Incorporating Speech Input into Gesture-Based Graphics Applications at The Brown University Graphics Lab

Robert C. Zeleznik, Joseph J. LaViola Jr., and Andrew S. Forsberg

Brown University Site of the NSF Science and Technology Center
for Computer Graphics and Scientific Visualization
PO Box 1910, Providence, RI 02912 USA
Email: {bcz,jjl,asf}@cs.brown.edu

Abstract

In this paper, we discuss The Brown University Computer Graphics Lab's ongoing effort to develop gesture-based graphics applications that are augmented by speech input. We focus on the Sketch project that has incorporated a variety of gestural interaction styles and also simple, effective speech input. The paper also states our positions on when visual/verbal interfaces are useful, our principles and methedologies for incorporating voice input into gesture based applications, and our reccomendations on research directions for visual/verbal interfaces.

1 Introduction

Over the past few years our research focus has been to push the envelope of gestural user interfaces and only recently has shifted to also incorporate speech input. Our initial motivation for gesture-only interfaces derived from an application focus on 3D modeling where we view a near ideal interface as paper and pencil sketching[4]. When sketching, there are essentially no user interface modes to interfere with the expression of a shape. Dissimilarly, the prevalent user interface for creating computer 3D models involves an extensive, ongoing dialogue with the user carried out via a plethora of user interface elements, including menus, icon bars, 2D and 3D widgets, and numerous interaction modes. As our interfaces have evolved, we have identified specific limitations to gesture-only interfaces and have begun to use speech input as the principal remedy.

2 Application Background

The striking difference between pencil sketching and computer modeling can be traced in large part to the distinction that pencil sketches are only 2D images, and thus do not have to continuously update a valid 3D model in the face of incomplete or ambiguous input. Thus to simplify the process of deriving a 3D model from user input, computer modeling techniques transform the model creation problem from that of depicting a 2D image to that of sequentially executing very specific operations, each of which is defined to produce complete, unambiguous output suitable for representing a 3D model. The net effect of these two fundamentally different approaches is that pencil sketching can rely on essentially a single skilled user task - drawing - where computer modeling generally relies on a plethora of "unskilled" user tasks that require a knowledgeable user to select the right task at the right time.

In contrast to previous approaches to 3D modeling, where each operation is visually represented in the user interface (as a button, menu item, etc), our goal in Sketch was to attempt to reconcile the look and feel of pencil sketching with the command orientation of conventional 3D modelers. In essence, we wanted to explore the possibility of replacing explicit visual user interface elements with gestural elements that a user has to know a priori. The result of our research is a range of gesture-based interfaces for tasks including CSG modeling[8], architectural design layout, mechanical CAD[1], and, interestingly, entering common music notation[2].

Our primary application, Sketch[8], is an entirely gesture based application that uses only a 3-button mouse and a single modifier key. Through this gesture interface, Sketch allows a user to construct a 3D model by directly specifying a primitive (from an extensive set that includes cubes, cones, pyramids, spheres, extrusions, revolves), its 3D position and orientation, and whether it is a positive volume or a negative (subtracted) volume. In addition, Sketch enables manipulation of all 6 degrees of freedom for the camera and for each object in the scene, in addition to manipulation of object features such as the polygonal outline of an extrusion. Finally, Sketch allows object movement to be constrained to translation along a single axis or plane, or to rotation about a specific axis.

The importance of Sketch is that the large set of distinct operations required to support its functionality were compressed into a manageable gesture system. Through this compression, the look-and-feel of Sketch resembles in some ways the pencil sketching ideal. However, Sketch is not without its shortcomings, which include limitations on the extensibility of the gesture set, modality of the multi-stroke gesture technique, incompleteness of operations, and the learning required to master the gesture set.

3 Principles and Methodologies

We developed a series of related systems, each with a different focus, to improve our understanding of gesture interfaces. Our goal was to isolate aspects of gestural interaction and explore a range of alternatives to understand their effectivness and applicability.

To address extensibility, we explored variations to our core gesture interfaces including widgets, mnemonics, two-handed input, and context-sensitivity. For many operations, abstract gestures are not appropriate and visual feedback is essential. For example, it is unreasonable to choose a color for an object through a single gesture. Instead, a color wheel or similar widget allows a user to visualize a space of colors in order to explore and choose a specific one. However, by incorporating widgets, we encountered the same problem that arises in the conventional 3D modeling application – how do you get the right widget to be on the display at the right time? Our approach to invoking widgets and abstract operations was to associate their invocations with their orthographic counterparts. For example, to display the color wheel, the user gestures the letter 'C', and to invoke a snapping mode the user draws the letter 'S'. A different approach to extending the functionality of a gesture system is to support two handed interactions[7]. The use of two hands effectively increases the amount of context that can be provided for each operation. For example, when using two hands to control two independent cursors, it is a simple matter to specify both ends of a line segment simultaneously – something which can't be done with a single cursor. A final technique for extending the set of available operations was to dynamically change the gesture set or mapping depending on where the user gestured in the 3D environment. For example, when drawing over a floor, the gesture set maps to objects like chairs and tables, whereas gesturing over a wall maps to windows and shelves[6].

The predominant type of gesture used in Sketch is a multi-stroke gesture. That is, each complete gesture in Sketch that represents an operation is composed of a series of distinct sub-gestures that we call strokes. The advantage of a multi-stroke gesture system is that, similar to the way that words in a language are composed of letters, a greater variety of gesture operations can be supported with a simple set of strokes. The disadvantage of multi-stroke gestures is that they effectively impose modes on the interface where a new gesture can't begin until an old one is finished or aborted, and it's up to the user to keep track of whether an old gesture has been finished or not. Thus, we explored single-stroke gestures as an alternative. We found that when gestures were used to represent abstract operations or mnemonics, single-stroke gestures were very effective. However, when gestures directly encode complex geometric shape information, such as when drawing a 3D primitive like a cuboid, we found that single-stroke gestures were more difficult to design and harder to execute than their multi-stroke counterparts.

Finally, although our Sketch system is extensive and has been used, for example in mechanical CAD applications to completely design a fabricatable metal part[1], there are still many operations that aren't easily captured in a gesture-only interface. In addition, as the complexity of the gesture-only interface grows, the amount of time required to learn the system also increases. Thus, to address the issues of complexity without making the system too complicated for new users, we have begun to apply speech input.

4 Applicability of Speech

An obvious situation where speech is appropriate within Sketch is as an alternative to gestural mnemonics. Thus, instead of drawing the letter 'C', a user may instead speak any of a range of equivalent phrases from "Give me a Colorwheel, please" to "Colorwheel"[4]. We provide speech as an alternative to gesture mnemonics because we have not found that speaking to invoke widgets is as fast or as natural in *all* situations as gesturing.

A second situation when speech is particularly appropriate in Sketch arises especially in the two-handed interface when the user is in the middle of an operation and wants to perform some other operation that either doesn't disrupt or that modifies their gestural interaction[5]. One example is to save the state of the application at various stages during a continuous interaction. Another example is to modify the snapping behavior of a line drawing interaction so that if a line is snapping to an unwanted surface, the user can say "stop snapping", or "stop snapping to that surface".

A third situation where we believe speech might be appropriate in Sketch is to improve the accuracy of the gestural interaction. In isolation, the imprecision of gestural input induces ambiguity in the gesture recognition process, just as speech interpreted in isolation generally cannot be interpreted without ambiguity. Thus for example, there are situations in Sketch where a user may draw an extrusion line for a specific primitive, but due to pixel-level imprecision, their gesture may be interpreted as some other operation. If either in response to a mis-interpreted gesture or as a general rule when drawing detailed gestures the user additionaly spoke something like "and now I extrude this", we could improve the accuracy of the gesture recognition.

A final situation that we have not explored but think will be important for Sketch is to use speech input to enable the user to interact in higher level ways. For example, currently a user can save a view of a scene and retrieve it only if it is given a special name or by sequencing through all saved views. Instead, we would like the user to be able to say something like "go back to the view I had a second ago, when the model was oriented like this and that cube was still purple." This form of interaction would require fundamental changes to both how we understand speech input and to the design of our system with regard to how we store and access our models.

5 Research Barriers

To achieve major advances in our gestural interfaces for 3D modeling, we identify three grand challenges:

First, there must be fundamental improvement in basic techniques for capturing 3D gestures and recognizing spoken words without lag, especially as we move to more immersive environments like a Cave. Current speech recognition is too slow, not sufficiently speaker independent, subject to environmental noise, invasive, and not adequately able to identify a particular speaker or tone of voice. Similarly, although gesture recognition of pen strokes is adequate, just capturing the raw data of 3D handwaving gestures is at best rudimentary.

Second, we feel that speaking to a computer is unnatural and uncomfortable, especially for long periods of time. Thus to support serious applications that will be used for eight hours a day, we think the issue of humanizing the computer is critical. To do this, we need to focus on natural dialogue models where the user can speak in "grunted phrases" and the computer can respond or interrupt with personalized natural language. [1]

Third, we feel the true grand challenge of speech input is to create truly expert systems. For example, when a new user wants to use some application, they generally choose not to read documentation, but instead ask someone who is already proficient with that application. We would like to capture this kind of expert tutoring in our gesture systems as bi-directional spoken communication where the system observes the users actions, and then converses with the user in order to guide them. Current efforts along this path are woefully inadequate at many levels ranging from the quality of speech recognition to the quality of speech generation, but are perhaps least adequate in terms of their ability to make reasonable interpretations of the user's natural language (gesture and speech) of input.

References

[1] M. Bloomenthal, R. C. Zeleznik, R. Fish, L. S. Holden, A. S. Forsberg, R. Riesenfeld, M. Cutts, S. Drake, H. Fuchs, and E. Cohen. Sketch-N-Make: Automated Machining of CAD Sketches, *Proceedings of the 1998 ASME 8th Computers In Engineering Conference*, 1998.

[2] A. S. Forsberg, M. K. Dieterich, and R. C. Zeleznik. The Music Notepad, *Proceedings of UIST '98*, ACM SIGGRAPH, 1998.

[3] A. S. Forsberg, J. J. LaViola, and R. C. Zeleznik. ErgoDesk: A Framework for Two and Three Dimensional Interaction at the ActiveDesk. *In the Proceedings of the Second International Immersive Projection Technology Workshop*, Ames, Iowa, May 11-12, 1998.

[4] A. S. Forsberg, J. J. LaViola, L. Markosian and R. C. Zeleznik. Seamless Interaction in Virtual Reality, *IEEE Computer Graphics and Applications*, 17(6):6-9, November/December 1997.

[5] J. J. LaViola. A Multimodal Interface Framework For Using Hand Gestures and Speech in Virtual Environment Applications. *To appear in the proceedings of The Gesture Workshop'99.* March 1999.

[6] J. J. LaViola, A. S. Forsberg, R. C. Zeleznik. Jot: A Framework for Interface Research. *IBM's interVisions Online Magazine*, Issue No. 11, http://www.alphaworks.ibm.com, February, 1998.

[1] Voice phone menus are a good example of a poorly humanized interface. Speech generation, although often based on sampled voices, is rarely convincing when phrases are constructed of pieces. In addition, the inability to interrupt a voice menu makes them almost intolerable.

[7] R. C. Zeleznik, A. S. Forsberg, and P. S. Strauss. "Two Pointer Input for 3D Interaction," *Proceedings of 1997 Symposium on Interactive 3D Graphics* (Providence, Rhode Island, April 27-30, 1997).

[8] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes. SKETCH: An interface for sketching 3D scenes. *Computer Graphics (Proceedings of SIGGRAPH '96)*, August, 1996.