The Perceptive Workbench: Toward Spontaneous and Natural Interaction in Semi-Immersive Virtual Environments

Bastian Leibe, Thad Starner, William Ribarsky, Zachary Wartell, David Krum, Brad Singletary, and Larry Hodges

GVU Center, Georgia Institute of Technology

Abstract

The Perceptive Workbench enables a spontaneous, natural, and unimpeded interface between the physical and virtual worlds. It uses vision-based methods for interaction that eliminate the need for wired input devices and wired tracking. Objects are recognized and tracked when placed on the display surface. Through the use of multiple light sources, the object's 3D shape can be captured and inserted into the virtual interface. This ability permits spontaneity since either preloaded objects or those objects selected on the spot by the user can become physical icons. Integrated into the same vision-based interface is the ability to identify 3D hand position, pointing direction, and sweeping arm gestures. Such gestures can enhance selection, manipulation, and navigation tasks. In this paper, the Perceptive Workbench is used for augmented reality gaming and terrain navigation applications, which demonstrate the utility and capability of the interface.

1. Introduction

Until now, we have interacted with computers mostly by using devices that are constrained by wires. Typically, the wires limit the distance of movement and inhibit freedom of orientation. In addition, most interactions are indirect. The user moves a device as an analogue for the action to be created in the display space. We envision an interface without these restrictions. It is untethered; accepts direct, natural gestures; and is capable of spontaneously accepting as interactors any objects we choose.

In conventional 3D interaction, the devices that track position and orientation are still usually tethered to the machine by wires. Devices, such as pinch gloves, that permit the user to experience a more natural-seeming interface often do not perform as well, and are less preferred with users, than simple handheld devices with buttons [9,19]. Pinch gloves carry assumptions about the position of the user's hand and fingers with respect to the tracker. Of course, users' hands differ in size and shape, so the assumed tracker position must be recalibrated for each user. This is hardly ever done. Also, the glove interface causes subtle changes to recognized hand gestures. The result is that fine manipulations can be imprecise, and the user comes away with the feeling that the interaction is slightly off in an indeterminate way. If we can recognize gestures directly, we take into account the difference in hand sizes and shapes.

An additional problem is that any device held in the hand can become awkward while gesturing. We have found this even with a simple pointing device, such as a stick with a few buttons [19]. Also a user, unless fairly skilled, often has to pause to identify and select buttons on the stick. With accurately tracked hands most of this awkwardness disappears. We are adept at pointing in almost any direction and can quickly pinch fingers, for example, without looking at them.

Finally, physical objects are often natural interactors (such as phicons [26]). However, with current systems these objects must be inserted in advance or specially prepared. One would like the system to accept objects that one chooses spontaneously for interaction.

In this paper we discuss methods for producing more seamless interaction between the physical and virtual environments through the creation of the Perceptive Workbench. The system is then applied to an augmented reality game and a terrain navigating system. The Perceptive Workbench can reconstruct 3D virtual representations of previously unseen real-world objects placed on its surface. In addition, the Perceptive Workbench identifies and tracks such objects as they are manipulated on the desk's surface and allows the user to interact with the augmented environment through 2D and 3D gestures. These gestures can be made on the plane of the desk's surface or in the 3D space above the desk. Taking its cue from the user's actions, the Perceptive Workbench switches between these modes automatically, and all interaction is controlled through computer vision, freeing the user from the wires of traditional sensing techniques.

2. Related Work

While unique, the Perceptive Workbench has a rich heritage of related work [1, 3, 5, 10, 11, 12, 15, 16, 18, 19, 26, 27, 28, 31]. Many augmented desk and virtual reality designs use tethered props, tracked by electromechanical or ultrasonic means, to encourage interaction through manipulation and gesture [2, 3, 5, 18, 19, 23, 28]. Such designs tether the user to the desk and require the time-consuming ritual of donning and doffing the appropriate equipment. However, the Perceptive Workbench leverages techniques from the computer vision community [21, 32] to maintain a wireless interface.

Most directly related to the Perceptive Workbench, the

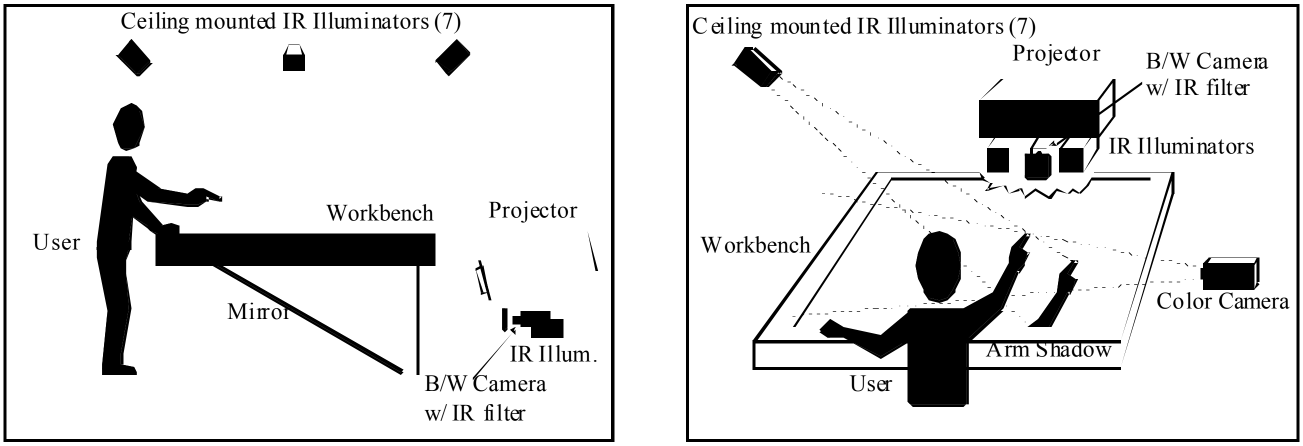

Figure 1 Light and camera positions for the Perceptive Workbench.
The top view shows how shadows are cast and the 3D arm position is tracked.

"Metadesk" [26] identifies and tracks objects placed on the desk's display using a near-infrared computer vision recognizer, originally designed by this paper's second author. Rekimoto and Matsushita's "Perceptual Surfaces" [16] employ 2D barcodes to identify objects held against their "HoloWall" and "HoloTable." In addition, the HoloWall can track the user's hands (or other body parts) near or pressed against its surface. Davis and Bobick's SIDEshow [7] is similar to the Holowall except that it uses cast shadows in infrared for full-body 2D gesture recovery. Some augmented desks have cameras and projectors above the surface of the desk and are designed to augment the process of handling paper or interacting with models and widgets through the use of fiducials [1, 10, 27, 31]. Krueger's VIDEODESK [11] used an overhead camera and a horizontal visible light table (for high contrast) to provide hand gesture input for interactions displayed on a monitor on the far side of the desk. In contrast with the Perceptive Workbench, none of these systems address the issues of introducing spontaneous 3D physical objects into the virtual environment in real-time and combining 3D deictic (pointing) gestures with object tracking and identification.

3. Apparatus

The display environment for the Perceptive Workbench is based on Fakespace's immersive workbench, consisting of a wooden desk with a horizontal frosted glass surface, on which a stereoscopic image can be projected from behind the Workbench.

We placed a standard monochrome surveillance camera under the projector that watched the desk surface from underneath (see Figure 1). A filter placed in front of the camera lens makes it impervious to visible light and to images projected on the desk's surface. Two infrared illuminators placed next to the camera flood the surface of the desk with infrared light that is reflected toward the camera by objects placed on the desk's surface. A ring of seven similar light-sources is mounted on the ceiling surrounding the desk. Each computer-controlled light casts distinct shadows on the desk's surface based on the objects on the table (Figure 2a). A second camera, this one in color, is placed next to the desk to provide a side view of the user's arms. This side camera is used solely for recovering 3D pointing gestures. All vision processing is done on two SGI R10000 O2s (one for each camera), which communicate with a display client on an SGI Onyx via sockets.

For display on the Perceptive Workbench, we use the Simple Virtual Environment Toolkit (SVE), a graphics and sound library developed by the Georgia Tech Virtual Environments Group [9]. In addition, we use the workbench version of VGIS, a global terrain visualization and navigation system [13, 14] as an application for interaction using hand and arm gestures.

4. Object Recognition and Tracking

As a basic building block for our interaction framework, we want to enable the user to manipulate the virtual environment by placing objects on the desk surface. The system should recognize these objects and track their positions and orientations while they are being moved over the table. The user should be free to pick any set of physical objects he wants to use.

To achieve this goal, we use an improved version of the technique described in Starner *et al.* [22]. The underside of the desk is illuminated by two near-infrared light-sources (Figure 1). Every object close to the desk surface (including the user's hands) reflects this light and can be seen by the camera under the display surface (Figures 1 and 2b). Using a combination of intensity thresholding and background subtraction, we extract interesting regions of the camera image and analyze them. The resulting blobs are classified as different object types based on a set of features, including area, eccentricity, perimeter, moments, and the contour shape.

There are several complications due to the hardware arrangement. The foremost problem is that our two light sources can only provide very uneven lighting over the whole desk surface, bright in the middle and getting weaker toward the borders. In addition, the light rays are

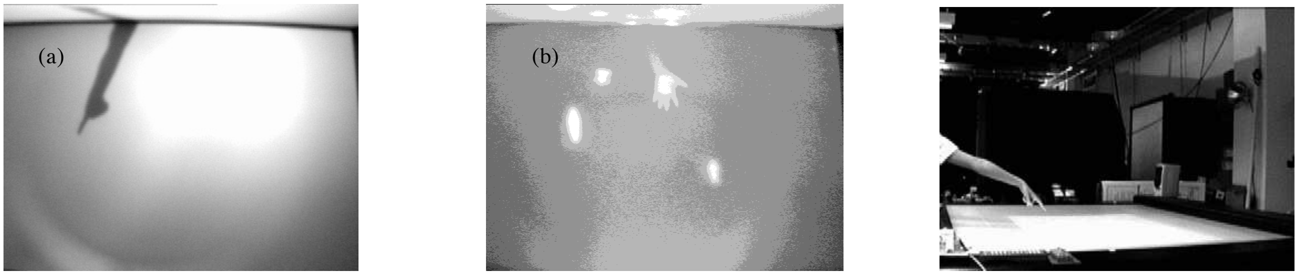

Figure 2 Images seen by the infrared and color cameras: (a) arm shadow from overhead IR lights; (b) reflections from IR lights underneath the desk; (c) image from side camera.

not parallel, and the reflection on the mirror surface further exacerbates this effect. As a result, the perceived sizes and shapes of objects on the desk surface can vary depending on position and orientation. Finally, when the user moves an object, the reflection from his hand can also add to the perceived shape. This makes it necessary to use an additional stage in the recognition process that matches recognized objects to objects known to be on the table and can filter out improper classification or even complete loss of information about an object for several frames.

In this work, we are using the object recognition and tracking capability mainly for "cursor objects." Our focus is on fast and accurate position tracking, but the system may be trained on a different set of objects to be used as navigational tools or physical icons [26]. A future project will explore different modes of interaction based on this technology.

5. 3D Reconstruction

Several methods have been designed to reconstruct objects from silhouettes [20, 24] or dynamic shadows [6], using either a moving camera, a moving light source, or a turntable for the object [24]. Several systems have been developed for the reconstruction of relatively simple objects, including the commercial system Sphinx3D.

However, the necessity of moving either the camera or the object imposes severe constraints on the working environment. To reconstruct an object with these methods, it is usually necessary to interrupt the user's interaction with it, take the object out of the user's environment, and place it into a specialized setting. Other approaches make use of multiple cameras from different viewpoints to avoid this problem at the expense of more computational power to process and communicate the results. In this project, using only one camera and the infrared light sources, we analyze the shadows cast on the object from multiple directions. As the process is based on infrared light, it can be applied independent of the lighting conditions and without interfering with the user's natural interaction with the desk or the current visual display environment.

Our approach is fully automated and does not require any special hardware (*e.g.* stereo cameras, laser range finders, structured lighting, *etc.*). Our method is extremely inexpensive, both in hardware and in computational cost. In addition, there is no need for extensive calibration, which is usually necessary in other approaches to recover the exact position or orientation of the object in relation to the camera. We only need to know the approximate position of the light sources (+/- 2 cm), and we need to adjust the camera to reflect the size of the display surface, which must be done only once. Neither the camera, light-sources, nor the object are moved during the reconstruction process. Thus recalibration is unnecessary. We have substituted all mechanical moving parts, which are often prone to wear and imprecision, by a series of light beams from known locations.

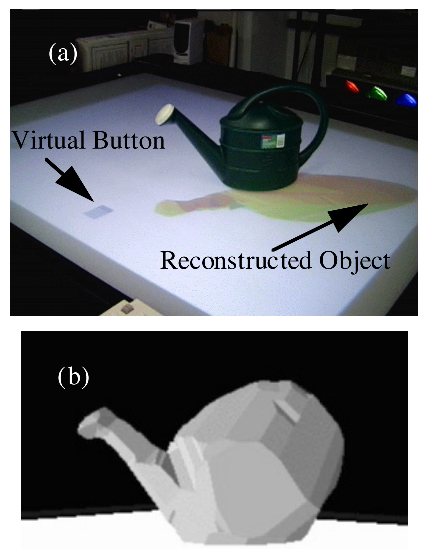

Figure 3 (a) 3D reconstruction of an object placed on the workbench display; (b) resulting polygonal object.

An obvious limitation for this approach is that we are confined to a fixed number of different views from which to reconstruct the object. The turntable approach, on the other hand, allows the system to take an arbitrary number of images from different view points. However, Sullivan's work [24] and our experience with our system have shown that even for quite complex objects, usually seven to nine different views are enough to get a reasonable 3D model of the object. Thus, to obtain the different views, we mounted a ring of seven infrared light sources in the ceiling, each one of which is switched independently by computer control. The system detects when a new object is placed on the desk surface, and the user can initiate the reconstruction by touching a virtual button rendered on the screen (Figure 3a). This action is detected by the camera, and, after only one second, all shadow images are taken. In another second, the reconstruction is complete (Figure 3b), and the newly reconstructed object is part of the virtual world.

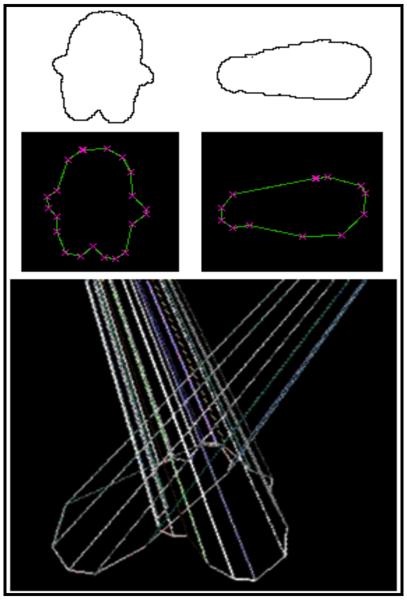

Figure 4 Steps of 3D object reconstruction including extraction of contour shapes from shadows and multiple view cones (bottom).

The speed of the reconstruction process is mainly limited by the switching time of the light sources. Whenever a new light-source is activated, the image processing system has to wait for several frames to receive a valid image. The camera under the desk records the sequence of shadows cast by an object on the table when illuminated by the different lights. Figure 4 shows a model reconstructed from a series of contour shadows, where the contour shadows are extracted by using different IR sources. By approximating each shadow as a polygon (not necessarily convex) [17], we create a set of polyhedral "view cones," extending from the light source to the polygons. The intersecting of these cones creates a polyhedron that roughly contains the object. Figure 3b shows the polygons resulting from the previous shadows and a visualization of the intersection of polyhedral cones.

6. Deictic Gesture Recognition and Tracking

Hand gestures for interaction with a virtual environment can be roughly classified into symbolic (iconic, metaphoric, and beat) and deictic (pointing) gestures. Deictic gestures are characterized by a strong dependency on location and the orientation of the gesturing hand. Their meaning is determined by the position at which a finger is pointing, or by the angle of rotation of some part of the hand. This information acts not only as a symbol for the gesture's interpretation, but also as a measure of by how much the corresponding action should be executed or to which object it should be applied.

For navigation and object manipulation in a virtual environment, many gestures are likely to have a deictic component. It is usually not enough to recognize that an object should be rotated, but we will also need to know the desired amount of rotation. For object selection or translation, we want to specify the object or location of our choice just by pointing at it. For these cases, gesture recognition methods that only take the hand shape and trajectory into account will not be sufficient. We need to recover 3D information about the user's hand and arm in relation to his body.

With vision-based 3D tracking techniques, the first issue is to determine which information in the camera image is relevant, *i.e.* which regions represent the user's hand or arm. This task is made even more difficult by variations in users' clothing or skin color and by background activity.

In a virtual workbench environment, there are few places where a camera can be placed to provide reliable hand position information. One camera can be set up next to the table without overly restricting the available space for users, but if a similar second camera were to be used at this location, either multi-user experience or accuracy would be compromised. We have addressed this problem by employing our shadow-based architecture (as described in the hardware section). The user stands in front of the workbench and extends an arm over the surface. One of the IR light-sources mounted on the ceiling to the left of, and slightly behind the user, shines its light on the desk surface, from where it can be seen by the IR camera under the projector (see Figure 1). When the user moves his arm over the desk, it casts a shadow on the desk surface (see Figure 2a). From this shadow, and from the known light source position, we can calculate the plane in which the user's arm must lie.

Simultaneously, the second camera to the right of the table (Figure 1 and 2c) records a side view of the desk surface and the user's arm. It detects where the arm enters the image and the position of the fingertip. From this information, it extrapolates two lines in 3D space, on which the observed real-world points must lie. By intersecting these lines with the shadow plane, we get the coordinates of two 3D points, one on the upper arm, and one on the fingertip. This gives us the user's hand position, and the direction in which the user is pointing. As shown in Figure 5, this information can be used to project a hand position icon and a selection ray in the workbench display.

We must first recover arm direction and fingertip position from both the camera and the shadow image.

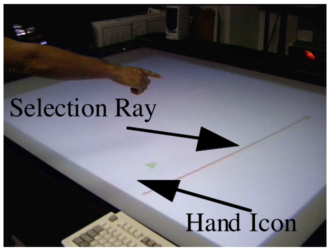

Figure 5 Pointing gesture with hand icon and selection ray.

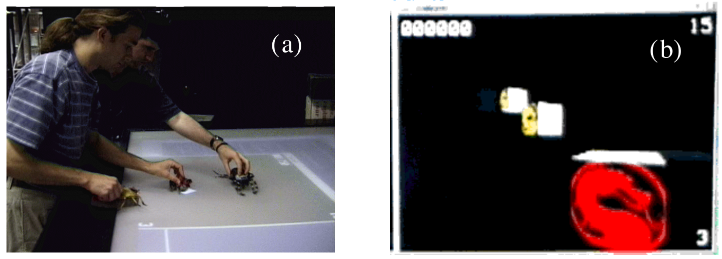

Figure 6 (a) Game masters controlling monster positions; (b) monsters moving in the 3D space as a result of actions in Figure 6a.

Since the user is standing in front of the desk and the user's arm is connected to the user's body, the shadow of the arm should always touch the image border. Thus our algorithm exploits intensity thresholding and background subtraction to discover regions of change in the image and searches for areas in which these touch the front border of the desk surface. It then takes the middle of the touching area as an approximation for the origin of the arm (Figure 2a). For simplicity we will call this point the "shoulder," although in most cases it is not. Tracing the contour of the shadow, the algorithm searches for the point that is farthest away from the shoulder and takes it as the fingertip. The line from the shoulder to the fingertip reveals the 2D direction of the arm.

In our experiments, the point thus obtained was coincident with the pointing fingertip in all but a few extreme cases (such as the fingertip pointing straight down at a right angle to the arm). The method does not depend on a pointing gesture and also works for most other hand shapes, including but not restricted to, a hand held horizontally, vertically, or in a fist. These shapes may be distinguished by analyzing a small section of the side camera image and may be used to trigger specific gesture modes in the future.

The computed arm direction is correct as long as the user's arm is not overly bent. In such cases, the algorithm still connected shoulder and fingertip, resulting in a direction somewhere between the direction of the arm and the one given by the hand. Although the absolute resulting pointing position did not match the position towards which the finger was pointing, it still managed to capture the trend of movement very well. Surprisingly, the technique is sensitive enough such that the user can stand at the desk with his arm extended over the surface and direct the pointer simply by moving his index finger, without arm movement.

Limitations

The architecture used poses several limitations. The primary problem with the shadow approach is finding a position for the light source that can give a good shadow of the user's arm for a large set of possible positions, while avoiding capture of the shadow from the user's body. Since the area visible to the IR camera is coincident with the desk surface, there are necessarily regions where the shadow is not visible in, touches, or falls outside of the borders. Our solution to this problem is to switch to a different light source whenever such a situation is detected, the choice of the new light source depending on where the shadows touched the border. By choosing overlapping regions for all light-sources, we can keep the number of light-source switches to a necessary minimum. In practice, four light sources were enough to cover the relevant area of the desk surface.

A bigger problem is caused by the location of the side camera. If the user extends both of his arms over the desk surface, or if more than one user tries to interact with the environment at the same time, the images of these multiple limbs can overlap and be merged to a single blob. As a consequence, our approach will fail to detect the hand positions and orientations in these cases. A more sophisticated approach using previous position and movement information could yield more reliable results, but we chose, at this first stage, to accept this restriction and concentrate on high frame rate support for one-handed interaction. This may not be a serious limitation for a single user for certain tasks; a recent study shows that for tasks normally requiring two hands in a real environment, users have no preference for one versus two hands in a virtual environment that does not model gravity [19].

7. Performance Analysis

Both object and gesture tracking perform at a stable 12-18 frames per second. Frame rate depends on the number of objects on the table and the size of the shadows, respectively. Both techniques are able to follow fast motions and complicated trajectories. Latency is currently 0.25-0.33 seconds but has improved since last testing (the acceptable threshold is considered to be at around 0.1 second). Surprisingly, this level of latency seems adequate for most pointing gestures. Since the user is provided with continuous feedback about his hand and pointing position, and most navigation controls are relative rather than absolute, the user adapts his behavior readily to the system. With object tracking, the physical object itself can provide the user with adequate tactile feedback as the system catches up with the user's manipulations. In general, since the user is moving objects across a very large desk surface, the lag is noticeable but rarely troublesome in the current applications.

Even so, we expect simple improvements in the socket communication between the vision and rendering code and in the vision code itself to improve latency significantly. For the terrain navigation task below, rendering speed provides a limiting factor. However, Kalman filters may compensate for render lag and will also add to the stability of the tracking system.

Calculating the error from the 3D reconstruction process requires choosing known 3D models, performing the reconstruction process, aligning the reconstructed model and the ideal model, and calculating an error measure. For simplicity, a cone and pyramid were chosen. The centers of mass of the ideal and reconstructed models were set to the same point in space, and their principal axes were aligned.

To measure error, we used the Metro tool [4]. It approximates the real distance between the two surfaces by choosing a set of (100,000-200,000) points on the reconstructed surface, and then calculating the two-sided distance (Hausdorff distance) between each of these points and the ideal surface. This distance is defined as

$$\max(E(S1,S2),E(S2,S1))$$

with E(S1,S2) denoting the one-sided distance between the surfaces S1 and S2:

$$E(S1,S2) = \max_{p\in S1}(dist(p,S2))$$

The Hausdorff distance directly corresponds to the reconstruction error. In addition to the maximum distance, we also calculated the mean and mean square distances. Table 1 shows the results. In these examples, the relatively large maximal error was caused by the difficulty in accurately reconstructing the tip of the cone and the pyramid.

	Cone	Pyramid
Maximal Error	0.0215 (7.26%)	0.0228 (6.90%)
Mean Error	0.0056 (1.87%)	0.0043 (1.30%)
Mean Square Error	0.0084 (2.61%)	0.0065 (1.95%)

Table 1: reconstruction errors averaged over three runs (in meters and percentage of object diameter)

While improvements may be made by precisely calibrating the camera and lighting system, by adding more light sources, and by obtaining a silhouette from the side camera (to eliminate ambiguity about the top of the surface), the system meets its goal of providing virtual presences for physical objects in a quick and timely manner that encourages spontaneous interactions.

8. Putting It to Use: Spontaneous Gesture Interfaces

The Perceptive Workbench interface can switch automatically between gesture recognition and object recognition, tracking, and reconstruction. When the user moves his hand above the display surface, the hand and arm are tracked as described in Section 6. A cursor appears at the projected hand position on the display surface and a ray emanates along the projected arm axis. These can be used in selection or manipulation, as in Figure 5. When the user places an object on the surface, the cameras recognize this and identify and track the object. A virtual button also appears on the display (indicated by the arrow in Figure 3a). Through shadow tracking, the system determines when the hand overlaps the button, selecting it. This action causes the system to capture the 3D object shape, as described in Section 5.

This set provides the elements of a perceptual interface, operating without wires and without restrictions as to objects employed. For example, we have constructed a simple application where objects placed on the desk are selected, reconstructed, and then placed in a "template" set, displayed as slowly rotating objects on the left border of the workbench display. These objects could act as new physical icons that are attached by the user to selection or manipulation modes. Or the shapes themselves could be used in model-building or other applications.

An Augmented Reality Game

We have created a more elaborate collaborative interface using the Perceptive Workbench. This involves the workbench communicating with a person in a separate space wearing an augmented reality headset. All interaction is via image-based gesture tracking without attached sensors. The game is patterned after a martial arts fighting game. The user in the augmented reality headset is the player, and one or more people interacting with the workbench are the game masters. The workbench display surface acts as a top-down view of the player's space. The game masters place different objects on the surface, which appear to the player as distinct monsters at different vertical levels in his space. The game masters move the objects around the display surface, toward and away from the player; this motion is replicated by the monsters, which move in their individual planes. Figure 6a shows the game masters moving objects and Figure 6b displays the moving monsters in the virtual space.

The mobile player wears a "see-through" Sony Glasstron equipped with two cameras. Fiducials or natural features in the player's space are tracked by the forward facing camera to recover head orientation. This allows graphics (such as the monsters) to be rendered roughly registered with the physical world. The second camera looks down at the player's hands to recognize "martial arts" gestures [21]. To effect attacks on the monsters, the user accompanies the appropriate attack gesture with a Kung Fu yell ("heee-YAH"). Each foe requires a different gesture. Foes that are not destroyed enter the player's personal space and injure him. Enough injuries will cause the player's defeat.

The system has been used by faculty and graduate students in the GVU lab. They have found the experience compelling and balanced. Since it's difficult for the game master to keep pace with the player, two or more game masters may participate (Figure 6a). The Perceptive Workbench's object tracker scales naturally to handle

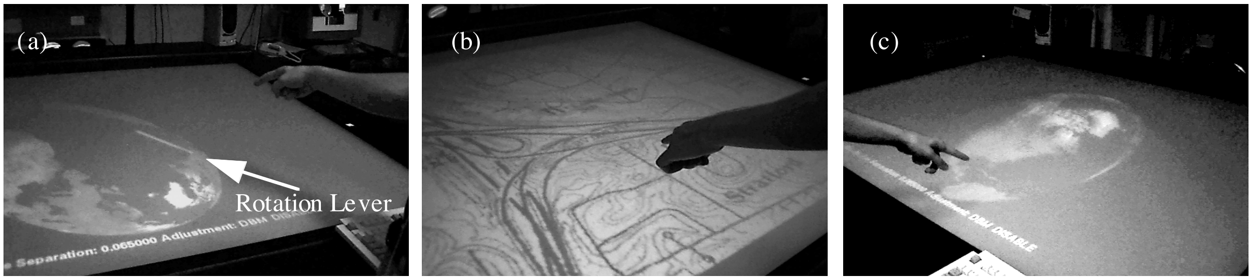

Figure 7 Terrain navigation using deictic gestures: (a) rotation (about an axis perpendicular to and through the end of the rotation lever); (b) zooming in; (c) panning. The selection ray is too dim to see in this view (see Figure 5.)

multiple, simultaneous users. For a more full description of this application, see Starner *et al.* [22].

3D Terrain Navigation

We have developed a global terrain navigation system on the virtual workbench which allows one to fly continuously from outer space to terrain or buildings with features at one foot or better resolution [29]. Since features are displayed stereoscopically [30], the navigation is both compelling and detailed. In our third person navigation interface, the user interacts with the terrain as if it were an extended relief map laid out below one on a curved surface. Main interactions include zooming, panning, and rotating. Since the user is head-tracked he can move his head to look at the 3D objects from different angles. Previously, interaction has been by using button sticks with six degrees of freedom electromagnetic trackers attached. We employ the deictic gestures of the Perceptive Workbench, as described in Section 6, to remove this constraint. Direction of navigation is chosen by pointing and can be changed continuously (Figure 7b). Moving the hand towards the display increases speed towards the earth and moving it away increases speed away from the earth. Panning is accomplished by lateral gestures in the direction to be panned (Figure 7c). Rotation is accomplished by making a rotating gesture with the arm (Figure 7a). At present these three modes are chosen by keys on a keyboard attached to the workbench. In the future we expect to use gestures entirely (*e.g.*, pointing will indicate zooming).

Although there are currently some problems with latency and accuracy (both of which will be diminished in the future), a user can successfully employ gestures for navigation. In addition the set of gestures are quite natural to use. Further, we find that the vision system can distinguish hand articulation and orientation quite well. Thus we will be able to attach interactions to hand movements (even without the larger arm movements).

9. Future Work and Conclusions

Several improvements can be made to the Perceptive Workbench. Higher resolution reconstruction and improved recognition for small objects can be achieved via an active pan/tilt/zoom camera mounted underneath the desk. The color side camera can be used to improve 3D reconstruction and construct texture maps for the digitized object. The reconstruction code can be modified to handle holes in objects. The latency of the gesture/rendering loop can be improved through code refinement and the application of Kalman filters. When given a difficult object, recognition from the reflections from the light source underneath can be successively improved by using shadows cast from the different light sources above or the 3D reconstructed model directly. Hidden Markov models can be employed to recognize symbolic hand gestures [21] for controlling the interface. Finally, as hinted by the multiple game masters in the gaming application, several users may be supported through careful, active allocation of resources.

In conclusion, the Perceptive Workbench uses a vision-based system to enable a rich set of interactions, including hand and arm gestures, object recognition and tracking, and 3D reconstruction of objects placed on its surface. These elements are combined seamlessly into the same interface and can be used in diverse applications. In addition, the sensing system is relatively inexpensive, retailing approximately $1000 for the cameras and lighting equipment plus the cost of a computer with one or two video digitizers, depending on the functions desired. As seen from the multiplayer gaming and terrain navigation applications, the Perceptive Workbench provides an untethered, spontaneous, and natural interface that encourages the inclusion of physical objects in the virtual environment.

Acknowledgments

This work is supported in part by a contract from the Army Research Lab, an NSF grant, and an ONR AASERT grant. We thank Brygg Ullmer, Jun Rekimoto, and Jim Davis for their discussions and assistance. In addition we thank Paul Rosin and Geoff West for their line segmentation code, the Purdue CADLAB for TWIN [25], and P. Cignoni, C. Rocchini, and R. Scopigno for Metro [4].

References

1. Arai, T. and K. Machii and S. Kuzunuki. Retrieving Electronic Documents with Real-World Objects on InteractiveDesk. UIST ‘95, pp. 37-38 (1995).
2. Bolt R. and E. Herranz. Two-handed gesture in multi-modal natural dialog. UIST ‘92, pp. 7-14 (1992).

3. Bimber O. Gesture Controlled Object Interaction: A Virtual Table Case Study. Computer Graphics, Visualization, and Interactive Digital Media, Vol. 1, Plzen, Czech Republic, 1999.
4. P. Cignoni, C. Rocchini, and R. Scopigno. Metro: Measuring Error on Simplified Surfaces, Computer Graphics Forum, vol. 17(2), June 1998, pp 167-174.
5. Coquillart, S. and G. Wesche. The Virtual Palette and the Virtual Remote Control Panel: A Device and an Interaction Paradigm for the Responsive Workbench. *IEEE Virtual Reality '99 Conference (VR'99)*, Houston, March 13-17, 1999.
6. Daum, D. and G. Dudek. On 3-D Surface Reconstruction Using Shape from Shadows. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'98),* 1998.
7. Davis, J.W. and A.F. Bobick. SIDEshow: A Silhouette-based Interactive Dual-screen Environment. *MIT Media Lab Tech Report No. 457.*
8. Kessler, .D., L. Hodges, and N. Walker. Evaluation of the CyberGlove as a Whole-Hand Input Device. *ACM Tran. on Computer-Human Interactions*, 2(4), pp. 263-283 (1995).
9. Kessler, D., R. Kooper, and L. Hodges. The Simple Virtual Environment Libary: User´s Guide Version 2.0. GVU Center, Georgia Institute of Technology, 1997.
10. Kobayashi, M. and H. Koike. EnhancedDesk: integrating paper documents and digital documents. *3rd Asia Pacific Computer Human Interaction*, pp. 57-62 (1998).
11. Krueger, M. *Artificial Reality II.* Addison-Wesley, 1991.
12. Krueger, W., C.-A. Bohn, B. Froehlich, H. Schueth, W. Strauss, G. Wesche. The Responsive Workbench: A Virtual Work Environment. *IEEE Computer*, vol. 28(7). July 1995, pp. 42-48.
13. Lindstrom, P., D. Koller, W. Ribarsky, L. Hodges, N. Faust, and G. Turner. Real-Time, Continuous Level of Detail Rendering of Height Fields. Report GIT-GVU-96-02, *SIGGRAPH*, pp. 109-118 (1996)
14. Lindstrom, P., D. Koller, W. Ribarsky, L. Hodges, and N. Faust. An Integrated Global GIS and Visual Simulation System. Georgia TR GVU-97-07 (1997).
15. May, R. HI-SPACE: A Next Generation Workspace Environment. Master's thesis, Washington State Univ. EECS, June 1999.
16. Rekimoto, J., N. Matsushita. Perceptual Surfaces: Towards a Human and Object Sensitive Interactive Display. *Perceptual User Interfaces (PUI)*, 1997.
17. Rosin, P.L. and G.A.W. West. Non-parametric segmentation of curves into various representations. *IEEE PAMI'95*, 17(12) pp. 1140-1153 (1995).
18. Schmalstieg, D., L. M. Encarnacao, Z. Szalavar. Using Transparent Props For Interaction With The Virtual Table. *Symposium on Interactive 3D Graphics (I3DG'99)*, Atlanta, 1999.
19. Seay, A.F., D. Krum, W. Ribarsky, and L. Hodges. Multimodal Interaction Techniques for the Virtual Workbench. <u>CHI 99</u>.
20. Srivastava, S.K. and N. Ahuja. An Algorithm for Generating Octrees from Object Silhouettes in Perspective Views. *IEEE Computer Vision, Graphics and Image Processing*, 49(1), pp. 68-84 (1990).
21. Starner, T., J. Weaver, A. Pentland. Real-Time American Sign Language Recognition Using Desk and Wearable Computer Based Video. *IEEE PAMI* , 20(12), pp. 1371-1375 (1998).
22. Starner, T., B. Leibe, B. Singletary, and J. Pair. MIND-WARPING: Towards Creating a Compelling Collaborative Augmented Reality Game. *Intelligent User Interfaces (IUI) (2000).*
23. Sturman, D. Whole-hand input. Ph.D. Thesis, MIT Media Lab (1992).
24. Sullivan, S. and J. Ponce. Automatic Model Construction, Pose Estimation, and Object Recognition from Photographs Using Triangular Splines. *IEEE PAMI*, 20(10), pp. 1091-1097 (1998).
25. TWIN Solid Modeling Package Reference Manual. Computer Aided Design and Graphics Laboratory (CADLAB), School of Mechanical Engineering, Purdue University, 1995. cadlab.www.ecn.purdue.edu/cadlab/twin/TWIN_TOC.html
26. Ullmer, B. and H. Ishii. The MetaDESK: Models and Prototypes for Tangible User Interfaces. *Proceedings of UIST'97*, October 14-17, 1997.
27. Underkoffler, J. and H. Ishii. Illuminating Light: An Optical Design Tool with a Luminous-Tangible Interface. *Proceedings of CHI '98*, April 18-23, 1998.
28. van de Pol, Rogier, W. Ribarsky, L. Hodges, and F. Post. Interaction in Semi-Immersive Large Display Environments. <u>Virtual Environments '99</u>, pp. 157-168 (Springer, Wien, 1999).
29. Wartell, Z., W. Ribarsky, and L. Hodges. Third Person Navigation of Whole-Planet Terrain in a Head-tracked Stereoscopic Environment. *IEEE Virtual Reality 99*, pp. 141-149 (1999).
30. Wartell, Z., L. Hodges, and W. Ribarsky. Distortion in Head-Tracked Stereoscopic Displays Due to False Eye Separation. *SIGGRAPH 99*,.pp. 351-358 (1999).
31. Wellner P. Interacting with paper on the digital desk. *Comm. of the ACM*, 36(7), pp. 86-89 (1993).
32. Wren C., A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-Time Tracking of the Human Body. *IEEE PAMI*, 19(7), pp. 780-785 (1997).