinds of edges and faces[19].

Admittance display mode

When the user touches the device, the device changes its mode into the admittance display mode. Most haptic rendering algorithms are based on impedance control [12][17],

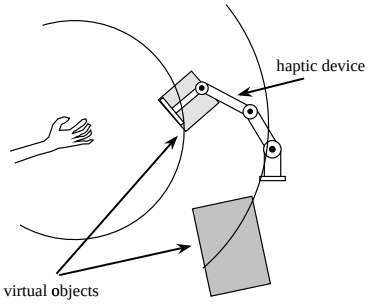


Figure 4. Surface display mode

where the position and the velocity of the haptic device are measured and the appropriate force is displayed to the user. With this approach, a force can be generated only when penetration occurs between two objects. To render a rigid object with the impedance display approach, the stiffness parameter should be large enough and a high sampling rate is required, otherwise the simulation tends to be unstable[5].

Admittance display mode, on the other hand, is based on the admittance control, i.e. measuring force and displaying motion. We take the admittance display approach, because it is consistent with the physically-based simulation algorithm with nonpenetration constraints shown in section 4.

3. Discussions

3.1. Vision-based tracking vs. emitter-based tracking

An advantage of the vision-based tracking technique over the conventional emitter based tracking technique, such as magnetic position tracker and ultrasonic sensor, is that the *relative* pose between the target object and the camera can be directly estimated. With the emitter-based sensing approach, on the other hand, the *absolute* position/orientation of the sensor or user's head with respect to the emitter is first obtained. To obtain the relative pose between the user and target object, the target object position/orientation with respect to the emitter should be given in advance. Unless another sensor is attached onto the target object, the user cannot move the object.

The group at the University of North Carolina[21] has been developing a see-through HMD using an optoelectronic head-tracking architecture, where a special ceiling equipped with infrared LED's is necessary. Bishop proposed a simple optical-based head tracking method in natural scene so as to make a custom integrated sensor chip[3]. Our vision-based approach has a potential to be more general than the above approaches in the sense that not only the haptic device but multiple movable objects in the natural scene can be tracked.

3.2. The real hand image vs. the synthesized hand image

The chroma keying technique allows us to extract a live image of the user's hand relatively easily. We can also merge some real objects other than the user's hand with the virtual scene (e.g., touching a virtual object with an real tool). In the proposed architecture, a single sensor (i.e. video camera) is used for both pose estimation and capturing the user's hand image. Therefore, the calibration problem between head tracking and hand tracking is fundamentally eliminated.

Strictly speaking, however, the right-hand side of eq.(2) is ${}^{camera}\boldsymbol{T}_{hand}$ with this technique. If the cameras are located far from the user's eyes, eq.(2) is not true. Therefore, the camera(s) should be mounted as close to the user's eyes as possible so that the user will not feel any sense of incompatibility with the real life image[16].

One limitation of this method is that the extracted user's hand image is always displayed on top of the virtual scene without knowing the proper spatial relationship between the user's hand and the virtual object. If we could implement the real-time stereo developed by Kanade, the above problem could be solved by introducing the "Z-keying"[11] or "depth-keying" technique instead of Chroma Keying. Knowing the depth of the user's hand in the camera image also means knowing the 3D position of the user's hand. Therefore, this depth information could also be used for hand tracking in the surface display mode. The depth information also enables us to apply an image warping technique[4] or texture mapping on a polygonal hand model (if the computation is fast enough) for compensating effects of the camera offset problem discussed above.

3.3. Other visual/haptic interface systems

So far, several systems have been developed aimed at WYSIWYF. Iwata et al.[10] have developed a desktop type haptic device combining a visual interface. They put a mirror between the haptic device and the user so that the user can see the reflected image generated by the graphics workstation. The user's hand image is synthetic, rendered based on the joint sensor information of the haptic device. With precise setup, WYSIWYF could be realized but the user's head must be fixed at a certain position.

Sato et al.[18] have developed a micro telemanipulation system. A CRT display is built in a table top and the user can directly point to a magnified image of the object by using a force-reflecting pen. The user's hand motion and the position/orientation of the pen are captured by cameras. Although this system is limited to a two-dimensional workspace, WYSIWYF is well realized.

McNeely's group has developed a virtual control panel simulation system[7]. Tracking an infrared LED attached

at the user's fingertip by two CCD cameras, several types of switches and buttons attached at the tip of manipulator will be placed at appropriate locations beforehand so that the user's fingertip can encounter each of them in the virtual environment. The user wears a head-mounted display and his/her head motion is tracked by a magnetic sensor. The virtual control panel image is rendered based on the tracked user's head motion, and a synthetic user's hand image is rendered based on the tracked infrared LED location. What the user sees is a perfect virtual scene, but potential inconsistency between the fingertip tracking and the head tracking (delay or offset) may break the correct WYSIWYF situation.

4. Physically-based simulation for haptic rendering

4.1. Rigid body dynamics and non-penetration constraints

In the interactive computer graphics field, Baraff[2] has proposed a physically-based dynamic simulation method for non-penetrating rigid bodies. We applied his algorithm to haptic rendering. The advantages of applying his approach to haptic rendering are (i) in principle, a rigid body can be rendered, and (ii) in practice, the time step size of the simulation is not critical even for simulating rigid bodies.

Baraff's approach is basically solving ODEs (ordinary differential equations) considering the constraint condition. Therefore, it is consistent with an admittance display (measuring force and displaying motion). If the object is unconstrained, solving ODEs is straightforward. If the object is under non-penetration constraints, however, two types of contact, (i) colliding contact and (ii) resting contact[22], may occur and we have to solve unknown contact forces or impulses before solving the ODEs. In this subsection, we discuss the resting contact case. In the next subsection, the colliding contact case will be considered.

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 is defined in such a way 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:

$$\ddot{d}_i = a_{i1}f_1 + a_{i2}f_2 + \cdots + a_{in}f_n + b_i \qquad (7)$$

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. b_i is the term containing coriolis and centrifugal forces and the external force.

For realizing nonpenetrating rigid body motion, the following conditions should be satisfied:

$$\ddot{d}_i \geq 0, f_i \geq 0 \text{ and } f_i \cdot \ddot{d}_i = 0 \qquad (8)$$

Putting eqs.(7) and (8) together for all n contact points, we get

$$\boldsymbol{A}\boldsymbol{f} + \boldsymbol{b} \geq 0 \qquad (9)$$

$$\boldsymbol{f} \geq 0 \text{ and } \boldsymbol{f}^T(\boldsymbol{A}\boldsymbol{f} + \boldsymbol{b}) = 0 \qquad (10)$$

The problem is to find f_i's which satisfy eqs.(9) and (10). This problem can be regarded as an optimization problem such as linear complementarity programming or quadratic programming. But solving such an optimization problem may require much computational effort and might not be adequate for the purpose of interactive simulation. Baraff[2] has proposed a fast algorithm for computing contact forces which is a kind of iterative method which pivots matrix $\boldsymbol{A}$. In the frictionless case, his algorithm is guaranteed to converge to the correct solution. In the case of friction, his algorithm also works well in practice.

4.2. Colliding contact and 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 the collision and before the collision at the i-th colliding point respectively. v_i^+ can be expressed by the following equation:

$$v_i^+ = v_i^- + a_{i1}j_1 + a_{i2}j_2 + \cdots + a_{im}j_m \qquad (11)$$

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 post-collision velocity. Newton's law of restitution says

$$v_i^+ + \epsilon_i v_i^- \geq 0 \qquad (12)$$

where ϵ_i denotes the coefficient of restitution at the i-th colliding point. The reason why we put "$\geq$" in eq.(12) 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 of other colliding points.

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

$$j_i \geq 0 \text{ and } j_i \cdot (v_i^+ + \epsilon_i v_i^-) = 0 \qquad (13)$$

Substituting eq.(11) to eq.(12), we get

$$a_{i1}j_1 + a_{i2}j_2 + \cdots + a_{im}j_m + v_i^- + \epsilon_i v_i^- \geq 0 \qquad (14)$$

Putting eqs.(14) and (13) together for all m colliding points, we get

$$\boldsymbol{A}\boldsymbol{j} + \boldsymbol{c} \geq 0 \qquad (15)$$

$$\boldsymbol{j} \geq 0 \text{ and } \boldsymbol{j}^T(\boldsymbol{A}\boldsymbol{j} + \boldsymbol{c}) = 0 \qquad (16)$$

The problem is to find j_i's which satisfy eqs.(15) and (16). Note that eqs.(15) and (16) have the same form as that of eqs.(9) and (10). Therefore, we can use the same algorithm used for finding contact forces to find these impulses. Once j_i's have been obtained, we can get object velocities after the collision and reset the state variables and restart solving the ODEs. For more details of the algorithm, see [2][22].

4.3. Simulation algorithm

At every simulation cycle, the following steps are performed:

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

STEP 2: If any resting contact points were found in **STEP 4** in the previous simulation cycle, compute constraint forces which satisfy eqs.(9) and (10).

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 colliding contact points and resting contact points for new state variables.

STEP 5: If any colliding contact points were found in **STEP 4**, compute impulses which satisfy eqs.(15) and (16). Otherwise go to **STEP 7**.

STEP 6: Compute object velocities 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 either position, velocity or acceleration depending on what kind of controller is implemented to the haptic device.

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

5. First WYSIWYF display prototype

5.1. System configuration

Figure 5 illustrates the prototype system configuration. Although a head-mounted camera/display would be ideal for WYSIWYF, we decided to use an existing LCD panel for our first prototype. A color CCD camera was attached at

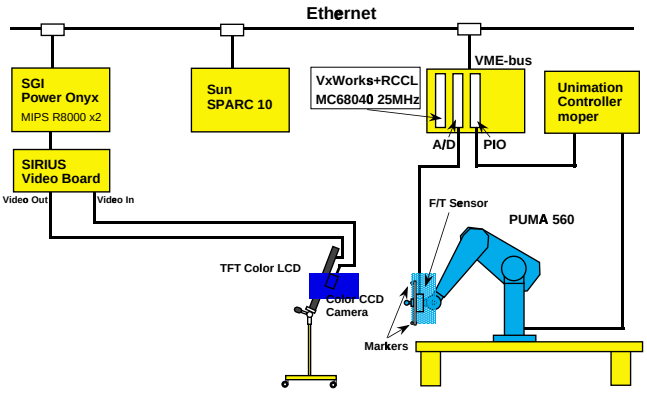


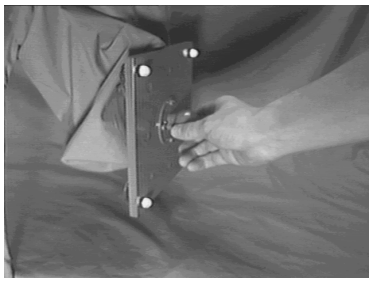
Figure 5. Prototype system configuration

the back plane of the LCD panel. The LCD/camera system is mounted on a movable platform so that the user can move it around to change his/her viewpoint.

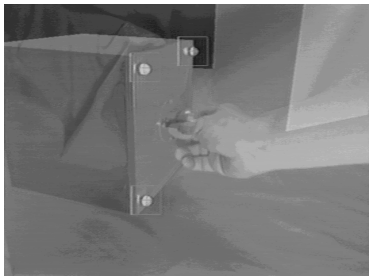
Pose estimation and rendering the virtual scene are performed by a SGI PowerOnyx with an optional SIRIUS Video Board. A PUMA 560, 6 DOF industrial robot, is used for the haptic device. We put an aluminum plate with four markers, small incandescent lamps covered by translucent lenses, at the tip of the PUMA for tracking. We implemented the pose estimation algorithm based on the extended Kalman filter[6]. We first took the relative pose between the camera and the plate as the state variables of the estimator. The motion of the haptic device, however, affects the state estimator and the background image tends to be shaky even when the camera/display system is stationary. Since we know exactly the motion of the haptic device, we can exclude it from the state variables. After this treatment, the background image became stable.

The SIRIUS Video Board has a built-in video keying circuitry. A somewhat disappointing design specification of the SIRIUS Video, however, prevents us from taking the video image directly into the memory for marker tracking while using the video-out port. Alternatively one has to display the video image on the screen first and take it indirectly into the memory. Unfortunately this solution conflicts with the video input path to the video keying circuitry, meaning that one has to do the chroma keying by software. We then introduced two modes, camera fixed mode and tracking mode. Chroma keying must be done by software in tracking mode, while in camera fixed mode chroma keying can be done by the built-in circuitry. The estimated frame rates are about 5 Hz in tracking mode for four markers with software chroma keying, 10 Hz without chroma keying, and 50 Hz in camera fixed mode with hardware chroma keying.

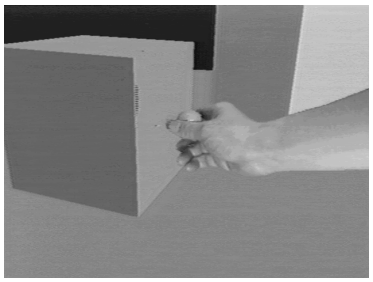
Physically-based simulation is performed on a VME-bus-based MC68040 CPU board (Motorola MVME162) with the VxWorks real-time OS. RCCL/RCI[13], real-time C li-



(a) Original video scene



(b) Overlaid image



(c) Final blended image

Figure 6. Results in tracking mode

braries for controlling the PUMA, have been installed on our VxWorks system. A SPARC 10 workstation is used for the VxWorks and RCCL host machine. A JR3 six-axis force/torque sensor is attached to the PUMA. The Unimation controller and the VxWorks system are connected by a parallel cable.

In every simulation cycle, the steps described in section 4.3 are carried out. Since the Unimation controller is position servo-based, the simulation module gives the current position/orientation of the virtual object to the RCCL/RCI module as a setpoint. RCCL/RCI then interpolates these points and generates a smooth trajectory. The generated trajectory data are sent to the Unimation controller via parallel lines. The above process is carried out every 20 msec and the lowest joint level servo loop in the Unimation controller is running at 1000 Hz. Current $\boldsymbol{T}_6$ data are sent to the PowerOnyx via socket communication.

The working environment was covered by blue cloth. The forearm and wrist portions of the PUMA were wrapped up by blue cloth as well (see Fig.8).

5.2. Experimental result

A simple frictionless virtual environment was built, where a 20 cm$\times$20 cm$\times$20 cm cube is sitting on top of a flat table. Figure 6 (a) is an original video image. Figure 6 (b) show the registration result in tracking mode. Small square windows in the image indicate searching windows for markers. Although one of the four markers is occluded by the user's hand, the system can continue the pose estimation. Figure 6 (c) is a final blended image.

In this system, the sensor knob is the only permitted portion of the haptic device for the user to access, which corresponds to the knob attached to the virtual cube. The plate supporting the markers, however, happens to be the same size as one face of the cube, and the user can feel the face and the surrounding four edges as well. In addition, the user's hand image may be occluded correctly by the virtual object in the final blended image as shown in Fig.7.

Although the PUMA is controlled by conventional high-gain position servos and we are updating the setpoint every 20 msec, which is a relatively slow rate, the system can keep stable and can render reasonably convincing haptic sensations. The user can manipulate the virtual cube quite realistically, feeling inertial force, constrained forces and colliding impulses.

Figure 8 shows a scene with the user using the prototype system. One problem of our current system is the location of the camera. Since the camera is located behind the LCD panel, far from the user's eyes, the size of the hand image captured by the camera is different from the actual apparent size that would be seen from the user's eyes. As we discussed in section 3.2, the camera should be located as close to the user's eyes as possible. Making the user's eyes close to the camera, however, means making his/her eyes close to the display, which makes it difficult for the user to see the display. This problem could be solved by introducing a head-mounted camera/display system in the future.

The second problem is low frame rate and latency in the tracking mode. Taking the live video image indirectly into the memory through the screen takes much time as well as software chroma keying, and latency is very large (about 0.5 sec in worst case). In the camera fixed mode with hardware chroma keying, there is no noticeable latency. We are planning to implement a special tracking board manufactured by FUJITSU Ltd.

6. Conclusions

This paper has proposed a new concept of a visual/haptic interface device, namely a WYSIWYF display, which ensures correct visual/haptic registration. There are three key components: (1) encountered type haptic rendering with the physically-based simulation, (2) vision-based tracking for

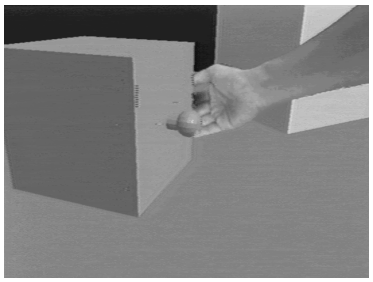


Figure 7. Correct occlusion of the user's hand

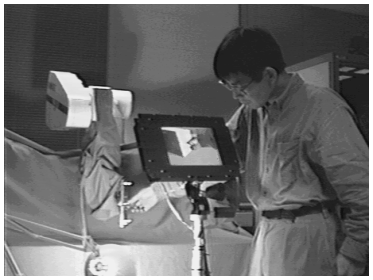


Figure 8. System overview in use

pose estimation, and (3) superimposing the user's live hand image with video-keying.

A simple virtual cube manipulation was carried out by a prototype system. Although our current prototype uses a LCD panel, not realizing a perfect WYSIWYF, the system has shown the potential superiority of the proposed WYSIWYF concept over conventional approaches.

Acknowledgment

The authors would like to express their grateful thanks to Mr. Michihiro Uenohara, visiting research scientist of CMU from TOSHIBA Corporation, Japan, for his comment on vision-based tracking and his help for making the LCD/camera system. Dr. David Baraff, assistant professor of CMU, gave them many valuable comments on physically-based simulation. Dr. John Lloyd helped them for implementing RCCL/RCI to their system. Mike Blackwell, senior research engineer of CMU, kindly maintained the SGI PowerOnyx and installed the SIRIUS Video. Finally, they would express their appreciation to SHARP CORPORATION for providing a LCD panel for them.

References

[1] M. Bajura et al., "Merging Virtual Objects with the Real World", In *Proc., SIGGRAPH'92* (Computer Graphics, vol.26), pp.203–210 (1992)

[2] D. Baraff, "Fast Contact Force Computation for Nonpenetrating Rigid Bodies", In *Proc., SIGGRAPH'94*, pp.23–34 (1994)

[3] G. Bishop, "Self-Tracker: A Smart Optical Sensor on Silicon", Ph.D. Thesis, Univ. of North Carolina at Chapel Hill (1984)

[4] S. Chen and L. Williams, "View Interpolation for Image Synthesis", In *Proc., SIGGRAPH'93*, pp.279–288 (1993)

[5] J. E. Colgate and J.M.Brown, "Factors Affecting the Z-Width of a Haptic Display", In *Proc, 1994 IEEE Int. Conf. on Robotics and Automation*, pp.3275–3210 (1994)

[6] D. B. Gennery, "Visual Tracking of Known Three-Dimensional Objects", Int. J. of Computer Vision, vol.7, no.3, pp.243–270 (1992)

[7] P. E. Gruenbaum et al., "Implementation of Robotic Graphics For a Virtual Control Panel", In *VRAIS-95 Video Proc.* (1995)

[8] M. Hammerton and A. H. Tickner, "Transfer of training between space-oriented and body-oriented control situations", British Journal of Psychology, vol.55, no.4, pp.433–437 (1964)

[9] K. Hirota and M. Hirose, "Simulation and Presentation of Curved Surface in Virtual Reality Environment Through Surface Display", In *Proc., VRAIS'95*, pp.211–216 (1995)

[10] H. Iwata, "Artificial Reality with Force-Feedback: Development of Desktop Virtual Space with Compact Master Manipulator", Computer Graphics, vol.24, no.4, pp.165–170 (1990)

[11] T. Kanade, "A Stereo Machine for Video-Rate Dense Depth Mapping and Its New Applications", In *Proc., Image Understanding Workshop*, ARPA, February (1996) (to appear)

[12] T. Kotoku, K. Komoriya and K. Tanie, "A Force Display System for Virtual Environments", In *Proc., IEEE Int. Workshop on Robot and Human Communication*, Tokyo, Japan, 1-3 September (1992)

[13] J. Lloyd and V. Hayward, "Multi-RCCL User's Guide", McGill University (1992)

[14] W. A. McNeely, "Robotic Graphics: A New Approach to Force Feedback for Virtual Reality", In *Proc., VRAIS'93*, pp.336–341 (1993)

[15] P. J. Metzger, "Adding Reality to The Virtual", In *Proc., VRAIS'93*, pp.7–13 (1993)

[16] J. P. Rolland et al., "Quantification of Adaptation to Virtual-Eye Location In See-Thru Head-Mounted Displays", In *Proc., VRAIS'95*, pp.56–66 (1995)

[17] S. E. Salcudean and T. D. Vlaar, "On the Emulation of Stiff Walls and Static Friction with a Magnetically Levitated Input/Output Device" In *Proc., International Mechanical Engineering Congress and Exposition*, Chicago, November, 1994, pp.303–309 (1994)

[18] T. Sato et al., "Micro Teleoperation System Concentrating Visual and Force Information at Operator's Hand" In *Proc., 3rd Int. Symp. on Experimental Robotics (ISER)*, Kyoto, October, 1993, pp.118–124 (1993)

[19] S. Tachi et al., "A Machine that Generates Virtual Haptic Space", In *VRAIS-95 Video Proc.* (1995)

[20] M. Uenohara and T. Kanade, "Vision-Based Object Registration for Real-Time Image Overlay", Computers in Biology and Medicine, vol.25, no.2, pp.249–260 (1995)

[21] M. Ward et al., " A Demonstrated Optical Tracker With Scalable Work Area for Head-Mounted Display System", In *Proc., 1992 Symp. on Interactive 3D Graphics*, pp.43–52 (1992)

[22] A. Witkin et al., "An Introduction to Physically Based Modeling", SIGGRAPH'94 Course Notes (1994)