Special issue of Communications of the ACM on Computer-Augmented Reality, July, 1993

CHARADE : REMOTE CONTROL OF OBJECTS USING FREE-HAND GESTURES

Thomas Baudel and Michel Beaudouin-Lafon

CONTACT ADDRESS

Laboratoire de Recherche en Informatique - CNRS URA 410
Bâtiment 490, Université de Paris-Sud
91405 Orsay Cedex - FRANCE
33+ 1 69 41 69 10
e-mail : thomas@lri.fr, mbl@lri.fr

NOTES TO THE TYPESETTER

- possible "pull quotes" are suggested in outlined type within the text.
- footnotes are *not* part of the article: they contain comments or suggestions for the typesetter.
- acknowledgment, list of references and short bios of the authors are at the end of the text.

KEYWORDS

ACM: H.5.2, I.3.6

Other keywords: Hand gesture input, Interaction model, Augmented reality, Remote control, Computer-aided presentations.

TEXT OF THE ARTICLE:

> *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 [Ada79]

It is easy to imagine using free-hand gestures to control objects in the real world but only recently have researchers attempted to make this a reality. Since the early work of Myron Krueger (see sidebar) and Richard Bolt [Bol80], two paradigms have been explored. The *manipulation paradigm* lets users manipulate objects directly using hand gestures in an application-specific way [App91, Viv91]. For example, Virtual Reality systems use this approach by presenting synthesized pseudo-physical objects to the user. The *sign language paradigm* lets users issue commands with hand gestures. Primary applications include multi-modal interfaces [Bol84] and deaf sign language recognition [Mur91]. Other systems recognize specific gestural commands. For example, Sturman [Stu92]

presents a system for orienting construction cranes with gestures and Morita et al. [Mor91] present a system for conducting an orchestra of synthesizers.

The expected advantages of using free hand gestures for interaction are:

• *Natural interaction*: Gestures are a natural form of communication and and are easy to learn.

• *Terse and powerful interaction*: A single gesture can be used to specify both a command and its parameters. The position and movements of the hand and fingers provide the potential for a higher power of expression.

• *Direct interaction*: The hand becomes *the* input device, eliminating the need for intermediate transducers. The user can interact with the surrounding machinery by simple designation and appropriate gestures.

Unfortunately, research laboratories have yet to produce an application that provides these advantages. Some reasons are intrinsic to gestural communication:

• *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 concise and quick to issue in order to minimize effort. In particular, the design of gestural commands must avoid gestures that require high precision over a long period of time.

• *Non self-revealing*: The user must know the set of gestures that the system recognizes. Hence, gestural commands should be simple, natural, and consistent. Appropriate feedback is also of prime importance.

Other reasons are due to limitations in the current technology and recognition techniques:

• *Lack of comfort*: Current hand gesture input devices require wearing a glove and being linked to the computer, reducing autonomy. Using video cameras and vision techniques to capture gestures [Fuk92] will eventually overcome this problem.

• *"Immersion Syndrome"*: Most systems capture every motion of the user's hand. As a consequence, every gesture can be interpreted by the system, whether or not it was intended, and the user is cut off from the possibility of communicating simultaneously with other devices or people. A hand gesture input system must therefore provide well-defined means to detect the *intention* of the gestures.

• *Segmentation of hand gestures*: Gestures are by nature continuous. A system that interprets gestures and translates 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 positions instead of dynamic gestures.

We developed Charade [Bau93] in order to investigate free-hand gesture input in the real world. Using Charade, a speaker giving a presentation can control a remote computer display with free-hand gestures while still using gestures for communicating with the audience. We describe the techniques and guidelines we have developed for Charade and explain how they apply to augmented reality environments.

CHARADE: HAND-GESTURE INPUT IN THE REAL WORLD

Computer-based presentations have several advantages over slides and overheads. The order of the presentation is not fixed, the production process is simpler, and last minute changes are easily made. Presentations can be enhanced with audio, video and even interactive programs. Unfortunately,

operating such systems is usually more difficult than using slides or overheads. The speaker has to use multiple devices (e.g. keyboard, mouse, VCR remote control) with unfamiliar controls. These devices are hard to see in the dark and operating them disrupts the course of the presentation.

Charade uses hand gestures to control computer-aided presentations. Our prototype (photo 1) allows browsing in a hypertext system (HyperCard™). The display of an Apple Macintosh™ is projected onto a screen. We call the projection of the display the *active zone*. A VPL DataGlove™ [Zim87] is connected to the serial port of the Macintosh. The DataGlove can measure the bendings of each finger and the position and orientation of the hand in 3D space relatively to the active zone. Gestures are recognized by an algorithm that runs in real time and leaves sufficient CPU power to run the presentation software on the same machine [Bau92, Bau93].

The user wears the DataGlove to interact with the system. When the pointing direction of the hand intersects with the active zone, a cursor appears on the screen and follows the hand. The speaker can issue commands by pointing at the active zone and performing gestures. The same gestures will be ignored if issued with an orientation of the hand that points outside of the active zone. The user can issue 16 gestural commands to navigate through the hypertext and conduct the presentation. For example, moving the hand from left to right goes to the next slide, while pointing with the index finger and circling an area highlights part of the screen.

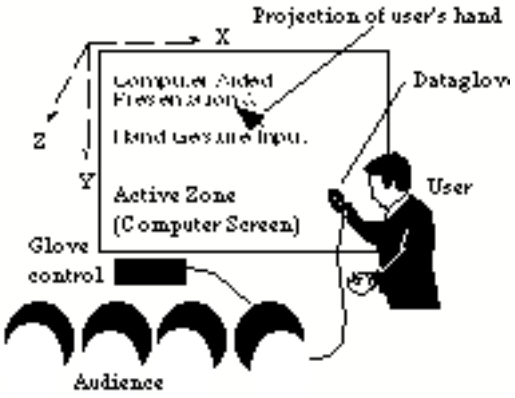

Photo 1 : Setting of Charade [1]

Gestural commands fit naturally in the course of the presentation; the presenter can remain barely conscious of the use of gestures as commands, making for a seamless interaction technique. The user can freely orient the presentation according to current needs rather than follow a pre-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 point at the screen since only gestures known to the system are interpreted as commands.

Usability Testing

We conducted an experiment to assess the learning time of the command set and the raw recognition rate of the algorithm. The recognition rate for first time users was 72% to 84%, while a trained subject regularly obtained 90% to 98%. We found two main types of errors: system errors and "user" errors. The system had difficulties identifying gestures that differ only in their dynamic phase, especially

[1] (*** see mock-up on separate sheet ***)

when finger bending is involved (such as “Pop Card” and “Pop Card x 2”). Finer tuning of our algorithm may remove these errors. “User” errors correspond to hesitations while issuing a command. This often occurs when the user is tense and has not had enough practice with the system. This problem disappears with a little training, when gestures are issued more naturally.

We also used Charade “in vivo” in order to iteratively design the interaction technique and determine whether the application was actually usable. Two trained users used Charade to make several presentations of the system to an audience. The error rate was surprisingly low (a few errors per presentation), because the most usual commands are the most natural ones and are better recognized. Most mistakes were noticed immediately and could thus be corrected in one or two gestures. In a few cases, the user did not immediately realize he had issued a command or did not know which command had been recognized, and it took somewhat longer to undo the effect of the command. Overall, the users found the interface easy to use and the small learning time was worth the improvements in the presentation.

LESSONS LEARNED

Experimenting with a real application led us to define constraints to ensure the usability of free-hand gesture input in the real world. These constraints are implied by the media and the context of use. They aim at avoiding the immersion syndrome, minimizing the fatigue generated by arm motion and solving the segmentation problem. We first describe the *rules* that define the general structure of gestural interaction. Then we present *guidelines* for designing gestural command sets.

Structure of Gestural Commands

Our model is based on the notions of active zone and gestural command. The *active zone* is a 2D area; in Charade it is the projection of a computer display on a wall. Gestures are interpreted only when the user’s hand points at the active zone. Each *gestural command* is described by a start position, a dynamic phase and an end position. The user issues a command by pointing at the active zone, adopting a start position and moving according to the dynamic part. Commands are ended either by leaving the active zone or by adopting an end position. These positions do not require the hand to be steady, allowing fast and smooth input.

The recognition of a command involves three steps:
• *Detection of the intention*. Since gestures are interpreted only when the hand is pointing at the active zone, the user can perform various gestures to communicate in the real world. It is also possible to use several active zones to command several systems.
• *Segmentation of gesture*. Start and end positions are defined by the wrist orientation (palm up, down, right or left) and finger positions. These dimensions are quantized in order to make hand positions both easier to recognize by the system and more predictable by the user. Depending on the user’s skill and training, between 30 to 80 hand positions can be recognized efficiently.
• *Classification*. Gestures are classified according to their start position and dynamic phase. The dynamic phase interprets the movement of the wrist and fingers to distinguish among commands. For example, our application uses the same start position to go to the next and previous pages; the direction of the gesture indicates which page to go to. In addition, opening the hand once or twice during the movement skips one or two pages.

We impose two more constraints to increase the usability of the system: all start positions must differ from all end positions and commands must not differ solely by their end position. The first

constraint allows commands to be issued smoothly: users do not have to hold their hands steady or stop between commands and can issue multiple commands with a single movement. The second constraint allows users to complete a command either by using an end position or by leaving the active zone. The latter can be achieved by lowering the arm, which is more restful and natural.

Designing Gestural Command Sets

The previous rules define the design space for gestural command sets. The following guidelines identify characteristics that provide natural interaction. Designing a specific command set for a particular application will also require an iterative design process and testing with users.

• *Use Hand Tension*: Start positions should be *tense*: they should correspond to non-relaxed configurations of the fingers and wrist, e.g. full extension of finger joints, clenching the fist or orienting the palm up. Since the tense period is short, it does not generate fatigue. Tension in hand positions makes explicit the user's intention to issue a command. Buxton [Bux86] argues that this emphasizes the structure of the human-computer dialogue in the same way as intonation distinguishes ordinary conversation from imperative orders. Using tension allows the user to move and perform gestures in the real world, since only specific and clearly intentional gestures are interpreted by the system. Conversely, end positions should *not* be tense. This happens naturally when lowering the arm and leaving the active zone: the arm's muscles become relaxed, indicating the completion of a command.

• *Provide Fast, Incremental, Reversible Actions*: The principles of direct manipulation [Shn83] apply to our remote interaction paradigm: gestures must be quick to execute to avoid fatigue; appropriate and continuous feedback must be provided to improve the user's confidence in the system; commands must be easy to undo, to easily cancel any unintended action.

• *Favor Ease of Learning*: A compromise must be made between natural gestures that are immediately assimilated by the user and complex gestures that give more control. We assign the most natural gestures to the most common commands. Natural gestures are those that involve the least effort and differ the least from the rest position. Users are thus able to start with a small set of commands, increasing their vocabulary and proficiency with experience with the application.

• *Use Hand Gestures for Appropriate Tasks*: Navigational tasks can be easily associated with gestural commands. For example, the hand should move upward for a "move up" command. Widely-used iconic gestures (e.g. stop, go back) can be associated with corresponding commands. Drawing or editing tasks also have natural gestures associated with them (e.g. select, draw a circle, move this here). Other tasks (e.g. change font, save) require non-symbolic gestures. One can use indirect selection gestures, in a way similar to menus in direct manipulation interfaces, but this detracts from the directness of the interaction technique. Instead, we recommend using speech input to complement gestural commands. Finally, free-hand gesture input should not be used for tasks that require precise interaction: when operating at a distance, we cannot obtain resolution higher than a few centimeters because of the instability of the hand in free space. Precise tasks, such as drawing or playing an arcade game, require physical contact to provide appropriate kinesthetic feedback, whereas we use a free-hand remote interaction paradigm.

Notation for Gestural Commands[1]

Since gestural commands are not self-explanatory, we have developed a notation that takes advantage of the characteristics of our interaction model. A gestural command is represented by a set of 3 icons (Figure & Photo 2): start position, dynamic phase and optional end position. Start and end position icons depict the configuration of the hand. The dynamic phase icon describes the movement of the hand, fingers and wrist. Circles along the trajectory indicate that the position of the projection of the hand in the active zone is a parameter of the command. For instance, in the "Highlight Area" gesture, start and end positions indicate the rectangle that should be highlighted. Gestural command sets can be redefined by the user. Figure 3 shows the commands chosen by the first author.

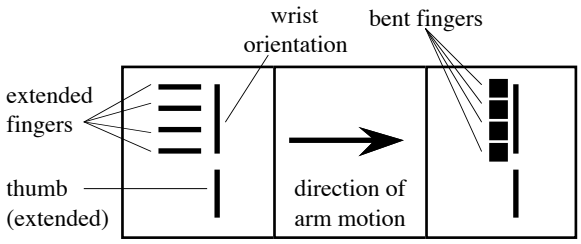

Figure & Photo 2 - "Next Chapter" gesture (for the *right* hand). When pointing at the active zone, this command is issued by orienting the palm to the right (thumb down), all fingers straight, and moving from left to right. The gesture can be completed by bending the fingers or moving the arm to the right until the projection of the hand leaves the active zone.[2]

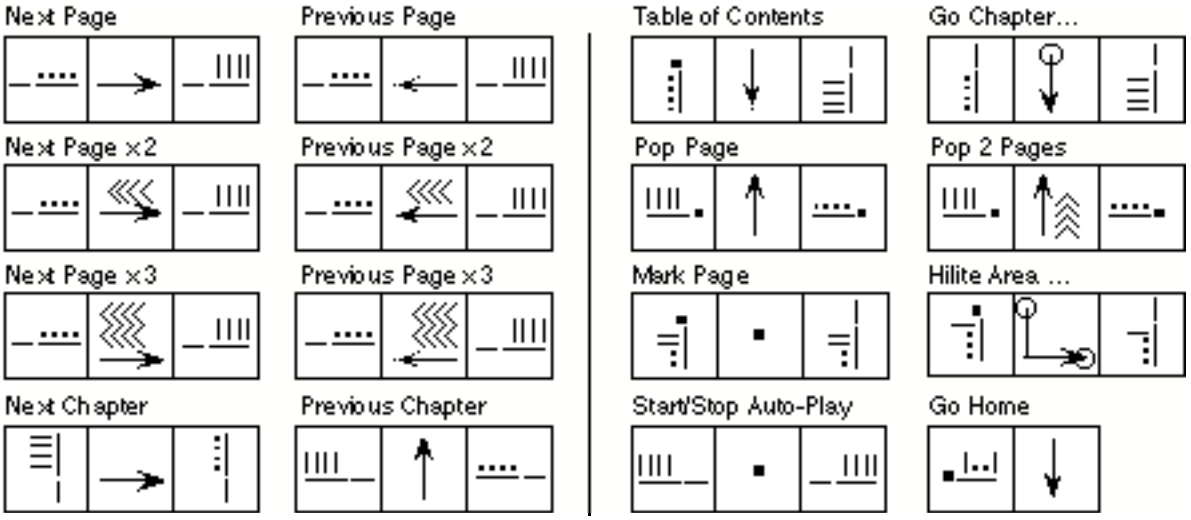

Figure 3 - a gestural command set for Charade. V-shapes indicate one or two finger bendings [3]

[1] (*** this section is autonomous. It can be put in a sort of sidebar within the article. ***)

[2] (*** put this picture in parallel with the photo showing the gesture being made (see mockup 2) ***)

[3] (*** this figure can be redrawn, provided that the interior of the rectangles be respected. It is also important that the idea of two columns for presenting the gesture set be respected, to enhance the relationship between paired gestures (e.g. next page & previous page…). The order in which the gestures are presented is also quite meaningful ***)

CONCLUSION AND FUTURE WORK

People are highly skilled in using gestures to communicate; yet few applications let them use gestures to control objects in the real world. We have introduced a novel interaction technique that allows users to take advantage of their natural skill with gesture to control computerized objects and avoid the immersion syndrome. Not only can they avoid using transducers, such as remote controls, but they can easily combine normal, expressive gestures with gestures intended as commands. Furthermore, this interaction technique can be implemented efficiently with off-the-shelf hardware and software.

Charade illustrates our approach using three key concepts:

- Definition of an active zone to distinguish gestures addressed to the system from other gestures;
- Recognition of dynamic gestures to ensure smooth command input;
- Use of hand tension to structure the interaction and make the user's intention explicit.

The first version of Charade was designed to assess the effectiveness of free-hand gesture input for real applications. We simplified the capture of gestures by using a DataGlove, even though this is ultimately not viable for day-to-day use of the system. Other systems [Kru90, Wel91] use video cameras to capture gestures. We plan to combine such techniques with our approach to create new kinds of applications, particularly within the domain of computer-augmented reality.

Most computer-augmented reality systems concentrate on output issues, such as registering a projected image with a real object (see Wellner's article in this issue). We have provided an input technique that distinguishes between gestural commands and other free-hand gestures. Active zones are a simple yet powerful way to register real objects, which can then be operated remotely. Visual or auditory feedback can be conveyed in the usual ways with the real object, or by projecting the information on the object. This superposition technique can also display virtual objects that can be controlled remotely. Sample applications include:

- Multi-User Interaction & Displays: Manufacturing plants, stock exchanges and security services have control rooms where workers share large panels of controls and displays. Free-hand gesture input could improve the user interface by allowing collaborative remote control of the displays. Gestures are particularly useful in these noisy environments. Two-handed gestures and gestures from multiple people may result in conflicts; we address related issues in [Bea92].

- Home Control Units: Appliances in the home or office, such as televisions, VCRs, stereos, answering machines, lights and radios, require frequent operation. Free-hand gestures would allow users to avoid searching for a particular remote control and still engage in their normal activities.

Which brings us back to the scenario described in the beginning of this paper. We have offered a solution to Douglas Adams's problem: his radio was confused because it could not differentiate between commands and ‘normal’ gestures. By properly segmenting the gestures and providing multiple methods of signaling the end of a command, we offer a somewhat more optimistic view of the future.

REFERENCES

[Ada79] Adams, D. *The Hitch Hiker's Guide to the Galaxy.* Pan Books Ltd., London, 1979, Chapter 12.

[App91] Appino, P., Lewis, J., Koved, L., Ling, D., Rabenhorst, D. and Codella, C. *An Architecture for Virtual Worlds*, Presence, 1(1), 1991.

[Bau92] Baudel, T., Beaudouin-Lafon, M., Braffort, A. & Teil, D. *An Interaction Model Designed for Hand Gesture Input*. LRI Research Report n° 772, September 1992.

[Bau93] Baudel, T. & Braffort, Annelies. *Reconnaissance de Gestes en Environnement Réel*. in Proceedings of Montpellier'93, Interface to Real and Virtual Worlds, Montpellier, France, March 1993.

[Bea92] Beaudouin-Lafon, M. & Karsenty, A. *Transparency and Awareness in a Real-Time Groupware System*. in Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'92), Monterey, CA, November 15-18,1992, pp 171-180.

[Bol80] 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.

[Bol84] Bolt, R. *The Human Interface*, Van Nostrand Reinhold, New York, 1984.

[Bux86] 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.

[Fuk92] Fukumoto, M., Mase, K. and Suenaga, Y. *"Finger-pointer": A Glove Free Interface,* in Proceedings of Human Factors in Computing Systems (CHI'92), Poster and Short Talks booklet, page 62.

[Kru90] Krueger, M. VIDEOPLACE and the Interface of the Future. in Brenda Laurel (Ed.), *The Art of Human-Interface Design*, Addison-Wesley, 1990, pp 405-416.

[Mor91] Morita, H., Hashimoto, S. and Ohteru, S. *A Computer Music System that Follows a Human Conductor*. IEEE Computer, July 1991, pp.44-53.

[Mur91] Murakami, K. and Taguchi, H. *Gesture Recognition Using Recurrent Neural Networks*, in Proceedings of Human Factors in Computing Systems (CHI'91), ACM Press, 1991, pp. 237-242.

[Shn83] Shneidermann, B. *Direct Manipulation: A Step Beyond Programming Languages*, IEEE Computer, August 1983, pp. 57-69.

[Stu92] Sturman, D. *Whole-Hand Input*, Ph.D. thesis, Media Arts & Sciences, Massachusetts Institute of Technology, 1992.

[Viv91] The Vivid Group.*Vivid's Group Mandala system* CHI'91 Interactive Experience, SIGGRAPH'92 art show.

[Wel91] Wellner, P. *The DigitalDesk Calculator: Tangible Manipulation on a Desk Top Display*. in Proceedings of the ACM Symposium on User Interface Software and Technology (UIST'91), ACM, New York, pp 27-33.

[Zim87] Zimmerman, T. and Lanier, J. *A Hand Gesture Interface Device*. in Proceedings of Human Factors in Computing Systems (CHI'87), ACM Press, 1987, pp. 235-240.

ACKNOWLEDGMENTS

This work was conducted while the first author was at LIMSI. We thank J. Mariani, F. Néel, G. Sabah and D. Teil from LIMSI for making this research work possible. A. Braffort participated in the design and implementation of Charade. We thank W. Mackay for discussions on this work and for her insightful comments on previous versions of this article.

ABOUT THE AUTHORS

Thomas Baudel is a PhD student in Computer Science at University of Paris-Sud. His research concerns the study of general methods for describing and implementing new interaction techniques.

His main interests are multi-threaded, multi-modal and gesture-based interfaces. *Author's Present Address*: LRI, Bâtiment 490, Université de Paris-Sud, 91405 Orsay Cedex, France. e-mail: thomas@lri.fr.

Michel Beaudouin-Lafon is a Professor in Computer Science at University of Paris-Sud. He is leading a research group on user interface engineering. The domains covered by the group are direct manipulation applications, CSCW, groupware systems and multi-modal interfaces. *Author's Present Address*: LRI, Bâtiment 490, Université de Paris-Sud, 91405 Orsay Cedex, France. e-mail: mbl@lri.fr.