A Demonstration of the OpenInterface Interaction Development Environment

Philip Gray and Andrew Ramsay
Computing Science Department
University of Glasgow, Glasgow UK
{pdg, adr}@dcs.gla.ac.uk

Marcos Serrano
LIG Laboratory, HCI group
University of Grenoble 1, Grenoble FRANCE
marcos.serrano@imag.fr

ABSTRACT
Multimodal interaction is hard to develop, especially in mobile/ubiquitous settings with novel interaction devices. This is true, at least in part, because prototypes in this domain are difficult to build, to change and to monitor in order to analyse, interpret and evaluate interaction data. In this demonstration we present the OpenInterface Interaction Development Environment (OIDE) that addresses these challenges via its component repository and its construction, debugging and logging tools.

ACM Classification: D.2.2 [Software Engineering]: Design Tools and Techniques. – User interfaces.

General terms: Design**,** Experimentation, Human Factors

Keywords: multimodal interaction techniques, development environments, prototyping, usage monitoring

INTRODUCTION
Multimodal interaction is hard to develop, especially in mobile/ubiquitous settings with novel interaction devices, in part because we don‘t have well-validated theory or practice on which to build and thus have to rely on exploration and evaluation of techniques as we develop them. Such empirically driven development is hard because prototypes in this domain remain difficult to build, to change and to monitor.

Tools are beginning to emerge that help with constructing interaction techniques based on sensor input and analysing and interpreting the low-level device data [3], but there is still no environment that allows a designer to work at multiple levels of abstraction, from low-level physical device properties through interaction abstractions (e.g., semantic fusion) to application and task level operations and objects.

The OpenInterface Project aims to address this problem via an integrated development environment for rapid prototyping and empirical evaluation of multimodal interaction techniques. This demonstration highlights key features of the environment as described below.

Copyright is held by the author/owner (s).
UIST'07, October 7-10, 2007, Newport, Rhode Island, USA.
ACM 978-1-59593-679-2/07/0010

THE OI DEVELOPMENT ENVIRONMENT FEATURES
The OpenInterface Interaction Development Environment (OIDE) is a component-based system built on top of a run-time kernel, called the OpenInterface Platform [1]. OIDE adds development tools offering access to interaction capabilities at multiple levels of abstraction. The demonstration presents the development environment in one of its application contexts: navigation of a large information space on a large display surface.

Run-Time Kernel
The OpenInterface Platform is responsible for creating and managing connections between components, processing the output of multiple components (e.g., for fusion and multicasting), loading and interpreting the OI application descriptions, or “pipelines”. Pipelines can be created manually or semi-automatically using the editor described below.

Component Repository
The demonstration will showcase a selection of components from the OIDE component repository, 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. OIDE supports descriptions, querying and access in an extensible set of description schemas.

Construction Tools
Fig. 1 illustrates the OIDE Graphical Editor in which components accessible via the repository can be inspected, configured, linked to other components and to external services or to application functionality. The result can be either an individual technique or a fully functional user interface to a real application.

Debugging and Logging Tools
The *event logger* is capable of recording multiple component-specific formatted data streams while an OI application is executing. The logger component can generate timestamps for each set of data it receives or it can use timestamps generated by the source component.

As the logging format is controlled by the source components, it is then simple to load the recorded data into external visualization toolkits, such as Matlab or Replayer [4].

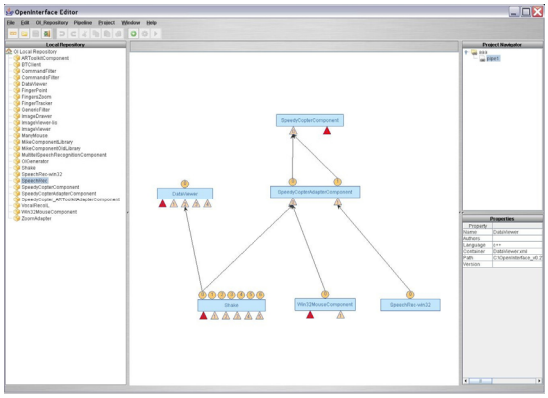

Figure 1: The OIDE Editor.

A generic *"oscilloscope"* display can be used to view data streams at runtime. The component can be extensively configured while in use, including selection of data streams to display, changing the scaling of the current data stream, and adjusting the sample rate for incoming data.

EXAMPLE APPLICATION

As an illustration, consider navigation in Google Earth using a novel input device. Three different techniques to perform this task have been developed, 2 using custom built hardware [2], and the 3rd using the SHAKE sensor device [5] (Fig. 2). The SHAKE contains a variety of sensors, including a triple axis accelerometer, two capacitive sensors and a vibrotactile actuator.

OIDE hosts a SHAKE interface component, which is used to process raw output from the SHAKE and emit events when selected conditions are met, for example, when the tilt of the device exceeds a certain threshold. Our OI pipeline links the SHAKE interface component to a generic keypress event generator. When events are produced by manipulations of the SHAKE, a virtual keypress event is sent to Google Earth, running outside the OI environment.

Navigation is performed by tilting the SHAKE in the 4 main directions (left/right/forward/back) to move west/east/north/south, and by pressing the two capacitive sensors (zoom in and out).

Figure 2 shows a subset of the data visualization options available in the OI framework. The first two oscilloscopes show data at the lowest level of abstraction, direct from the SHAKE itself, while the 3rd is showing application-level data from Google Earth.

It is also possible to display further levels of abstraction. For example, an intermediate stage between the raw and application-level visualizations would be a display of the virtual keypresses events being generated. A 4th instance of the oscilloscope component could be used for this purpose.

In addition to sending events from OI to Google Earth, our example also features feedback from Google Earth into the OI environment, through a component which is able to use the Google Earth API. This component obtains the current terrain height from Google Earth, and triggers a vibration event on the SHAKE device. The intensity of the vibration is proportional to the height of the terrain.

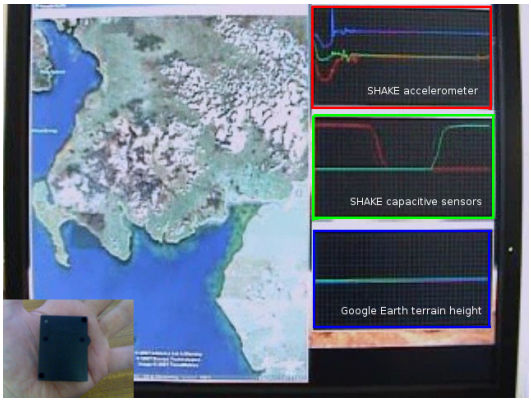

Figure 2: Google Earth navigator with real-time "usage oscilloscope" components. The SHAKE is illustrated in the bottom left-hand corner.

CONCLUSIONS

This demonstration presents the OpenInterface Interaction Development Environment. Its key features (component repository, construction, debugging and logging tools) are illustrated via explorations of interaction with a large information space. The ability of the environment to provide conditional logging of the most relevant and/or interesting data provides an added benefit to the interaction designer.

For more information, please visit www.oi-project.org.

ACKNOWLEDGMENTS

The OpenInterface Project is an IST Framework 6 STREP funded by the European Commission. The authors acknowledge the contribution to the demonstration of their OI colleagues and of non-OI member Emmanuel Dubois.

REFERENCES

1. Benoit, A., Bonnaud, L., Caplier, A., Damousis, I., Tzovaras, D., Jourde, E., Nigay, L., Serrano, M. and Lawson, J-Y. 2006. Multimodal Signal Processing and Interaction for a Driving Simulation: Component-based Architecture. Journal on Multimodal User Interfaces,1, 1,49-58.

2. Dubois, E., Truillet, P., and Bach, C. 2007. Evaluating Advanced Interaction Techniques for Navigating Google Earth. In Proc HCI 2007, Vol 2. to appear.

3. Hartmann, B., Abdulla, L., Mittal, M., and Klemmer, S. R. 2007. Authoring sensor-based interactions by demonstration with direct manipulation and pattern recognition. In Proc CHI '07. ACM Press, New York, NY, 145-154.

4. Morrison, A., Tennent, P., and Chalmers, M. 2006. Coordinated Visualisation of Video and System Log Data. In Proc CMV2006. IEEE Computer Society, 91-102.

5. Williamson, J., Murray-Smith, R., and Hughes, S. 2007. Shoogle: Excitatory Multimodal Interaction on Mobile Devices. In Proc CHI '07. ACM Press, 121-124.