A novel window manager for large screen display

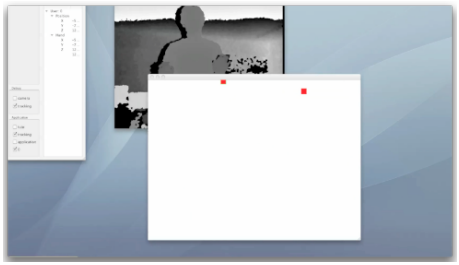
Benoit Lange, William Dyce, Nancy Rodriguez

A novel window manager for large screen display.

Benoit Lange
UPMC Univ Paris 06 and
CNRS UMR 7606, LIP6,
ICS - Institut du Calcul et de la Simulation
4 place Jussieu
F-75005, Paris, France
benoit.lange@lip6.fr

William Dyce
University of Montpellier II
Place Eugne Bataillon
34095 Montpellier Cedex 5,
FRANCE
William.Dyce@etud.univ-montp2.fr

Nancy Rodriguez
LIRMM, UMR 5506 CNRS,
University of Montpellier II
161, rue Ada
34392 MONTPELLIER
CEDEX 05, FRANCE
nancy.rodriguez@lirmm.fr



CHI12, May 510, 2012, Austin, Texas, USA.
ACM 978-1-4503-1016-1/12/05.

Abstract

The topic of this paper is Window Manager (WM) of Operating System (OS). These tools have not evolve since along time. Most of them are already based on WIMP paradigm and only few methods have been proposed to deal with Post-WIMP. Mobile devices or game devices have proposed some solutions to interact with gestures. For example on game console, Wii is developed around the Wiimote: this device is designed around two kinds of sensors: a video camera and accelerometers. More recently, Microsoft has developed an unique video camera to interact with their console: the Kinect.

In this paper, we are presenting our approach to redesign a window manager, based on Kinect from Microsoft. We use this camera to analyze what happen behind the screen, the goal of this approach is to develop a WM for large screen display. Our WM is designed to identify person, then analyze some gestures to interact with window. Some development of this approach have been realize to mange several user on the same computer. Two methods of window tiling are proposed both are based on location of user in space. Finally, our solution supports all off-the-shell applications (based on standard mouse interaction) but also Post-WIMP applications using TUIO.

Author Keywords
Interaction using Specific Capabilities or Modalities, Interaction Techniques and Devices, Usability, Accessibility and User Experience

ACM Classification Keywords
H.5.2 [User Interfaces]: Windowing systems; H.5.2 [User Interfaces]: Input devices and strategies; H.5.2 [User Interfaces]: Prototyping.

Introduction
Computer science is a quickly evolving domain; hardware capabilities have an exponential growth. IT systems are surrounding human environment, but the manipulation of these systems is not evolving as fast as the systems themselves.

Van-Dam has proposed an evolution of the WIMP paradigm in [11], called Post-WIMP. Direct manipulation from user the mainstream of this paradigm. This paradigm emerged in 1997, but it was not adopted at this time, the WIMP paradigm still dominates personal computer interfaces. New devices interfaces have been developed around the Post-WIMP: the Wiimote, Kinect, PS move, etc. The Wiimote is the most popular device using this paradigm, it is used with Wii nintendo console. This device is composed of acceleration sensors and an IR video camera, the combination of which give to the Wiimote, ability to recognize user motion. Other commercial game products have been developed around Post-WIMP paradigm such as Microsoft's Kinect and Surface, and Sony's PS Eye. Most of these devices were created to propose new interactive experiences.

In this paper we present an interactive Windows Manager (WM) designed to see behind the screen. Our goal is to manage users without any parameters, we give to the WM the capacity to analyze data from a camera. This WM is designed to manage large screen display. It has to support several user at the same time, off-the-shelf pieces of software or homemade applications with single- or mult-touch input.

This paper is decomposed as follow, Section 2 presents some related work on the Post-WIMP solution and management of windows. Section 3 presents our approach and integrated features. Section 4 gives methods used to implement our WM and some results with their drawbacks. And Section 5 concludes this paper and gives some perspectives.

Related Works
This section presents work on automatic interface-layouting adjustment, a new kind of interaction and user detection.

A large set of research has been focused on the automatic adaptation of interfaces. This concept has been widely present by Coutaz in [1, 10]. The goal of this technique is to adapt the interface depending from the window context. The layout and arrangement of windows is defined on the windows' size. This approach is often used for critical interfaces, for instance to redirect all the output of a plane when screen failure occurs. A special screen-layout is also used for large screen displays. Large screens are used in control or visualization rooms. In these special rooms, it is necessary to have layouts to maximize user-understanding, some solution for collaboration of user are proposed in [5, 4, 3]. Most of these approaches are only based on visual data and not on intaction with other users.

Another of our areas of interest has been the Post-WIMP

interface. A large set of hardware has been designed around this paradigm but most of the well-known solution have been industrial, not academic. Multi-touch table have been an important way to promote the Post-WIMP paradigm. A well known approach is called the DiamondTouch table [2]. This table is a multi-touch table based on a receiver and transmitter. When a user touchs the table, a signal is produced and interpreted by the whole system. Other solutions have been proposed by researchers, for example Han in [6] presented a low cost multi-touch table, able to collect touch-gestures from the user. This solution is based on IR light tracking: a piece of software analyzes the IR reflection produced by the hand and then processes the IR reflection in order to interact with the application. This technique has been also proposed by Microsoft with their multi-touch table Surface: the solution used here is based on IR projector.

Mistry developed a wearable augmented reality device in [8]. The goal of this device is an augmented reality device for every days. Our final point of interest is user detection. In [9], authors have developed some solutions to recognize the shoes of users and to adapt the interface accordingly. In this paper, we are presenting a smart Window Manager (WM), designed to interpret the content of what is behind the screen.

Our Approach

Collaboration between different users has always been an important field of research. People working in groups generally use large screens in order to work on data. However, operating systems do not deal efficiently with automatic layout of windows from different applications. Furthermore, collaborative manipulation of interface objects is not well supported. On a standard architecture, only one pointer is used to manipulate windows. Nowadays, most use OSs which supports multiple cursors, but software does not take advantage of this facility.

The solution presented in this paper is designed to deal with the VA (Visual Analytics) paradigm. This system manages some data-visualization applications and deals with their location on a large screen. The users' communications with these applications is two-way: they are no merely reading and understanding the data which is presented to them but interacting with it as well. We propose a smart Window Manager compatible with any software and based on knowledge. To create this knowledge, the WM is equipped with a video camera with a depth sensor. Information from user are extracted to produce this knowledge. These pieces of information include location and state of users. Features from interesting points are extracted, the hands are tracked produce certain interactions. These interactions can be provided by a simple touch (using only one hand) for regular software or based on multi-touch (multi-hand tracking) for special software.

To extract information, theWM uses a depth and a regular image from the camera. The depth image is used to extract morphological information and the location of user; the RGB camera is used to provide colorimetric information about users (for identification) and also to capture hand-gestures. The manipulation pipeline is defined below:

- User calibration,
- Skeleton tracking,
- Gesture recognition,
- Gestures are dispatched to software.

For the two first steps, we use OpenNi SDK to track, extract and follow users. Then, we analyze the hand behavior using the RGB camera. Our solution is based on two gestures: a push and a double push. These gestures aim to be similar to mouse event, thus the simple push is mapped to a simple click event, while the double push is mapped to a double click. Right- and left-clicks are assigned to the user's hands depending on which one is dominant.

Our WM is suitable for collaborative work, different users can use the system at the same time; the number of users is limited only by the size of the room. Our WM has two different layout methods, the first method is based on a tree map and the second method is based on the movement of users. The first method is a standard screen partitioning method. The system partitions space depending on these pieces of information. The location of each window can be computed easily, depending from the number of windows and the location of the center of gravity of each user.

The second placement method is based on user tracking. Each window is defined by its own information (width, height and user). We extract the center of gravity of each user and then we move the center of the window based on the movement of this point. Users can interact with each other by bringing their windows with them.

An interesting point is window manipulation. Our WM dispatches hands information to the destination window. Hand information is used as mouse cursors, special gestures are recognized to trigger actions. To recover events from each hand, our system uses TUIO (a multi-touch protocol for tangible user interfaces,[7]). Applications can also only use a hand location as a normal OS cursor.

Results

In this section, some results are presented. We have done some experiments to test our solution by overriding the OSX window manager.

Figure 1 presents the screen view for the user, the window at the top is optional and represents data from the depth sensor. The second window is a simulated application based on the TUIO protocol. Both hands are mapped to the screen (hand locations are represented by RED squares). If the user is moving in front of the camera, the window will also move.

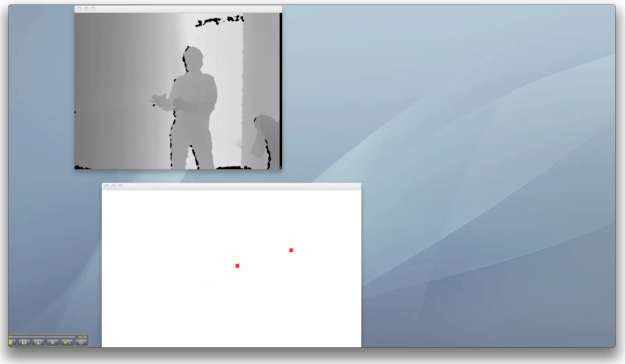


Figure 1: This picture shows one user interacting with the application. Depending on his hands' locations virtual pointers are moved in the virtual application

In Figure 2, we capture interactions of a user. Our solution is suitable suitable for many different user-positions: standing or sitting for example. Moreover, when users move into the room their window follows them. The overlapping of windows is also handled.

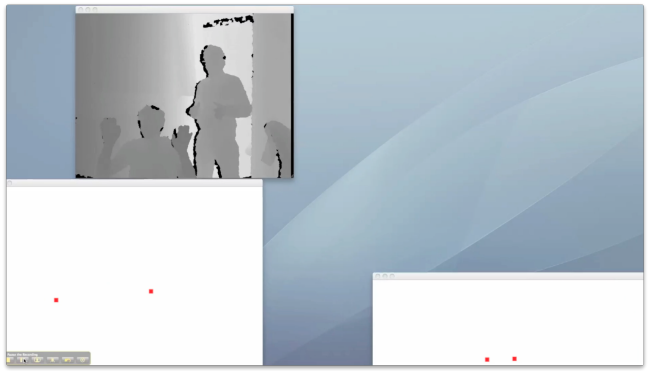


Figure 2: In this picture, users are tracked and each movement of both hands as well as that of the user's center of gravity is mapped to a dedicated window.

In Figure 3, we show the interaction of a user with a regular application, in this case TextEdit. The user is able to move the mouse with his dominant hand.

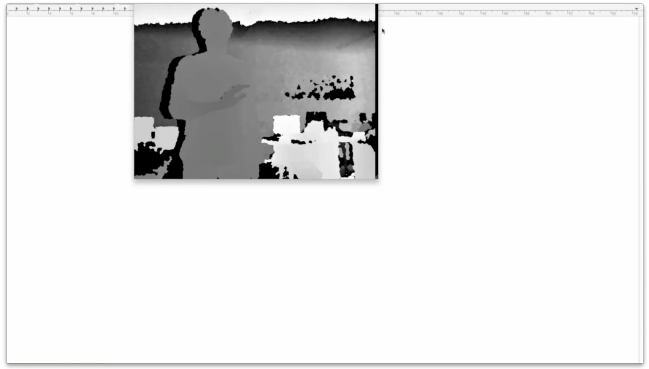


Figure 3: interaction with regular application, user movements are mapped to a mouse pointer, the location of the window can also be manipulated by the user's location.

Conclusions

In this paper, we have presented a new window manager designed to complement the regular OS window manager. Current solutions do not deal with multiple users and post-WIMP interaction. Most of them are used on large screen displays without dealing with user location, thus windows of these systems suffer of bad placement.

Our solution is based on user detection and tracking. We extract the number of users and their locations. With these pieces of information, we are able to organize several windows of a set of applications. The goal is to improve their placement. Moreover, when a user is moving, his dedicated window follows him.

To interact with off-the-self WIMP applications we emulate a mouse, but we also add a TUIO server to our application to manipulate TUIO applications. With our approach we are able to improve the collaboration between scientists.

Several feature need to be improved. Firstly, we do not deal with facial recognition , this method can be an efficient way of loading and saving the state of windows and also an important feature to improve our solution. Indeed, we would know more about the user, were we to have facial recognition implemented.

A second improvement would be to deploy this solution on another OS. We have used the OSX SDK to be able to interact with the window manager, an analogous step would need to be be performed for other operating systems.

Finally, a better solution is to implement our WM directly into an operating system.

References

[1] Coutaz, J. User interface plasticity: model driven engineering to the limit! In *Proceedings of the 2nd ACM SIGCHI Symposium on Engineering interactive Computing Systems*, ACM (2010), 1–8.

[2] Dietz, P., and Leigh, D. Diamondtouch: a multi-user touch technology. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, ACM (2001), 219–226.

[3] Duval, T., Fleury, C., Nouailhas, B., and Aguerreche, L. Collaborative exploration of 3d scientific data. In *Proceedings of the 2008 ACM symposium on Virtual reality software and technology*, ACM New York, NY, USA (2008), 303–304.

[4] Farr, W., Hut, P., Ames, J., and Johnson, A. An experiment in using virtual worlds for scientific visualization of self-gravitating systems. *Journal of Virtual Worlds Research 2*, 3 (2009).

[5] Germans, D., Spoelder, H. J. W., Renambot, L., and Bal, H. E. Virpi: A high-level toolkit for interactive scientific visualization in virtual reality.

[6] Han, J. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, ACM (2005), 115–118.

[7] Kaltenbrunner, M., Bovermann, T., Bencina, R., and Costanza, E. Tuio - a protocol for table based tangible user interfaces. In *Proceedings of the 6th International Workshop on Gesture in Human-Computer Interaction and Simulation (GW 2005)* (Vannes, France, 2005).

[8] Mistry, P., and Maes, P. Sixthsense: a wearable gestural interface. In *ACM SIGGRAPH ASIA 2009 Sketches*, ACM (2009), 1.

[9] Richter, S., Holz, C., and Baudisch, P. Bootstrapper: Recognizing tabletop users by their shoes. In *CHI 2012*, no. 4, ACM (2012).

[10] Thevenin, D., and Coutaz, J. Plasticity of user interfaces: Framework and research agenda. In *Proceedings of INTERACT*, vol. 99 (1999), 110–117.

[11] Van Dam, A. Post-wimp user interfaces. *Communications of the ACM 40*, 2 (1997), 63–67.