AN INTERACTION MODEL DESIGNED FOR HAND GESTURE INPUT

Thomas Baudel
Michel Beaudouin-Lafon

L.R.I. - CNRS URA 410
Bâtiment 490,
Université de Paris-Sud
91405 Orsay Cedex - FRANCE
33+ 1 69 41 69 10
thomas@lri.fr, mbl@lri.fr

Annelies Braffort
Daniel Teil

L.I.M.S.I.
Bâtiment 508,
Université de Paris-Sud
91405 Orsay Cedex - FRANCE
33+ 1 69 85 81 10
braffort@limsi.fr, teil@limsi.fr

RÉSUMÉ
Nous décrivons un modèle d'interaction permettant l'utilisation des gestes de la main dans la communication homme-machine de manière non-intrusive: l'utilisateur peut à la fois interagir avec la machine et agir dans le monde ambiant. Des règles simples permettent de détecter l'intention de l'utilisateur, et déterminer si celui-ci s'adresse à la machine ou non. Une application de ce modèle est évaluée: il s'agit d'un système de présentation assistée par ordinateur permettant à l'utilisateur de naviguer aisément dans un ensemble de transparents au moyen de commandes gestuelles. Les résultats obtenus nous permettent de présenter des applications de ce modèle et d'envisager de nouvelles directions de recherches pour son amélioration.

ABSTRACT
This paper describes an interaction model that allows a user to interact with a computer system through gestures while using gestures at the same time for interaction in the real world. This model uses simple rules for detecting the user's intention to address a gestural command to the system. Some rules are embedded in the model, while others are part of a set of guidelines for designing gestural command sets. Using this model, we have designed an application that allows a user to give a lecture by navigating through a set of slides with gestural commands. From the results of a usability testing of this application, we present other potential applications and directions for future work.
This research aims at providing usable applications of hand gesture input in the real world. The results presented in this paper are of interest to researchers working on new interaction techniques based on gestures as well as designers of actual systems.

MOTS CLÉS: Modèles d'interaction, Interfaces gestuelles, Virtualité incarnée, Télécommandes.

KEYWORDS: Interaction models, Hand gesture input, Embodied virtuality, Remote controlled user interfaces.

INTRODUCTION

> *The machine was rather difficult to operate. For years, radios had been operated by means of pressing buttons and turning dials; then, as the technology became more sophisticated, the controls were made touch sensitive […] now all you had to do was wave your hand in the general direction of the components and hope. It saved a lot of muscular expenditure of course, but meant you had to stay infuriatingly still if you wanted to keep listening to the same programme.* D. Adams, 1979 [1]

Using hand gestures as an input media is not a new idea. In 1979, the "put that there" experiment [3] already used primitive gestural input. More recently, the availability of new devices to track hand gestures (such as the VPL Dataglove® [19]) and the advent of virtual reality systems have popularized the concept of gestural input. The expected advantages of hand gesture input can be summarized as follows:

• *Natural interaction*: Gestural input makes use of the natural human communicative skills to interact with a computer. This should result in natural and easy to learn interfaces.

• *Terse and Powerful interaction*: Devices such as the Dataglove capture 16 (non independent) dimensions. The theoretical throughput of such devices is very high compared to traditional devices such as the keyboard or mouse. This enlarged bandwidth should result in a higher power of expression.

• *Direct interaction*: With an ideal hand gesture input device, the hand would become *the* device. This makes it possible to emulate other devices (like the mouse or chord keyboards using finger alphabets). The user becomes free to wander around without any transducer device to carry and interact with the surrounding machinery by simple designation and adequate gestures.

Needless to say, no system exhibits all of these characteristics. Most people who have tried a system that uses hand gesture input have experienced frustration and disappointment. We see two main reasons for this:

• *"Immersion Syndrome"*: Since the system captures every motion of the user's hand, every gesture can be interpreted by the system, whether or not it was intended. The user can be cut from any possibility of acting or communicating simultaneously with other devices or persons. To remedy that problem, the system must have well defined means to detect the *intention* of the gesture.

• *Segmentation of hand gestures*: Gestures are continuous by essence. A system that interprets gestures to translate them into a sequence of commands must have a way of segmenting the continuous stream of captured motion into discrete "lexical" entities. This process is somewhat artificial and necessarily approximate. This is why most systems recognize steady postures instead of real gestures.

Hand gesture input also has intrinsic drawbacks that affect the usability of such systems:

• *Fatigue*: Gestural communication involves more muscles than keyboard interaction or speech: the wrist, fingers, hand and arm all contribute to the expression of commands. Gestural commands must therefore be terse and fast to issue in order to minimize effort. In particular, the design of gestural commands must avoid gestures that require a high precision over a long period of time.

• *Lack of comfort*: Current hand gesture input devices require wearing a glove and being linked to the computer, reducing autonomy. Fukumoto [9] uses a video camera to capture gestures in order to overcome this problem.

Despite the availability of devices and promising outcome, we found no use of hand gesture input that goes far beyond the laboratory experiment into real world applications. Although several algorithms and methods for interpreting hand gestures are now available, practical interaction techniques and "style guides" remain to be defined for hand gesture input.

Our research aims at providing usable applications of hand gesture input in the real world. By examining characteristics of the structure of gestural communication, we developed an interaction model that overcomes the immersion syndrome and segmentation problems presented above. This model includes guidelines for designing gestural command sets and a notation for gestural commands. The model was then evaluated by developing a sample application.

RELATED WORK

Most work on gesture based user interfaces concerns pen-based input, and commercial products are now available (such as the PenPoint system [6]). Morrel-Samuels [14] characterizes thoroughly the distinction between gestural and lexical commands (in the sense of textual or spoken commands). Though considering mainly 2D mark-based interfaces, most of his work also applies to 3D hand gesture interfaces. Rubine [13] has developed algorithms for 2D gesture recognition that we have used to analyze and classify 3D hand gestures.

Concerning hand gesture input, research is less advanced and has not reached the stage of commercial products. Three main directions have been investigated:
• *Virtual Reality Systems*, in which the user interacts mainly by means of direct manipulation of the objects of the application, presented as embodied physical objects [10]. Most works in this area merely present hand gesture recognition in the specific context of their application [2].
• *Multi-Modal Interfaces*, which aim at providing natural and powerful interaction by using the natural human-to-human communication means: speech combined with gesture and gaze (see for instance [4], [17]).
• *Recognition of Gestural Languages*, where the gestures are interpreted as commands. Deaf sign language recognition constitutes the main stream of those attempts (see for instance [12]). Other approaches consist in recognizing specific gestural commands. For instance, Sturman [16] presents a system that recognizes gestures for orienting construction cranes; Morita et al. [11] show how to interpret the gestures of a human conductor to lead a synthesized orchestra.

Our work fits in the last category, and is also of interest to multi-modal interfaces that wish to use gestural input. Our approach follows the concept of "embodied virtuality" [18], which aims at "augmenting" reality with computers instead of substituting computers to reality. Hence, it contrasts with virtual reality systems, which promote the notion of immersion.

The rest of the paper is organized as follows: we describe the interaction model; then we present the application used to evaluate the model. Finally, we discuss the results from the evaluation, other possible applications, and directions for future work.

INTERACTION MODEL

We define an interaction model as the description of the morphology (or structure) of the human-computer interface of an application. For example, command-line interfaces, direct manipulation [15] and iconic interfaces are interaction models. Despite important progress made in this area [7], we lack a formalism to describe interaction models. We will therefore use an informal description and a set of design rules.

We propose an interaction model based on hand gesture where the user is free to move and perform gestures in the real world . In order to distinguish between gestures that are addressed to the system and gestures that are not, we use the notion of *active zone*. The active zone is a 2D area, typically the projection of a computer screen on a wall. Gestures are interpreted only when the user designates this area, i.e. when the projection of the hand enters the active zone. Each gestural command is described by a start posture, a dynamics and an end posture. The intention to issue a command is detected when the user's hand points at the active zone *and* a start posture is recognized. The start and end postures do not require the hand to be steady, allowing commands to be input smoothly. If the user's hand leaves the active zone while performing a valid command, that command is issued. Hence, the final posture is optional.

In order to facilitate gesture recognition, we have assigned different roles to the different degrees of freedom of the hand (table 1):

Dimension	*Role*
Hand Position X, Y, Z	User
Hand Orientation Yaw, Pitch Roll (wrist orientation)	 User Segmentation
Finger Positions	Segmentation
Projection X, Y on active zone	(Segmentation) Classification
∂Z (distance from active zone)	Classification
∂Roll	Classification
∂Fingers	Classification

Table 1 - Role of each dimension.
∂ means the variation of a dimension.

• *"User"*: Since we want the user to be able to move around freely, the position and orientation of the hand in space cannot be taken into account directly. Rather, these dimensions are used to determine the projection of the hand on the active zone and the distance to it. Segmentation and classification occur only if this projection is within the active zone.
• *Segmentation* (recognition of start and end postures) use the wrist orientation and finger positions. The dimensions used for segmentation are quantized in order to make postures both easier to recognize by the system and more predictable by the user. We use 7 orientations of the wrist, 4 bendings for each finger, and 2 for the thumb. This gives theoretically 3584 postures, among which at least 300 can be obtained with some effort and between 30 to 80 are actually usable (depending on the user's skill and training).
• *Classification* of the different gestures according to their dynamics uses the movement of the projection of the hand, the rotation of the wrist, the movements of the fingers, and the distance from the hand to the active zone (allowing for push-like gestures).

Guidelines for Defining Gestural Commands

The description we have given so far delimits the set of commands that can be issued. However it does not help in defining a set of commands that provides natural interac-tion. We found that the gestural commands had to conform to a set of design rules, or guidelines, that we now present.

Power of Expression vs. Ease of Use

The choice of appropriate gestural commands results from a compromise between the selection of natural gestures, which will be immediately assimilated by the user, and the power of expression, in which more complex gestural expression gives the user more efficient control over the application. Of course, the notion of "natural" gesture depends heavily on the tasks to be performed: are common gestural signs easily applicable to meaningful commands ? Navigational tasks are quite easy to associate to gestural commands (move up, left, down, quit, go back, stop, select, etc.). Drawing or editing tasks may also have several significant natural gestures associated to them (draw a circle, draw a rectangle, remove this, move this here, etc.).

Abstract tasks are much harder to "gesturize" and require non symbolic gestures to be introduced (change font, save, etc.). Using deaf sign language vocabulary could be considered as an alternative and would have the advantage of benefiting an important community of people with disabilities. We have not tried to apply our interaction model to such abstract tasks, but obviously they would require important user training. It is probably not wise to use gestural commands for every task. A better solution that we wish to investigate is to complement gestural interaction with verbal commands, by means of speech recognition.

Ease of Learning

In order to increase the usability of the system, a general rule is to assign the most natural gestures, those that involve the least effort and differ the least from the rest position, to the most common commands. The users are then able to start with a small set of commands, increasing their vocabulary and proficiency with application experience.
Another key to the good design of gestural commands is the notion of *tense posture*. The start postures should require a certain tension of the muscles, such as full extension of the hand or clenching one's fist. This reduces the risk of misinterpreting gestures if the user waves the hand at the active zone, since the usual postures of the hand are relaxed positions. More importantly, this makes the user's intention of issuing a command more explicit or, as explained by Buxton [5], "it determines the ebb and flow of tension in a dialogue". The tension required for issuing a command can be extremely short and therefore should not generate fatigue. The end of a gestural command, on the other side, should correspond to a relaxed position. This is already the case when the hand leaves the active zone (e.g. when lowering one's arm). It should also be true of end postures.

Fast, Incremental, Reversible Actions

Gestures must be fast to execute and must not require too much precision in order to avoid fatigue. In particular, an aspect of prime importance when designing a gestural command set is the resolution of each dimension as captured by the input device. If the position of the hand cannot be determined with less than 1 cm of precision, precise tasks cannot be performed. For instance, the application should not rely on drawing fine details or manipulating objects smaller than a few centimeters.

It is also important to provide good user feedback. First, a lexical feedback, such as the shape of the cursor, should inform the user of the state of the recognition system. At a higher level, the feedback of the command itself should be related to the gesture used to issue it; for instance, if moving the hand from left to right goes to the next page of a document, an appropriate feedback is to have the next page wipe over the current one from left to right.

Finally, since recognition of a gesture can be wrong and commands can be issued involuntarily, the command set must provide an undo command or symmetric commands that let the user easily undo any unintended action.

Notation for Gestural Commands

When designing gestural command sets, the need for a notation of available commands becomes obvious: designers must be able to document commands for application users. We examined deaf sign language notations as a possible model, however these are usually incomplete and difficult to understand. Since our interaction model precisely delimits the types of gestures that can be recognized, we devised a simple and complete icon-based notation that should be easily understood by the layman.

Fig. 1 shows an example of this notation. We suppose here that the *right* hand is used for issuing commands. A gestural command is represented by a set of 3 icons. The first icon describes the start posture, the second one describes the dynamics of the gesture (the trajectory of each articulation), and the last icon shows the end posture. Start and end posture icons show the orientation of the wrist and the position of the fingers. The dynamics icon shows the trajectory of the projection of the hand, with optional marks to specify finger motions, if such motion is not implicitly defined by the differences between start and end posture icons.

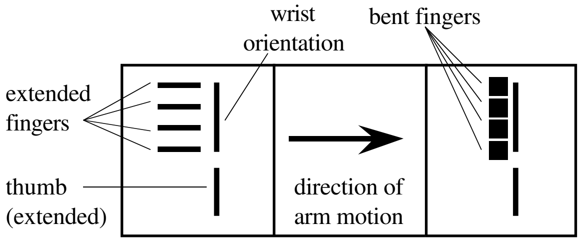

Figure 1 - "go NextChapter" gesture (for the right hand). When pointing at the active zone, this command is issued by orienting the palm to the right (thumb down), all fingers straight, and moving from left to right. The gesture can be completed by bending the fingers or moving the arm to the right until the projection of the hand leaves the active zone.

Only a subset of the notation is presented here. Additional marks in the start and end icons can specify restrictions of the active area. Other marks express the finger and wrist motions, as well as the variation in distance from the active area (to enable "button press"-like gestures).

EXAMPLE

In order to evaluate the interaction model, we have implemented a sample application that allows browsing in a hypertext system (namely HyperCard© on Apple Macintosh®). Our application shows what hand gesture input can bring to interactive systems and how it must respect the user's need to continue interacting and communicating in the real world, while still taking advantage of a computer ready to obey at the beck and call.

The system consists of a video projector showing the computer screen to the audience (fig. 2). The speaker wears a VPL Dataglove® and can issue commands by pointing at the screen and performing adequate gestures. By means of 16 gestural commands (fig. 3), the user can freely navigate in a stack, launch a slide-show, highlight parts of the screen, etc. For instance, moving the hand from left to right goes to the next slide, while pointing with the index and circling an area highlights part of the screen.

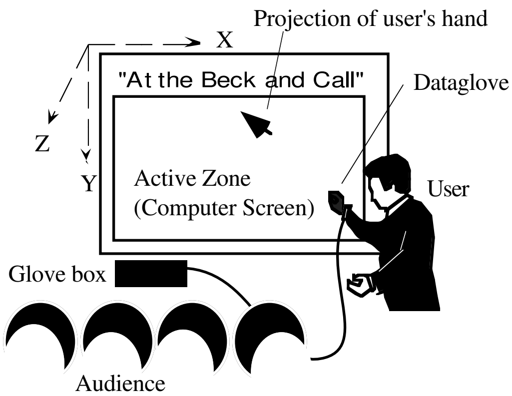

Figure 2 - Setup for the sample application.

Using gestures to navigate in the system enables the user to suit the action to the word: the gestural commands fit quite naturally the course of the presentation, and most gestures are actually performed at the limit of consciousness. This sense of control lets the user feel free to orient the presentation according to his or her feelings rather than follow the ordered set of slides. The user can still perform any action in the real world, since the gestures are interpreted only when the hand points to the screen. The user can even show and point at them, since only gestures known to the system will be interpreted as commands.

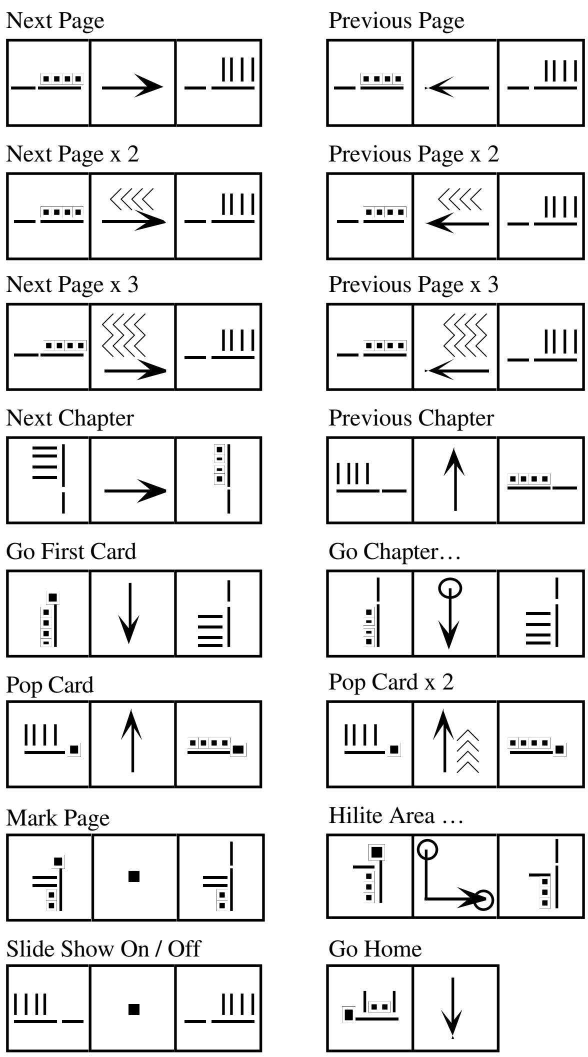

Figure 3 - Gestural command set for the sample application. Circles in the dynamics icons indicate that the start and/or end positions are used by the application. V-shapes in the dynamics icons indicate one or two finger bendings during the arm motion.

Implementation

The algorithm that parses samples received from the Dataglove is implemented as a driver on a Macintosh IIx. When a gesture is recognized, an event is sent by the driver to the active application (in this case HyperCard), containing information on the issued gesture (name, start and end position, etc.).

Segmentation is achieved by a tree search (fig. 4) that indicates whether the current sample corresponds to a posture to be recognized, taking advantage of the quantization of the dimensions used for segmentation of commands. All start postures that have the same wrist orientation are grouped in the same first branch. The following branches respectively indicate the thumb, index, and other finger positions. Given a hand configuration, no more than 6 lookups are required to determine if it corresponds to the start posture of one or more gestures. The same applies to end postures, which are stored in a second tree.

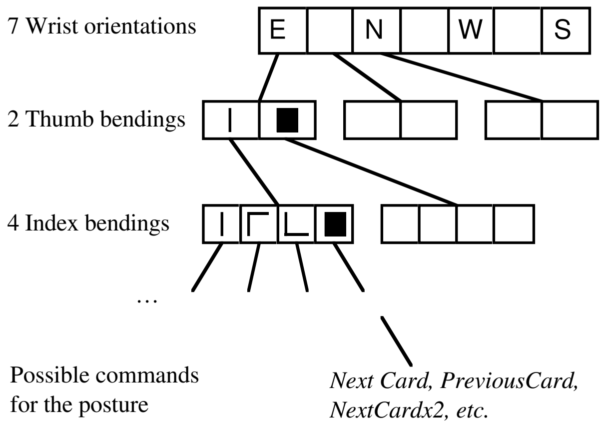

Figure 4 - Tree for recognizing start postures.

We use the algorithm defined by Rubine [13] to analyze the dynamics of gestures. This algorithm was designed for 2D gesture classification. It extracts features from the gestures, such as the total angle traversed, the total length of the path followed by the hand, etc. Mean values for each gestural command and each feature are determined by training the system when the application is designed. When a command is issued, the features characterizing the gesture are compared to the mean values for each possible command, determining which gestural command was meant by the user. In order to use this algorithm with full-hand gestures, we extended it by adding features for each finger bending, wrist orientation and distance from the active zone.

The Dataglove is sampled at 60 Hz. Processing of each Dataglove sample is in constant time, and no significant overhead of the driver has been observed. The driver uses 22Kbytes of code, and a typical command set uses 40 Kbytes of memory. A separate application is used to enter the command set by a combination of interactive dialogue for the start and end postures and data glove input for the dynamics. We have used an average of 10 training examples for each gestural command. This has proved sufficient to provide user-independent recognition.

DISCUSSION

In this section, we present the evaluation of the sample application, the lessons learned from this work, and the other applications that we foresee for the interaction model.

Usability Testing

We have conducted a usability test on the sample application presented in the previous section. First, we tried to assess the learning time of the command set and the recognition rate of the algorithm. Ten users were presented the application and the graphical notation of gestures. After trying each gesture once, they performed one series of 50 gestures. Each gesture was prompted by the system, and the recognition rate was computed as the proportion of gestures recognized correctly. Two subjects had a recognition rate around 50%: their hands were too small for the Dataglove we used. The other subjects had a recognition rate of 72% to 84% . A trained subject regularly obtains 90% to 98%.

We have characterized two main types of errors. First, gestures that differ only in their dynamics part are often mistaken, especially when finger bending is involved (such as PopCard and PopCardx2). This indicates that our adaptation of Rubine's algorithm should be tuned, although the lack of resolution of the Dataglove may also be responsible. The other type of error corresponds to hesitations while issuing a command. This often occurs when the user is tensed and has not integrated the interaction model. This problem disappears with a little training, when gestures are issued more naturally.

The second part of the usability test consisted in an "in vivo" use of the system. Two trained users made several presentations of the system to an audience, using the sample application. The purpose of this test was not to evaluate a recognition rate, but rather to determine whether the application was usable in a real setup. Most mistakes were noticed immediately and could thus be corrected in one or two gestures. In a few cases, the user did not immediately realized he had issued a command, or did not know which command had been issued, and it took somewhat longer to undo the effect of the command.

Overall, the error rate was surprisingly low, because the most usual commands are the most natural ones and are better recognized. As a result, the system did not show rejection by the users: they find the interface easy to use, and the interesting presentation effects are worth the (small) learning time of the application. Of course, due to the high cost of the device and installation, and the relative discomfort due to the loss of autonomy (the user is "linked" to the computer), we do not expect this application to become widespread in its current state.

Lessons Learned

The recognition of gestures in our interaction model allows for a natural interaction, at the expense of some constraints on the set of gestural commands: an end posture cannot also be a start posture; gestural commands cannot differ only by their end posture. However, we have found that these constraints were not much of a problem in practice, since the guidelines avoid them. For instance, start postures are tensed while most end postures are relaxed.

A more significant problem was due to the lack of precision of the hardware and setup that we used. First, the samples from the Dataglove are not stable even when the device is immobile. Second, since we use the projection of the hand on the screen, any instability (be it due to the hardware or to the user's arm) is amplified. In practice, the best resolution is about 10 pixels, which makes precise designation tasks impossible. Although filtering would aid, it would not relieve the problem of arm movements. Hence, it does not seem that this problem is likely to be solved within the interaction model. Precise tasks generally require a physical contact with a fixed stand, whereas our model is a hand-free remote manipulation paradigm.

When we started this work, we did not expect to be able to perform real-time recognition of gestures and run the application on the same machine. By characterizing the interaction model, we were able to devise a very simple recognition technique without significant loss in the power of expression. We even claim that such simplification enhances the model in that it makes it easier to learn and to use: using an active zone to address the computer and using tense postures to start gestural commands is similar to the use of gaze and pointing in human-to-human communication; quantizing dimensions makes the system more predictable.

Comparing our interaction model to the expected advantages of hand gesture input mentioned in the introduction, we have achieved natural and terse interaction in the sample application, as shown by user testing. This is mostly due to the careful design of the command set, and thus relates to the style guide part of the model. However, interaction is not necessarily powerful enough, depending on the tasks to be performed (abstract tasks would benefit from voice input as a complement to gestures). Finally, interaction is direct. The user is free to move and interact with the real world. However, other devices cannot be emulated as soon as precise interaction is required. These limitations should help to define which applications can benefit from the interaction model. We now present some of them.

Other Applications

We are in the process of developing another application of the interaction model. This application is an extension to the popular *twm* window manager for the X Window System. The software must be implemented on an Xstation with an LCD Display in order to prevent the magnetic field of the VDT from interfering with the Polhemus tracker that is part of the Dataglove. Gestural commands for moving, resizing, and iconifying windows should save a significant homing time when the user is typing on the keyboard, and also when the mouse is used, since these commands are usually activated by pointing with the mouse at small icons in the title bar of the windows.

Other applications of the interaction model include the following:

• *Multi-User Interaction & Large Panel Displays*: Elrod et al. presented a system to interact with Large Control Panels [8]. Air traffic control, factories, stock exchange and security services all use control rooms in which the workers have to inspect large panels of controls and displays collectively. Our interaction model could improve the user interface of these rooms by allowing easy remote control of the displays by means of designation and gestural commands. The use of gestures is here particularly indicated because designation works in a noisy environment.

• *Multi-Modal Interfaces*: Pure speech-based interfaces also face the "immersion syndrome". Combining gestural commands with speech would improve both media: speech would complement gesture to express abstract notions, and gesture would complement speech to designate objects and input geometric information. Furthermore, gestures can be used with our model to detect intention of speech, i.e., distinguish speech addressed to the system from utterances to the "real world".

• *Home Control Units*: In the longer term, we foresee the remote control of home or office devices: a few cameras linked to a central controller would track the gestures and recognize every intent to use a device such as TV, hi-fi system, answering machine, etc. This would avoid the proliferation of remote control units that are cumbersome to use and hard to find whenever they are needed.

CONCLUSION AND FUTURE WORK

We have described an interaction model that uses gesture input in a way compatible with the real-world. A sample application of this model has been implemented and evaluated, validating two main points of the model: the notion of *active zone* to avoid the immersion syndrome, and the idea of using *tense postures* for segmenting gestures.

This application, together with other applications that we foresee for this model, leads to three areas for future work. The first one consists of tuning the current implementation in order to obtain a better recognition of the dynamics part of gestures and a better accuracy of the Dataglove data (by means of filtering). The second one is to replace the Dataglove by cameras to recognize gestures, in order to have a less intrusive input device. The last area concerns multimodal interfaces, in particular the use of speech to complement gestures. This requires an extension of the interaction model, and in particular a re-design of the guidelines in order to define a set of commands (gesture+speech) for a given application.

ACKNOWLEDGMENTS

This work was conducted while the first author was at LIMSI. We thank J. Mariani, F. Néel and G. Sabah from LIMSI for making this research work possible. H. Levy helped improve the readability of this article.

REFERENCES

1. Adams, D. *The Hitch Hiker's Guide to the Galaxy.* Pan Books Ltd., London, 1979, Chapter 12.
2. Appino, P., Lewis, J., Koved, L., Ling, D., Rabenhorst, D. and Codella, C. *An Architecture for Virtual Worlds*, Presence, 1(1), 1991.
3. Bolt, R.*"Put-That-There": Voice and Gesture at the Graphics Interface*, Computer Graphics, 14(3), July 1980, pp 262-270, Proc. ACM SIGGRAPH, 1980.
4. Bolt, R. *The Human Interface*, Van Nostrand Reinhold, New York, 1984.
5. Buxton, W. There's More to Interaction than Meets the Eye: Some Issues in Manual Input. in Norman, D.A. and Draper, S.W. (Eds.), *User Centered System Design*, Lawrence Erlbaum Associates, Hillsdale, N.J., 1986, pp. 319-317.
6. Carr, R. *The Point of the Pen*. Byte, February 1991, pp. 211-221.
7. Card, S., Mackinlay, J. and Robertson G. *A Morphological Analysis of the Design Space of Input Devices*. ACM Transactions on Information Systems, Vol. 9, No. 2, April 1991, pp. 99-122.
8. Elrod, S., Bruce, R., Goldberg, D., Halasz, F., Janssen, W., Lee, D., McCall, K., Pedersen, E., Pier, K., Tang, J. and Welch, B. *Liveboard: A Large Interactive Display Supporting Group Meetings and Remote Collaboration,* CHI'92 Conference Proceedings, ACM Press, 1992, pp. 599-608.
9. Fukumoto, M., Mase, K. and Suenaga, Y. *"Finger-pointer": A Glove Free Interface,* CHI'92 Conference Proceedings, Poster and Short Talks booklet, page 62.
10. Krueger, M., *Artificial Reality (2nd ed.)*, Addison-Wesley, Reading, MA, 1990.
11. Morita, H., Hashimoto, S. and Ohteru, S. *A Computer Music System that Follows a Human Conductor*. IEEE Computer, July 1991, pp.44-53.
12. Murakami, K. and Taguchi, H. *Gesture Recognition Using Recurrent Neural Networks*, CHI'91 Conference Proceedings, ACM Press, 1991, pp. 237-242.
13. Rubine, D. *The Automatic Recognition of Gestures*, Ph.D. Thesis, Carnegie-Mellon University, 1991.
14. Morrel-Samuels, P. *Clarifying the Distinction Between Lexical and Gestural Commands,* International Journal of Man Machine Studies, Vol. 32, 1990, pp. 581-590.
15. Shneidermann, B. *Direct Manipulation: A Step Beyond Programming Languages*, IEEE Computer, August 1983, pp. 57-69.
16. Sturman, D. *Whole-Hand Input*, Ph.D. thesis, Media Arts & Sciences, Massachusetts Institute of Technology, 1992.
17. Thorisson, K., Koons, D. and Bolt R. *Multi-Modal Natural Dialogue*, CHI'92 Conference Proceedings, ACM Press, 1992, pp. 653-654.
18. Weiser, M. *The Computer for the 21st Century*, Scientific American, September 1991.
19. Zimmerman, T. and Lanier, J. *A Hand Gesture Interface Device*. CHI'87 Conference Proceedings, ACM Press, 1987, pp. 235-240.