Multimodal Slideshow: Demonstration of the OpenInterface Interaction Development Environment

David Juras, Laurence Nigay, Michael Ortega, Marcos Serrano

Grenoble Informatics Laboratory (LIG), University of Grenoble
B.P. 53, 38041, Grenoble, France

{David.Juras, Laurence.Nigay, Michael.Ortega, Marcos.Serrano}@imag.fr

ABSTRACT

In this paper, we illustrate the OpenInterface Interaction Development Environment (OIDE) that addresses the design and development of multimodal interfaces. Multimodal interaction software development presents a particular challenge because of the ever increasing number of novel interaction devices and modalities that can used for a given interactive application. To demonstrate our graphical OIDE and its underlying approach, we present a multimodal slideshow implemented with our tool.

Categories and Subject Descriptors

H.5.2 **[Information Interfaces and Presentation]**: User Interfaces – Input devices and strategies, Interaction styles, Prototyping, User interface management systems (UIMS); D.2.2 **[Software Engineering]**: Design Tools and Techniques – User interfaces

General Terms

Algorithms, Design, Experimentation, Human Factors.

Keywords

Multimodal Interfaces, Development Environment, Prototyping.

1. INTRODUCTION

By extending the sensory-motor capabilities of computer systems to better match the natural communication means of human beings, multimodal interfaces enhance interaction between users and systems. We are currently facing a vast world of possibilities for interaction modalities going beyond the screen-keyboard-mouse. Everyday objects can take part in the interaction in ubiquitous computing including an augmented table and the user can freely switch from one modality to another one according to her/his context: running in the street, at home, in front of a big screen in an airport, etc. Facing the vast world of possibilities in terms of modalities, the open source OpenInterface (OI) framework enables us to quickly explore different interaction possibilities. The OI framework allows multimodal user interfaces to be created more quickly, enabling rapid prototyping and therefore more iterations as

part of an iterative design method for achieving usable multimodal user interfaces. The OI framework is made of an underlying platform, namely OpenInterface kernel, and the OpenInterface Interaction Development Environment (OIDE). The OI kernel [1] is a component-based platform that handles distributed heterogeneous components based on different technologies (Java, C++, Matlab, Python, .NET). The kernel therefore enables us to manage interaction modalities developed in different programming languages. The OIDE is a component-based system built on top of the OI run-time kernel. OIDE adds development tools offering access to interaction capabilities at multiple levels of abstraction. The underlying approach and the characterization space of the components handled by the OIDE is fully described in [4].

This demonstration highlights the key features of the OIDE by considering an example, a multimodal slideshow. Another demonstration based on Google Earth is described in [2]. In this demonstration, we consider key tasks of a slideshow, namely <change transition> <next/previous slide> <zoom x y>, that can be performed by using various atomic and/or combined interaction modalities.

2. OPENINTERFACE INTERACTION DEVELOPMENT ENVIRONMENT: OIDE

The OIDE is a graphical environment that allows direct manipulation and assembling of components in order to specify a "pipeline" defining a multimodal interaction. Figure 1 shows the graphical editor with a simple pipeline made of four components.
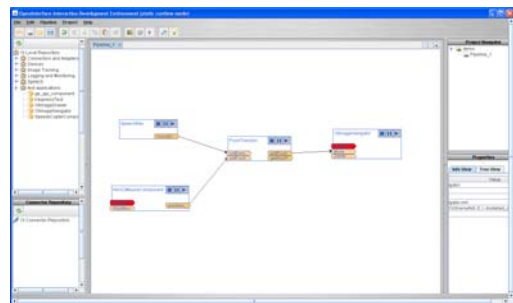

Figure 1. OIDE: a graphical editor for assembling components. (from [4])

The OIDE component repository includes modalities both developed "in-house" (e.g., speech recognition, video "finger tracker", accelerometer-based gesture recognition) and also accessible via proxies to other component sets (e.g., Phidgets, ARToolkit). Our components include device drivers, interaction techniques, multimodal fusion facilities, development services and

developer-defined combinations. The palette of available components is displayed on the left part of the pipeline editing zone in Figure 1. The demonstration will showcase a selection of existing components in the context of the multimodal slideshow application. For example Figure 2 presents components used in the demonstration: they correspond to two devices, the Shake device and the Wiimote device. We first consider the Shake device component. The Shake device [5] contains several sensors including a triple axis accelerometer, two capacitive sensors and a vibrotactile actuator. We partially describe the Shake device by considering two generic device components: one representing the capacitive sensor (1D-Discrete-Pressure) and another one for the triple axis accelerometer (3D-Continuous-Motion). For the Wiimote device we also partially define it by two components, the button which continuously sends 0 or 1 (1D-Continuous-Pressure) and the accelerometer. The Shake and the Wiimote share one generic device component: by adopting this approach with generic device components, they can easily be interchanged, for example for the task of navigation in the pile of slides. We illustrate this point by changing the Wiimote by the Shake for the case of gestures (Up/Down) for navigating in the pile of slides (Previous/Next).
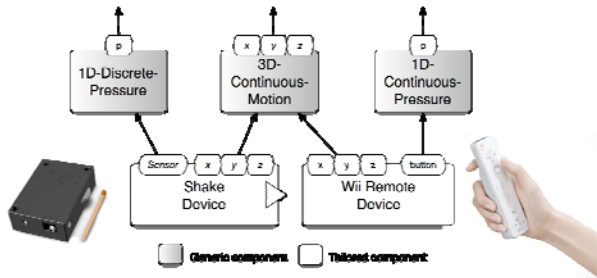

Figure 2. OIDE components: the Shake and Wiimote devices.

3. SLIDESHOW APPLICATION

Using the OIDE for assembling components and changing the specified pipeline, the demonstration will show the usage of various interaction modalities in the application context of the multimodal slideshow. The OIDE components establishing the links to the slideshow application correspond to the performed tasks that include <next/previous slide> and <zoom x y>.

For the defined tasks, we first consider atomic modalities including speech commands, the buttons of the wiimote as well as gestures (Up/Down) performed using the Wiimote and the Shake: to do so we assemble existing generic and tailored components and then modify the corresponding pipeline.

We then consider the combination of modalities. We define composition components based on the CARE properties (Complementarity, Assignation, Redundancy and Equivalence) [3]. Those composition components perform temporal fusion of input modalities. Future work will focus on other types of composition, such as spatial or semantic composition. For instance we demonstrate the redundant usage of two modalities: speech commands and gestures for changing a slide. In such a case the current displayed slide will be changed only if the user issues the speech command while performing a gesture. We also demonstrate the complementary usage of two modalities by considering the zoom command. The zoom on a specific point of the current slide can be specified by pressing a balloon along with a pointing gesture using

the Wiimote (Figure 3). A high pressure on the balloon is used to zoom in, and a low pressure to zoom out. Pressure is achieved using the interface-Z sensor (www.interface-z.com).

Finally the OIDE also includes a set of debugging and monitoring tools to help designers in building multimodal applications. We will demonstrate the usage of the generic "oscilloscope" display that is used to view data streams at runtime. Such tools are used for fine tuning of the modalities, for example for fixing the amplitude of the performed gestures using the Wiimote or the Shake as well as for tuning the intensity of the pressure of the balloon for zooming in.
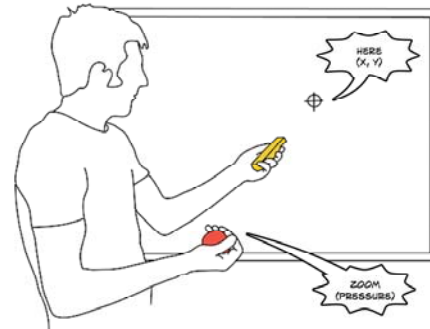

Figure 3. Example of bi-manual interaction as a case of two modalities used in a complementary way.

4. CONCLUSION

This demonstration illustrates our graphical tool for developing multimodal interaction, the OpenInterface Interaction Development Environment (OIDE). Its key features and in particular its support for quickly modifying the multimodal interaction as part of a user-centered iterative design, is illustrated via the exploration of various atomic or combined interaction modalities for a multimodal slideshow.

5. ACKNOWLEDGMENTS

6. REFERENCES

[1] Benoit, A., et al.. 2006. Multimodal signal processing and interaction for a driving simulation: component-based architecture. JMUI, Springer,1, 1, 49-58.

[2] Gray, P., Ramsay, A., Serrano, M. 2007. A demonstration of the OpenInterface Interaction Development Environment. UIST'07 Adjunct Proc., ACM Press, 39-40.

[3] Nigay, L., Coutaz, J. 1997. Multifeature Systems: The CARE properties and their impact on software design. Intelligence and Multimodality in Multimedia Interfaces, AAAI Press.

[4] Serrano, M., Juras, D., Nigay, L. 2008. A three-dimensional characterization space of software components for rapidly developing multimodal interfaces. In proc. ICMI'08, ACM Press, to appear.

[5] Williamson, J., Murray-Smith, R., and Hugues, S. 2007. Shoogle: multimodal excitatory interaction on mobile devices. In Proc. CHI'07, ACM Press, 121-124.