

A Dancing Programmer in an Immersive Virtual Environment

Noritaka OSAWA^{*}, Kikuo ASAI^{*}, Yuji Y. SUGIMOTO^{*} and Fumihiko SAITO[†]

^{*}National Institute of Multimedia Education, JAPAN

[†]Solidray Co. Ltd, JAPAN

{osawa,asai,yuji}@nime.ac.jp saito@solidray.co.jp

Abstract

Our immersive programming system allows programs to be edited and controlled by direct manipulation and hand gestures in an immersive virtual environment utilizing multimodal interfaces. It lets beginners focus on learning the essence of programming without requiring familiarity with the keyboard and commands of a conventional editor.

1. Background

It usually takes beginners a long time to make a program. This is partly because they need to understand the new concepts, syntax, and semantics of a programming language. However they often spend a lot of time editing a program and correcting typos and misspellings, according to our observation, when a text programming language is used because many novice programmers are not used to touch typing (especially in non-western countries) or the operations of a text editor or conventional integrated development environment (IDE) system. We think that this situation adversely affects not only their programming efficiency but also their understanding of the programming concepts. Multimodal interactions in immersive virtual environments will ameliorate the situation since the interactions can be better modeled after reality than a keyboard interface or GUI.

Three-dimensional space should be used in order to fully utilize the body's freedom of movement. To view objects in 3D, a 2D display is insufficient because it cannot give one an adequate sense of depth in 2D. Therefore a stereoscopic view is preferable to help one to touch, grasp, and move virtual objects in 3D.

We have been developing an immersive programming system called *ougi*, which allows a programmer to edit and manipulate programs through direct manipulation and gestures using his/her hands in an immersive environment. The programmer appears to be dancing because he/she moves and gestures in the immersive virtual environment. The current implementation supports multimodal programming in a subset of the Java programming language although the system can be extended to support other languages.

There has been considerable research on visual 3D programming and algorithm animation [2][4][5][8][9], or visual object-oriented languages [3]. However, a general-purpose object-oriented programming system for immersive environments has not been implemented. Moreover, interactive multimodal interfaces for

programming in immersive environments have not been fully studied. Direct manipulation such as a grab-move-release (or get-and-put) operation is basic in 3D, but simply replacing the drag-and-drop operation in a GUI with a grab-move-release operation in 3D is insufficient. The design of 3D interactive interfaces is not a trivial problem.

2. Immersive Programming

Although our system is an immersive and multimodal programming system, it does not completely eliminate textual representations. They are retained as useful because we have all learned written languages for many years and understood the meanings of many words. We are not sufficiently trained in understanding the meanings of graphical symbols such as icons. It is difficult to understand abstract concepts only from graphical symbols because we do not have standard symbols for abstract concepts. Since we are familiar with mathematical formulae, the usual mathematical expressions are also used in the system.

It is reasonable to represent language elements by nested structures according to the programming language syntax. That representation is similar to nested boards or nested boxes such as [5]. However, the nested structures are not always shown in the *ougi* system since they may limit the visibility of the inside of the structure or become a visual clutter.

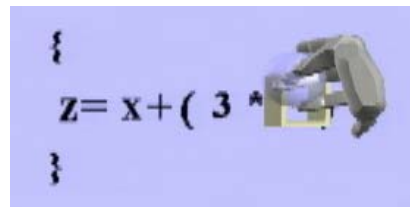


Figure 1: Block statement with an incomplete expression.

Figure 1 shows a block statement containing an incomplete expression without the nested structure being displayed. It looks like ordinary text, but has a hollow box as a placeholder that can include an expression or a variable. A virtual hand is putting a variable into the placeholder.

When the virtual hand intersects with the regions of the expression, the nested structure of the expression is as shown in Figure 2. This reveals the syntax structure of

that expression. Each region has a handle like a cylinder. Direct manipulation of the handle is used to move and copy the region. The handle is necessary for easy operations. If it were not used, the nested structures would have to be large enough to be separated from the outer and inner structures, otherwise it would be difficult to choose a middle structure without touching neighboring ones. The handle is also utilized to get and set properties. A hand gesture such as pinching and turning opens a panel for further interactions.

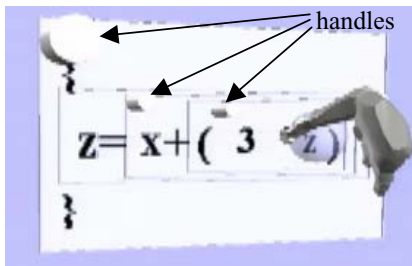


Figure 2: The block's nested syntax structure is revealed when a virtual hand intersects with it.

Although values, variables, or elements of a language are represented as visual objects in conventional visual programming, we do not think that information about classes or types, and signatures of methods are effectively visualized in visual programming. In the *ougi* system, they are represented using 3D glyphs that show superclass-subclass relationships by inclusion relationships of the 3D glyphs, which are based on 2D glyphs[7]. One can understand type conformance among classes, variables, and values without referring to additional documents or diagrams. We think the use of the glyphs will help programmers.

A class hierarchy is represented by a 3D graph, whose layout is arranged by a force-directed technique based on heat models with multiple foci[6]. This technique allows one to assign an importance or interest to a node as a virtual temperature in a graph and to navigate in a graph using virtual heat radiated from ones fingers.

Programming needs commands for program generation, compilation, execution etc. 3D widgets like buttons for the commands are arranged around your body egocentrically. When you move, the widgets move along with you.

3. Prototype System

A prototype system has been developed using the Java programming language, the Java 3D class library, and *it3d* library (Interactive Toolkit library for 3D applications) which has distributed I/O functions, useful 3D widgets, and a gesture recognition engine. The use of Java enhances the portability of the system, which should work on a wide range of computer systems. The prototype

system runs on a PC workstation (Compaq AP550 with dual 1-GHz processors and an Elsa Synergy III graphics board supporting dual displays). Six-DoF sensors (Polhemus Fastrak) and sensor gloves (Virtual Technologies CyberGlove) are used to detect the position and motion of the user's body and hands.

The prototype system works in a virtual reality environment called TEELeX (Tele-Existence Environment for Learning eXploration) [1] at the National Institute of Multimedia Education in Japan. TEELeX is a kind of surround display system. It has a large cubic screen for immersion. Each face is 3 meters by 3 meters. Circular polarization is employed to give a stereoscopic view to users, who only need to wear lightweight stereo glasses. The prototype system uses one stereoscopic face.

4. Summary

We have been developing a prototype immersive programming system utilizing multimodal interfaces. After we finish its development, we will conduct experiments to evaluate the effectiveness of learning programming using physical movement in an immersive environment. We also plan to develop an immersive programming language that is more suitable for multimodal interfaces in immersive environments.

Acknowledgements

This research was partially supported by Grant-in-Aid for Scientific Research (11358002) in Japan.

References

- [1] Kikuo Asai, Noritaka Osawa, and Yuji Y. Sugimoto, "Virtual Environment System on Distance Education," *Proc. of EUROMEDIA '99*, pp. 242-246, 1999.
- [2] M.H. Brown and M.A. Najork, "Algorithm animation using 3D interactive graphics," *ACM Symp. on User Interface Software and Technology*, pp. 93-100, 1993.
- [3] Margaret M. Burnett, Adele Goldberg and Ted G. Lewis, *Visual Object-Oriented Programming: Concepts and Environments*, Manning, 1995.
- [4] H. Lieberman, "A three-dimensional representation for program execution," *IEEE Workshop on Visual Languages*, pp. 111-116, 1989.
- [5] Marc A. Najork, "Programming in Three Dimensions," *Journal of Visual Languages and Computing*, Vol. 2, No. 7, pp. 217-242, 1996.
- [6] Noritaka Osawa, Kikuo Asai, Yuji Y. Sugimoto, "Immersive Graph Navigation Using Direct Manipulation and Gestures," *Symposium on Virtual Reality Software & Technology 2000 (VRST 2000)*, pp. 147-152, 2000.
- [7] Noritaka Osawa, "Generation and Evaluation of Glyph Representing Superclass-subclass relationships," *Proc. of IEEE Symp. on Visual Languages*, pp. 81-82, 2000.
- [8] F. Van Reeth and E. Flerackers, "Three-dimensional graphical programming in CAEL," *IEEE Symp. on Visual Languages*, pp. 389-391, 1993.
- [9] J. Stasko and J. Wehrli, "Three-dimensional computation visualization," *IEEE Symp. on Visual Languages*, pp. 100-107, 1993.