On the usability of gesture interfaces in virtual reality environments

Marcio C. Cabral
Department of Computer Science - University of São Paulo
Rua do Matão, 1010 - São Paulo - Brazil
mcabral@ime.usp.br

Carlos H. Morimoto
Department of Computer Science - University of São Paulo
Rua do Matão, 1010 - São Paulo - Brazil
hitoshi@ime.usp.br

Marcelo K. Zuffo
Department of Electrical Engineer - University of São Paulo
Rua Prof. Luciano Gualberto, 158 - São Paulo - Brazil
mkzuffo@lsi.usp.br

ABSTRACT
This paper discusses several usability issues related to the use of gestures as an input mode in multimodal interfaces. The use of gestures has been suggested before as a natural solution for applications that require hands-free and no-touch interaction with computers, such as in virtual reality (VR) environments. We introduce a simple but robust 2D computer vision based gesture recognition system that was successfully used for interaction in VR environments such as CAVEs and Powerwalls. This interface was tested under 3 different scenarios, as a regular pointing device in a GUI interface, as a navigation tool, and as a visualization tool. Our experiments show that the time to completion of simple pointing tasks is considerably slower when compared to a mouse and that its use during even short periods of time causes fatigue. Despite these drawbacks, the use of gestures as an alternative mode in multimodal interfaces offers several advantages, such as quick access to computing resources that might be embedded in the environment, using a natural and intuitive way, and that scales nicely to group and collaborative applications, where gestures can be used sporadically.

AUTHOR KEYWORDS
Usability of Gesture Interfaces, Virtual Reality, Computer Vision

INTRODUCTION
As new technologies such as Ubiquitous Computing, Smart Places, Augmented and Virtual Reality Environments arise and become part of our everyday lives, the understanding of how gestures can be more effectively used to communicate with computers becomes more important. Such technologies pose several challenges on the GUI (Graphical User Interface) paradigm designed for desktop interaction, particularly on current input devices such as keyboards and mice.

Potential applications for these new technologies require input devices that can be easily carried along with the user and instantly available when needed (or made available by the environment). Although wearable devices might be an option, a computer vision based gesture recognition system seems to be a more natural alternative since it can be considered as part of the environment and operates remotely, i.e., it does not require the user to wear or have any physical contact with a device.

Gestures have been proposed and used in several multimodal interfaces to create more natural and intuitive ways of communicating with computers [2, 6, 7, 17, 21], including interfaces for people with physical disabilities [10]. A gesture based interface has the potential of eliminating the need of pointing devices, thus saving the time and effort in using such devices for the interaction process. But performing gestures for long periods of time can be physically stressing and therefore, as pointed by Nielsen *et al.* [19], a gesture interface should be used as an alternative to existing interface techniques or complementing them. Besides, a gesture may not be the most efficient way to perform all application related tasks. In this paper we conduct 3 experiments to study the performance of gesture input regarding 3 different situations: a simple GUI interface, a 3D object visualization tool, and a 3D scene exploration tool.

Our experiments were conducted in Virtual Reality (VR) environments, such as the CAVE (Cave Automatic Virtual Environment) [4, 23] shown in Figure 1a. Such environments offer unlimited possibilities for scientific visualization and exploration, by providing computer generated multi-sensory (mostly limited to audio-visual) information to the user. The CAVE is a room constructed of large screens on which the graphics are projected onto all walls, including the floor and/or ceiling, that allows multiple users to share the same experience. Powerwall is a simpler solution, usually built with just one or two large screens, thus the sensation of immersion is not as complete. Other possible alternatives, but not as comfortable, are head mounted displays and binoc-

ular omni-oriented monitor (BOOM) displays, that are also able to provide the wide field of view presentation required to create the experience of being immersed in the virtual environment.

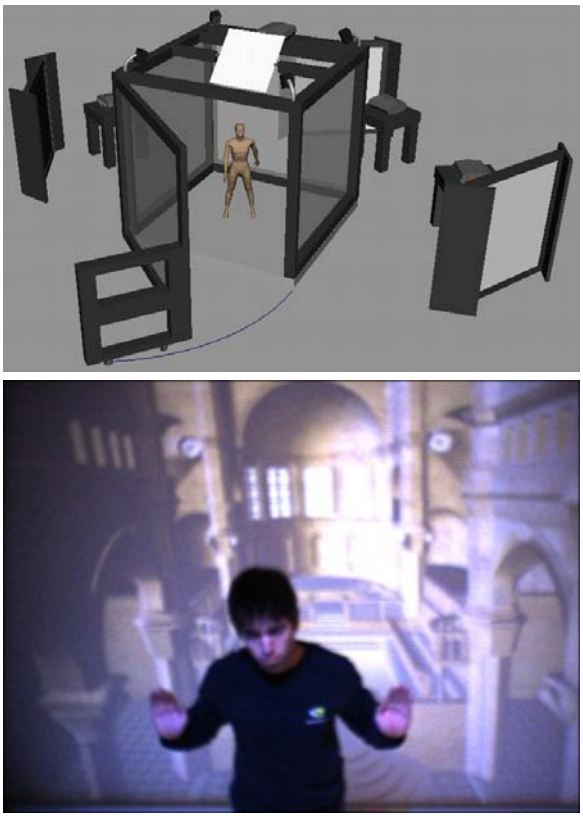

Figure 1. Top image: VR environment (CAVE) with 5 projection screens. Bottom image: view of the camera when the user is positioning his hands to initialize the system (observe the poor illumination conditions).

GESTURE BASED INTERFACES

Cassell [2] classifies gestures as conscious or spontaneous, and interactional or propositional. Conscious gestures have meaning without speech, while spontaneous gestures only have meaning in context of speech. Propositional gestures carry the conversational content, while interactional gestures consists of cues that affect the conversational process, such as humor or sarcasm. Another classification is given by Hummels and Stapers [8], that classify gestures as static and dynamic gestures, i.e., a gesture can be defined by a static pose of the hand or body (static gesture), or by their physical motion in two or three dimensions (dynamic gesture).

The purpose of a gesture recognition system is to identify specific human gestures. Once a gesture is identified, it is translated into symbols, which can be used to convey information or for device control. The collection of symbols used in the interface is called the "gesture vocabulary". Nielsen *et al.* [19] uses a *technology based vocabulary*, i.e., only gestures that are easily recognized by current technology are used to compose the vocabulary. A similar idea was used by Freeman and Weissman [6] to remotely control a television set, and by Hong *et al.* [7] to control a *Simon Says* kind of application, where simple gestures such as waving hands are recognized using finite state machines. Streitz *et al.* [21] go far from the current technical limitations to propose the use of gestures as one of many possible input modes to be used by *roomware*. Roomware is the software in a futuristic scenario where a room may be populated with computer enhanced walls, desks and chairs, that can be used for collaborative work, for example.

Before gestures can be incorporated into real interfaces, quite a few technological issues must be solved, as well as usability issues. So far, most publications deal with the technical issues and very few of them deal with usability, particularly concerning performance evaluation.

Usability [18] is mainly concerned with 5 main topics: learnability, efficiency, memorability, errors, and satisfaction. Therefore, an interface with good usability should be easy to learn (learnability), allow expert users to efficiently perform their tasks (efficiency) and casual users to reuse the interface without the need to learn the tasks again (memorability). The number of errors should be small, and when they occur, they should be easy to correct. Subjective factors, such as trust and comfort, are also important for usability, and they are included in the satisfaction item. For a gesture interface, the ergonomics of the gesture is also important, since the interface should not be physically stressing. The next section describes a 2D gesture interface that was developed considering these usability issues.

DESIGN OF A 2D GESTURE INTERFACE

In order for gestures to be easy to learn and remember, they should be natural. Finger and arm pointing is probably one of the most basic gestures we all use. Moeslund *et al.* [17] describe a computer vision system that is used to recognize pointing gestures, where the user just extends his/her arm towards a large screen.

Because keeping the arm extended repetitively, even for short periods, can be stressing, we propose the mapping of the hand motion to control the cursor, similar to a mouse in a common graphic user interface. Thus the user may keep her/his arms closer to the body, and hold this position for longer periods. Nielsen *et al.* [19] list the following principles in ergonomics to build a good gesture interface:

1. Avoid outer positions
2. Relax muscles
3. Relaxed neutral position is in the middle between outer positions
4. Avoid repetition
5. Avoid staying in static position
6. Avoid internal and external force on joints that may stop body fluids

Figure 1 shows the rest position of our gesture interface. The head (H), the right hand (R) and the left hand (L) are

tracked by the computer vision system. To control applications from inside the VR environments, the user has to learn just a few basic commands, executed with both left and right hands. The left hand is used for several operations, that are activated depending on the position of L relative to H. The 2D position of the hands and face in the images are identified by the coordinate of their center of mass, defined by (γ_x, γ_y), where γ may correspond to R, L or H.

The computer vision system can identify the following five positions for L: Rest (or initial position), North, South, East, and West. All these positions are computed relative to the position of H in the camera image.

To allow the user to rest his/her arm, the Rest and South positions do not activate any interface action. The North position is used to switch the mode that controls the cursor/screen behavior, for example, switching the navigation mode such as translations or rotation in a 3D exploration application. The West position is used to switch operation modes, like switching from x-y panning to y-z panning, controlled by the motion of R. The East position is used for clicking and double clicking.

The right hand is basically used for pointing, i.e., to control the position of the cursor. In order for the mapping of the hand motion to cursor motion to be natural and comfortable, the system allows the user to define an arbitrary quadrilateral confined in the right side of her/his body by a calibration procedure. After calibration, the system maps the position of R inside this quadrilateral to screen coordinates, using the following affine transformation:

$$\begin{aligned} x_s &= A_{11}x_a + A_{21}y_a + A_{13} \\ y_s &= A_{12}x_a + A_{22}y_a + A_{23} \end{aligned} \quad (1)$$

where $\mathbf{A}$ is a 3×3 image-to-screen transformation matrix, (x_s,y_s) are the screen coordinates, and (x_a, y_a) are the image coordinates of R. The coefficients of the matrix A are computed from the 4 points defined by the calibration procedure. During the procedure, the user must hold the right hand position and lift his left arm above the head to confirm each position. This procedure is very simple and takes less than 10 seconds, so that recalibration can be done as often as required. To allow for the user to move, the coordinates of L and R are always considered relative to H, similar to [12].

The position of R could be used directly to control the cursor position, but due to the wide field of view of the camera, the region of maximum excursion of the hands is relatively small compared to a high resolution display. Therefore, small estimation errors in the position of the hands create noticeable cursor jitter. To smooth the cursor behavior, we use a Finite Impulse Response (FIR) digital filter with a low-pass frequency response. The filter used in our prototype is described by the following time-domain difference equation:

$$\begin{aligned} &a(0) * y(t) = \\ &[b(0), b(1), \ldots, b(n)]^T [x(t), x(t-1), \ldots, x(t-n)] \end{aligned} \quad (2)$$

where $n = 5$, $y(t)$ and $x(t)$ are respectively the output and input of the filter at instant t, the coefficient $a(0)$ is 1, and the coefficients b were obtained empirically. The actual values used in our experiments are $b[0..5]$ = {0.0174, 0.025, 0.0295 , 0.0295 , 0.025 , 0.0174}. This filter significantly reduces the cursor jitter without compromising the interface real-time response.

Interaction modes

A COMPUTER VISION SYSTEM FOR 2D GESTURE RECOGNITION

The interaction modes define how the gestures and poses are mapped to an application's actions. We define 3 modes that are common to VR environments. In the first mode of interaction, defined earlier with R used for pointing and L used for click and control, the gesture interface behaves very similar to a point-and-click GUI interface, which is familiar to most computer users, and therefore, it would be very easy for them to learn. We use this mode to compare the performance of gesture based interaction with mouse-based interaction.

A very common application for VR environments is the exploration of 3D scenes and objects. For such applications we defined two other modes of interaction, one for visualization and manipulation of 3D objects and a another for the navigation in 3D scenes.

For the visualization and manipulation of 3D objects, the interaction mode is defined by the following mapping:

- R controls the navigation when L is in the Rest/South position.
- the main navigation mode changes when L reaches the North position, sequentially, in a circular list containing the modes:
  1. `scale`,
  2. `rotation`, and
  3. `translation`.
- the orientation of the current navigation mode changes when L reaches the West position, sequentially, in a circular list corresponding to the following values:
  1. `plane x-y` or `z-axis`,
  2. `plane y-z` or `x-axis`,
  3. `plane z-x` or `y-axis`.
- When L crosses to the right side, a click is generated. A double click toggles the animation of the 3D visualization tool. When the animation is frozen, the system behaves like a GUI interface.

The third mode of interaction is the navigation in 3D scenes. In such applications, the user is located inside the object,

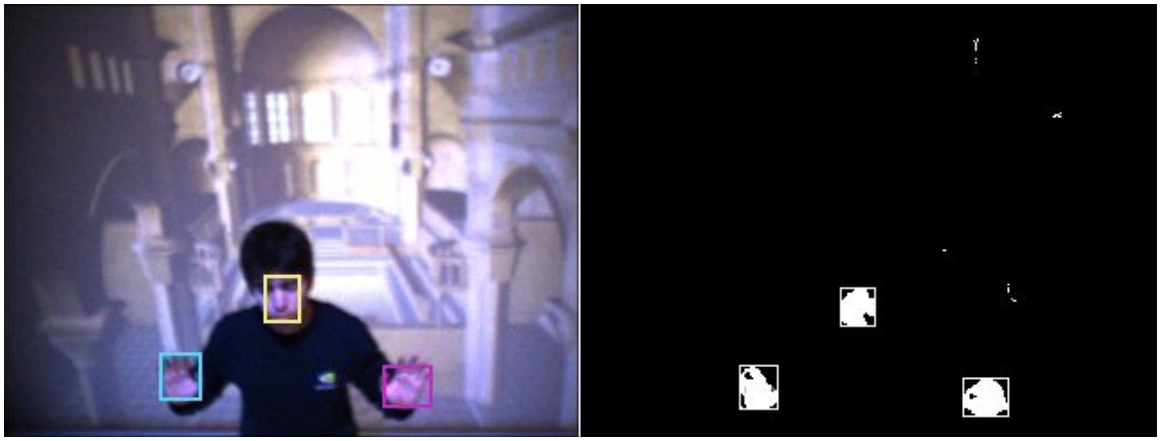

Figure 2. Captured image with blobs being tracked and its segmented skin color regions.

while for 3D object visualization, the user is located outside the object. For navigation, the following actions are defined:

- R controls the viewing direction;
- L controls the camera translation. When L is up (North), the camera moves towards the viewing direction. When L is down (South), the camera moves backwards from the viewing direction;

For all 3 interaction modes, the gesture interface can be switched on/off by lifting both hands at the same time above the head and back to the Rest position.

The use of computer vision to recognize gestures allows the user to interact with virtual objects with the great advantage of not requiring the user to hold or wear any special equipment or attach any devices to his/her body, such as data gloves and 3D mice, which are common devices used by many VR applications.

Common techniques used to detect and track the hand and body for gesture recognition are background subtraction and skin color segmentation. Pfinder [22] is an example of a complete computer vision system used for gesture recognition in the Alive project. The system uses several domain-specific assumptions to make the vision task tractable, such as a static background. Starner *et al.* [20] describe a computer vision system that is able to recognize 40 symbols from the American Sign Language. The hands are tracked using color information, and no assumption is made about the background but constant illumination conditions for good color segmentation. Because part of the face is always visible in their system, they suggest the use of that part for color calibration.

Gesture recognition in VR environments, such as CAVEs, is a very challenging problem due to the severe limitations of camera positioning and lighting (see Figure 1b). Several multi-view approaches for body pose estimation have been published in the last few years. Most of them try to fit an articulated body model to the 3D data [3, 16], but fast tracking of such articulated structures is far from trivial. Kehl *et al.* [11] propose to limit the number of gestures by detecting only pointing gestures. Leubner [14] also utilizes multi-view geometry in a back projected immersive wall but the system is limited to 8 frames per second.

Our system, described in detail in [1] uses color based tracking to achieve high frame-rate in combination with a dynamic color calibration scheme to achieve robustness to changes in illumination conditions. High framerate is desirable in order to minimize the response delay to the user's actions. Lemmerman [13] uses special cameras with framerates of up to 250Hz, but the user must wear markers to simplify the image processing algorithms.

Color segmentation

We consider that the user will be interacting in a CAVE or Powerwall environment, and assume that these environments define a main screen towards which the user directs the gestures for interaction. This way we can limit the computer vision system to a single camera.

To start the system, the user must stand in front of the main screen, showing both his hands open about the user's chest region. Figure 2 on the left shows this initial position. Because the field-of-view of the camera is known and fixed, the system expects the head and hands to appear in predetermined regions in the camera image.

The user's head is first detected in its corresponding region using skin color information. We have used sample images from 16 people of different ethnicities to train a simplified skin color model using the RGB color space. A Bayesian decision rule, obtained from the training data, is used to segment skin color pixels.

Pixels classified as skin are filtered using morphological operations and then clustered into blobs using a connected component labelling algorithm. A Principal Component Analysis (PCA) face detection technique is used to select the blob with the highest probability of being a frontal face. Once the face is detected, selected regions of the face are

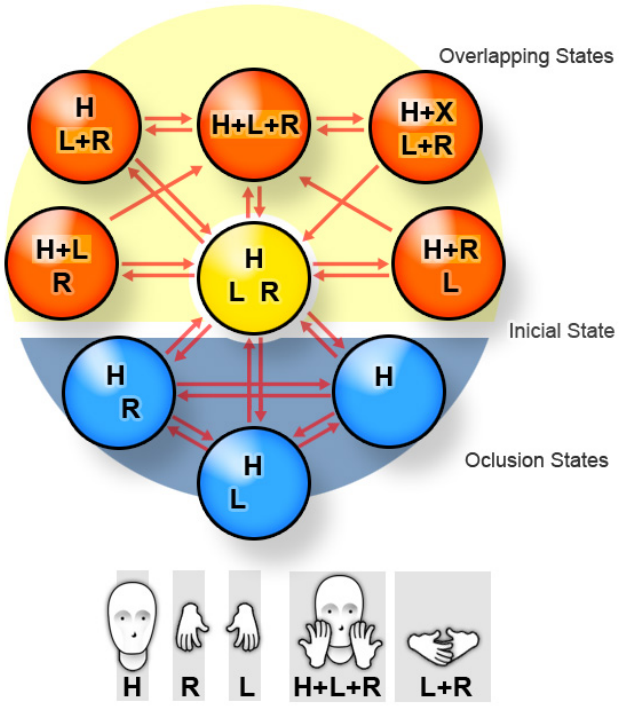

Figure 3. Spatial-temporal blob consistency graph. The tokens inside each state of the graph show the blobs being tracked associated with the belief of what they correspond.

used to fit a per user's skin color model. The per user's color model is updated periodically to compensate for illumination changes. The right image in Figure 2 shows the blobs segmented from the input image shown on the left using the adjusted color model.

Head and hands tracking

The bounding boxes around the segmented blobs are used for tracking. Although the image quality is poor due to the illumination conditions, the system is able to segment and track the user's head and hands. Kalman filters (KFs) are used to estimate the size and position of each blob in the next frame. The actual blobs detected using the dynamic skin color models are used to update the filter parameters.

Tracking using KFs may fail when there is occlusion or when the blobs overlap each other, resulting in a single blob. Occlusion happens when a hand gets behind the body or arm. We assume that the head is never occluded, but a hand, or both hands, may overlap the head region. Due to the nature of the interaction, the hands overlap each other often.

In order to reliably track the blobs even when the blobs are occluded or overlap, a spatial-temporal blob consistency graph (see Figure 3) is used.

Once the system is initialized, it starts tracking H , L and R , as three independent blobs. Once L (or R) becomes occluded by the user's arm or body, the Kalman filter is not updated. The system waits up to 3 seconds for the blob to reappear near the location where it was lost. If a blob is not detected after this period, the system searches for a blob within the region where L (or R) is most expected to show, relative to H . Only when H is lost, the system is not able to recover, and expects the user to return to the initial position.

The upper nodes of the graph represent overlapping states of the hands and head that the system is able to identify. The lower nodes represent occlusion nodes.

When the system is at the initial state, the prediction of the position and size of the blobs define, for each blob, a search region within the next frame. When such regions overlap, a single box is used for predicting the position of both objects. When a single blob is found inside this bounding box the machine assumes that both objects have joined, determined by the labels L+R or H+X, where X can be associated with either L or R .

We have implemented a prototype system that runs on a notebook Pentium IV 2.8GHz with a *firewire* (IEEE 1394) camera attached to it. The system is able to process 30 frames per second of resolution $320 \times 240 \times 24$, which is the maximum framerate achieved by the camera (Dragon Fly Cameras [9]). In the CAVE application, in order to improve image quality for segmentation, the system is set to run at 15 frames per second, increasing the exposition time of the sensor to provide brighter images.

EXPERIMENT PROCEDURES AND RESULTS

The CAVE and Powerwall environments used in our experiments are shown in figures 1 and 4, respectively. For the CAVE environment, the camera is placed above the main screen used for interaction, and the side walls are used to convey the immersive experience. Since the back wall is used, it reflects the light from the other projections, and therefore the background cannot be assumed to be static. Also, since the floor screen is the only one where the CAVE has front projection, it is used to compensate the low light ambient illumination, providing better contrast and robustness to the computer vision system.

In the Powerwall environment, the camera is placed below the screen being projected and the user stands in front of it. Again, no assumption of the background was made.

We have tested the performance of the gesture interface in both CAVE and Powerwall environment. Three experiments were devised to evaluate the usability of the gesture interface. The first experiment compares the performance of gesture interface with mice interfaces, using a simple point and click task. The other experiments evaluate the user satisfaction with the navigation and visualization applications.

Point and click experiment

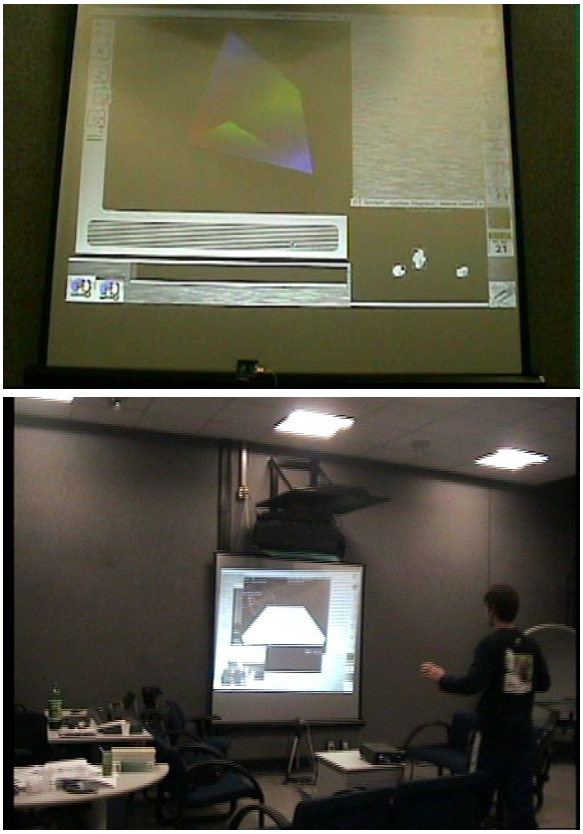

Figure 4. Top: powerwall configuration. The camera is placed below the projected screen. Bottom: the user stands in front of the screen.

This experimental task is essentially a Fitts' pointing task and its objective is to compare the performance between mouse and gesture input.

Fitts law [5] is commonly used as a way to measure and compare different types of pointing devices. It was proposed in 1954 by Fitts and it provides a numerical measure of the time a user requires to move the cursor (pointing device) to a specific location on the screen, given the distance to the target and also its size. Mackenzie *et al.* [15] propose a variation of the Fitts law that allows for a better adaptation to empirical data. The modified equation is given by

$$T = a + b \log_2(\tfrac{A}{W} + 1) \qquad (3)$$

where T is the time to execute the movement, A is the amplitude of the movement and W is the size of the target. The coefficients a and b are empirically determined through the use of linear regression. The inverse of b is called Index of Performance (IP) and the term $\log_2(\frac{A}{W} + 1)$ is called Index of Difficulty (ID). Observe that the difficulty of a task is proportional to the amplitude and inversely proportional to the size of the target, that is, the farther the target the more difficult it is to point it, and the smaller the target the more difficult it is to point it.

Modality	**a**	**b**	**IP**
Mouse	$240,43$	$215,31$	$0,0046$
Gesture	$121,83$	$649,61$	$0,0015$

Table 1. Coefficients a e b from equation 3 are obtained empirically.

The task was to point and click at targets appearing in random order. Targets changed sizes and positions in order to collect data to fit Fitts' Law. If the subject clicked off-target, a miss was logged but the trial continued until the target was clicked. An extra trial was added to make up for the missed trial. Only trial with no misses were collected for time performance analysis. Subjects were asked to complete the task as quickly as possible and as accurately as possible.

A within-subject design was used. Each subject performed the task with both techniques (gesture and mouse). 8 graduate students were used as subjects. The subjects average age was 25 years old. All students have experience with virtual reality and computer graphics, and an average of 7.3 years using computers with mouse/keyboard. The subjects were instructed to utilize the system in a natural way, and it was emphasized that the system was being tested and not the subjects themselves.

For each technique, a trial practice session was given to allow the users to explore the interface. The practice session was followed by two data collection sessions. In each one of these sections, the users had to click on ten different targets.

Due to the pilot nature and small scale of the experiment, we expect the statistical power of the results to be weak. The average time to completion of each trial was 7.3 seconds using the mouse and 26.1 seconds using gestures in the Powerwall environment. Only this VR environment was used for this experiment due to the constraint on the screen size, so the user does not have to worry about targets appearing in other screens.

This considerable difference in performance may be partially explained by the inexperience of the users with the gesture interface, but it was expected that the ID would be considerably larger due to the discrepancy in the amplitudes of gestures and mouse motion. The gestures are executed by the whole arm, reaching amplitudes of several centimeters, while mouse movements are executed by the hand, wrist and elbow, with amplitudes of a few centimeters.

As predicted by Fitts' law, movement time increased for longer distances. Due to the nature of arm movement, the amplitude in the gesture modality does not affect the time of completion as much as the mouse modality.

The coefficient b indicates the slope of the fitted line, or IP$^-$1. Observe in Table 1 that the value of IP for the gesture interface is approximately one third of the IP for the mouse interface, but they are both small, so the the inclination of the lines in Figure 5 look approximately the same. From this

data it is obvious that the performance of the user using the mouse is much better than the performance of the user using gestures, and this ratio favors the mouse even better as the Index of Difficulty increases. This result is also expected due to the nature of arm $\times$ hand movements involved.

3D visualization experiment

The objective of the 3D visualization experiment is to qualitatively evaluate the usability of the gesture interface in VR environments. The subjects for this experiment were told to explore the 3D object manipulation tool with the gesture interface. This application allows the user to select operation modes and manipulate 3D objects around the screen, changing their sizes, positions and orientations. Figure 6b shows this application being tested in the Powerwall environment.

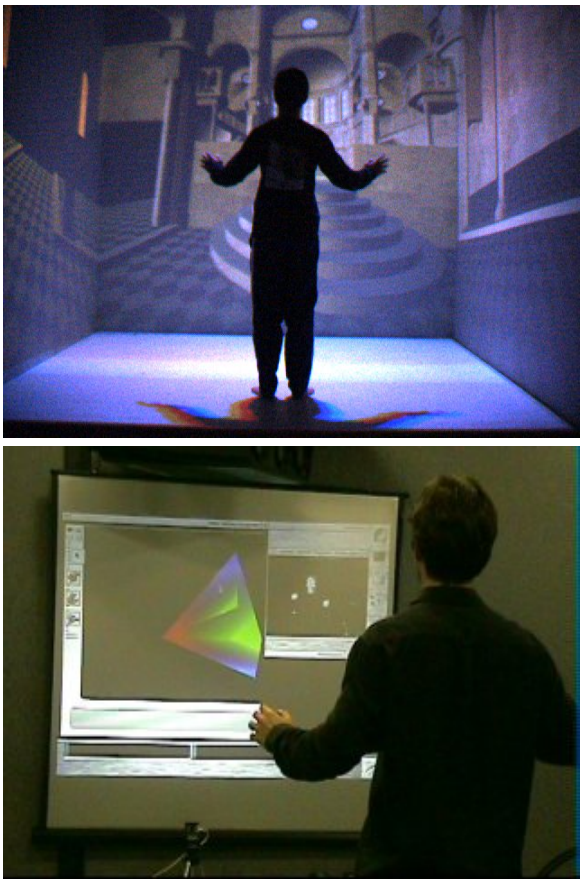

Figure 6. Top: object visualization experiment. Bottom: 3D navigation experiment.

In this experiment, the users also spent some time familiarizing themselves with the application and the interface. The subjects were asked to answer questionnaires before and after the experiment, regarding their experience with VR environments and applications, familiarity with gesture interfaces, and mouse/keyboard paradigm. During the experiment, the subjects were also encouraged to express their opinion about the system. The questionnaires contained questions about users' experience in VR, familiarity with gesture interfaces, mouse/keyboard paradigm and VR input devices.

The results obtained showed that the computer vision system is quite robust to 3D object manipulation. About 50% of the users reported to be excellent the response time of the system to rapid and sudden movements and to key positions that activate the menu. However 40% of the users complained about fatigue of the arms.

3D navigation experiment

For the 3D navigation experiment, the gesture interface substituted a joystick that was previously used as the input device. Actions that are allowed by the interface include changing viewing position and navigating through the environment. The top image in Figure 6 shows the navigation/exploration application used in this experiment, a walkthrough across a virtual environment, using the CAVE environment.

Subjects used in this experiment reported that the freedom of movement and the ability to freely move in the environment without wearing or caring other devices was very important to improve the sensation of immersiveness, and also show the robustness and real time performance of the computer vision based gesture recognition system.

CONCLUSION

We have presented a brief review of several systems that propose the use of gestures to interact with computer systems. Most of them only discuss technological issues related to their implementation, and only a few discuss the usability of this interaction paradigm.

Based on the 5 main usability principles, we have developed a simple 2D vision based gesture recognition system that is easy to learn, remember, and is quite efficient and robust to changes in illumination conditions. The vision system was used to implement 3 different modes of interaction that were successfully applied to Virtual Reality (VR) environments. The first mode mimics a common GUI interface, the second allows for the manipulation of 3D objects, and the third mode allows for the navigation in 3D scenes.

The usability of each of these interaction modes were evaluated by different experiments. The first experiment was a simple Fitts' pointing test, where the performance of the gesture and a mouse based GUI interfaces were compared. The other modes of interaction were subjectively evaluated. As expected, the performance of the gesture interface for pointing task is considerably worse than a mouse, due to the lack of experience of the subjects and mainly to the large amplitude of the gestures compared to the mouse movements. This large amplitude also explains the fatigue reported by some of the subjects. Despite this drawbacks, the subjective experiments show that the interface is easy to use and learn, and is appropriate for short sporadic use, such as controlling a TV when the phone rings and helping collaborative work during a group session. We are extending this work to process the gestures of multiple users. Depending on the visual

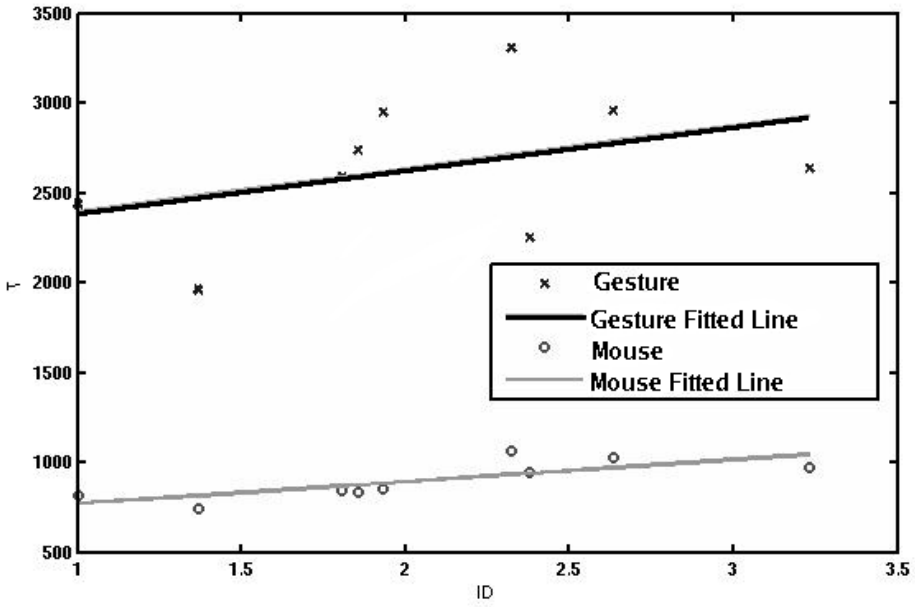

Figure 5. Line $T = a + bID$ for both modalities: mouse and gesture

angle of the camera, several users can be tracked at the same time, and therefore a single system could be used by several users. Another advantage is that the system is always available without the need of extra devices such as data gloves, laser pointers or remote controls.

Acknowledgment

The authors would like to thank the laboratory of integrated systems (LSI) for providing access to its virtual reality facility and all researchers who, somehow, helped this work to be accomplished. The authors would also like to thank FINEP for providing research funds for equipment expenditures.

REFERENCES

1. M.C. Cabral. Interface baseada em gestos para ambientes de realidade virtual usando visão computacional. Master's thesis, Departamento de Ciência da Computação, Universidade de São Paulo, 2005.

2. J. Cassell. A framework for gesture generation and interpretation. In R. Cipolla and A. Pentland, editors, *Computer Vision in Human-Machine Interaction*, New York: Cambridge University Press, 1998.

3. G. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture. *CVPR*, 2003.

4. C. Cruz-Neira, D. Sandin, and T. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the cave. *ACM SIGGRAPH*, July 1993.

5. P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology: Human Perception and Performance*, 47, 1954.

6. W.T. Freeman and C.D. Weissman. Television control by hand gestures. In *Proc. of the Int. Workshop on Automatic Face and Gesture Recognition*, Zurich, Switzerland, June 1995.

7. P. Hong, M. Turk, and T. Huang. Gesture modeling and recognition using finite state machines. *Proc. 4th IEEE Int. Conf. on Face and Gesture Recognition*, 2000.

8. C. Hummels and P.J. Stapers. Meaningful gestures for human computer interaction. In *Proc. of the 3rd International Conference on Automatic Face and Gesture Recognition*, Nara, Japan, April 1998. Computer Society Press, Los Alamitos, CA.

9. Point Grey Research Inc. Dragon fly firewire camera specifications - http://www.ptgrey.com/products/dragonfly/specifications.html.

10. S. Keates and P. Robinson. The use of gestures in multimodal input. In *Proc. of the ACM SIGCAPH ASSETS 98*, 1998.

11. R. Kehl and L. V. Gool. Real-time pointing gesture recognition for an immersive environment. *IEEE Intl. Conference on Automatic Face and Gesture Recognition*, 2004.

12. F. C. M. Kjeldsen. Visual interpretation of hand gestures as a practical interface modality. *Phd Thesis - Columbia University*, 1997.

13. D. Lemmerman, A. Forsberg, L. Monroe, K. Chartrand, B. Greene, D. Modl, and J. Olivares. Towards human-centered user interfaces in an immersive context. *Proc. IEEE VR 2004 - Beyond Wand and Glove Based Interaction*, 2004.

14. C. Leubner, C. Brockmann, and H. Muller. Computer-vision-based human-computer interaction with a back projection wall using arm gestures. *Euromicro Conference*, 2001.

15. S. MacKenzie, A. Sellen, and W. Buxton. A comparison of input devices in element pointing and dragging tasks. *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, 1991.

16. I. Miki, M. Trivedi, E. Hunter, and P. Cosman. Human body model acquisition and tracking using voxel data. *Int. Journal of CV*, 53(3), 2003.

17. T.B. Moeslund, M. Störring, and E. Granum. A natural interface to a virtual environment through computer vision estimated pointing gestures. In I. Wachsmuth and T. Sowa, editors, *Gesture and Sign Language in Human Computer Interaction: International Gesture Workshop, GW 2001*, volume 2298, London, UK, April 2001. Springer LNAI2298.

18. J. Nielsen. *Usability Engineering*. Morgan Kauffmann, San Diego, California, 1993.

19. M. Nielsen, T. Moeslund, M. Störring, and E. Granum. A procedure for developing intuitive and ergonomic gesture interfaces for human computer interaction. In *Proc. of the 5th Internation Gesture Workshop, GW 2003*, Genova, Italy, April 2003.

20. T. Starner and A. Pentland. Visual recognition of american sigh language using hidden markov models. *IEEE Int. Symp. on Computer Vision*, 1995.

21. N. Streitz, P. Tandler, C. Mller-Tomfelde, and S. Konomi. Roomware: Towards the next generation of human-computer interaction based on an integrated design of real and virtual worlds. In J. Carroll, editor, *Human-Computer Interaction in the New Millenium*. Addison-Wesley, 2001.

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

23. J. Zuffo and M. K. Zuffo. Caverna digital - sistema de multiprojeção estereoscopico baseado em aglomerados de pcs para aplicações imersivas em realidade virtual. In *4th SBC Simposio de Realidade Virtual*. Sociedade Brasileira de Computação, Outubro 2001.