DiABlu – Digital Arts' Bluetooth

Jorge Cardoso and Nuno Rodrigues,

Research Center for Science and Technology in Art (CITAR)
Universidade Católica Portuguesa – Porto – Campus da Foz
Rua Diogo Botelho 1327, 4169-005 - Porto, Portugal
jccardoso@porto.ucp.pt
nrodrigues@porto.ucp.pt

Abstract. Digital art installations can often gain from the capability of detecting the presence of people observing them. With this information, the artists can enhance the experience of whom interacts with their work. While this detection can be made by means of web cameras or sensors, these systems are generally dificult to implement for people with a low knowledge of programming. We propose a system that uses bluetooth to do this detection and allows easy integration with applications often used by digital artists. The system also allows users to interact with the installation using their mobile devices. It's intended to be used in art installations by digital artists who wish to give their audience a new way to interact with their pieces.

1 Introduction

Digital art installations can often gain from the capability of detecting the presence of people observing the installation. With this information, the artists can enhance the experience of whom interacts with their work. Sometimes, detecting the presence of people is even the only way to implement the conceptual meaning of the work of art.

There are many ways to detect the presence of people near an installation. We can use web cameras with more or less advanced detection techniques, or a wide range of general purpose sensors combined with sensor control interfaces like the iCubeX system [3]. Implementing these solutions, however, is a distraction to the artist from more important aspects of the installation. Often, these systems mean building special structures to position web cameras and sensors and have to be fine tuned to every location.

We propose a system for detecting the presence of people by detecting the presence of bluetooth enabled devices. Our system allows easy integration with applications used for building digital art installations, namely by our students here at the School of Arts. The system is called "Digital Arts' Bluetooth – DiABlu"[1].

[1] More information about this project can be obtained on the project website at http://citar.ucp.pt/diablu or http://soundserver.porto.ucp.pt/diablu

Our goal is to develop a system that is easy to use and integrate with other applications, like Flash [4], Processing [2], Max/MSP [1], Pure Data [5], etc, by using the widely used Open Sound Control [6] (OSC) protocol.

Besides allowing the detection of bluetooth devices, the DiABlu System will also allow users to interact, through their mobile devices, with the installation.

Throughout this paper we use the names of the main components of the DiABlu system, with the following meaning:

Target Application The application that is developed by the final user and that needs information about bluetooth devices. This application can be developed in Max/MSP, Pure Data, Processing, Flash, or any other environment that supports the Open Sound Control (OSC) protocol.

DiABlu Server The base DiABlu application that connects to the Target Application and provides information about the nearby bluetooth devices.

DiABlu Client Mobile application that connects to the DiABlu Server and allows the user to input keystrokes and text messages that will be delivered to the Target Application.

The remainder of this paper is organized as follows: in section 2 we explain the components of the DiABlu system and how they communicate between themselves; section 3 describes some typical use cases of the system; finally, section 4, concludes.

2 Design

The general architecture of the DiABlu System is given in Fig. 1.

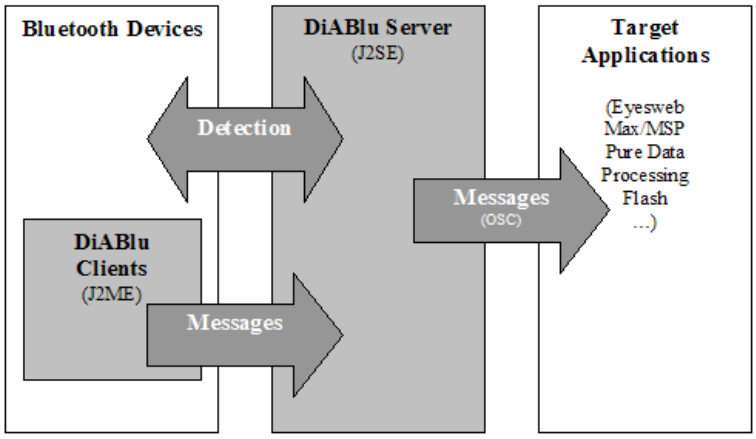

Fig. 1. DiABlu's system architecture

2.1 DiABlu Server

The DiABlu Server is the core of the DiABlu system. This application is responsible for detecting nearby bluetooth devices and informing the Target Application of the number of present devices and their IDs and names.

Basically, the DiABlu Server performs the following actions:

1. Scan the environment for the presence of bluetooth devices.
2. Inform the Target Application of the nearby devices.
3. Accept bluetooth connections from devices and receives data (keystrokes and text messages)
4. Inform the Target Application of the data received.

All communication between the DiABlu Server and the Target Application is made using the Open Sound Control[6] (OSC) protocol.

2.2 DiABlu Client

The DiABlu Client is a mobile application developed in Java ME for devices that support the MID profile plus the bluetooth Java API. This application allows the handheld user to interact with the Target Application via the DiABlu Server. At this time, the interaction is limited to keypresses, i.e., the DiABlu Client transmits the codes of the keys the user has pressed to the DiABlu Server, which, in turn, will transmit them to the Target Application.

The DiABlu Client is a general application, in the sense that it is independent of the Target Application. Basically, it allows the user to:

1. Discover nearby DiABlu Servers and connect to one. This makes it possible for the user to choose to interact with one from a number of nearby installations.
2. Send a text messages to the Target Application.
3. Send keystrokes to the Target Application.

2.3 Target Application

The Target Application is any application, developed by the final user of the DiABlu System, that is capable of receiving data via the OSC protocol. The Target Application receives updated information about the names, IDs and number of bluetooth devices near the computer running the DiABlu Server. It also receives the key codes that a given DiABlu Client's user pressed while connected to the DiABlu System.

3 Usage Scenarios

There are three typical high-level use cases for the DiABlu System:

No Interaction In this use case, the Target Application only needs to know how many devices there are in the vicinity and/or their names. The installation does not have any direct interaction capability, it just reacts to the presence of bluetooth devices.

Shared Interaction This use case represents all applications that besides reacting to the presence of bluetooth devices, allow their users to directly interact with the application. Interaction is done by means of the DiABlu Client application, that must be installed in the device and is limited to sending keystrokes and text messages. There are no restrictions, imposed by the Target Application, on the number of users that may be interacting simultaneously with it.

Exclusive Interaction This is similar to the Shared Interaction use case, except that the Target Application limits the number of users directly interacting to one. This is a typical use case for navigational interfaces in which at most one user may be navigating at a time.

4 Conclusion

This paper presented the DiABlu System, a bluetooth detection and interaction system for the digital arts community. We have described the general functionality and architecture of the system and typical use cases for this kind of application.

We plan to use the DiABlu System on projects developed at the School of Arts to gain experience and insight on the kind of functionality needed by our users in order to further develop and enhance the system.

References

1. Cycling74. Max/MSP. `http://www.cycling74.com` (accessed 16/03/2006).
2. Ben Fry and Casey Reas. Processing. `http://processing.org` (accessed 16/03/2006).
3. Infusion Systems. I-CubeX. `http://infusionsystems.com/catalog/index.php` (accessed 16/03/2006).
4. Macromedia. Flash. `http://www.macromedia.com` (accessed 16/03/2006).
5. M. Puckette. Pure Data: another integrated computer music environment. In *Proc. the Second Intercollege Computer Music Concerts*, pages 37–41, 1996.
6. Matthew Wright and Adriane Freed. OpenSound Control: A New Protocol for Communicating with Sound Synthesizers. In *Proceedings of the 1997 International Computer Music Conference.*