Cross-display Object Movement in Multi-display Environments

Miguel A. Nacenta
Department of Computer Science, University of Saskatchewan
110 Science Place
S7H 2Z2, Saskatoon, Canada
nacenta@cs.usask.ca

ABSTRACT

The action of moving GUI elements (e.g. cursors, windows, and icons) from one display to another is fundamental for environments that have more than one display. Although many techniques exist that allow cross-display object movement, it is difficult to know which technique is appropriate for which system, and how different techniques compare to each other.

To address this lack of knowledge I propose an analysis of cross-display object movement in three levels: how the destination display is referred to, how the gesture corresponds to the physical arrangement of the displays in the environment, and how the user gets feedback about her actions. The analysis enables a categorization of existing techniques that will support the choice of the right interaction technique for the right environment.

ACM Classification: H.5.2 *Information Interfaces and Presentation*: User Interfaces - Interaction Styles; Input devices and strategies;

General terms: Design, Human Factors.

Keywords: Multi-display Environments, Multi-monitor, Cross-display Object Movement, Interaction techniques.

INTRODUCTION

One of the fundamental advantages of using multi-display environments (MDEs) is the ability to place different interface objects on different displays. For example, in regular dual-monitor workstations, users tend to place applications on different displays according to their attentional requirements [5], and users of collaborative MDEs often need to move elements from their display to another person's [10].

Many cross-display object movement techniques have been proposed that enable these transitions (e.g. Pick-and-drop [15], Multi-browsing [6], Mouse Ether [1]); however, it is hard for designers to select the appropriate technique for an MDE because of two reasons: first, there is a lack of understanding of the process of cross-display object movement; and second, there is a lack of understanding of the design space of these techniques. This might result in a poor choice of interaction techniques because the technique will not match the requirements of the task and because there is no indication as to which techniques will perform better and why. For example, if we need to design the interaction of a working environment with personal and public displays (Figure 1, Left) we could choose a technique similar to Multibrowsing [6] that requires that users select the destination display by name from a list, which takes added time and effort, especially if there are many displays in the system. Most of the time, users will just want to move elements to a particular display which is easy to identify (for example, by a pointing gesture) instead of choosing an arbitrarily selected name from a list of labels such as the one represented in Figure 1 (Right).

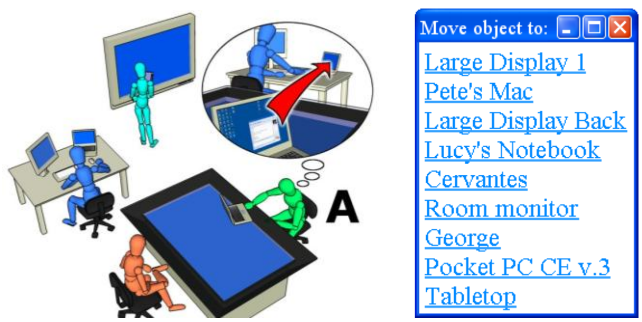

Figure 1. Left) Complex MDE. Right) A menu to select the destination of an object (similar to [6]).

In this work I address the issues stated above through an analysis of the cross-display movement process and a classification of techniques based on three main dimensions: the referential domain, the display configuration, and the control paradigm. By knowing how the destination display is referred to, how the gesture corresponds to the physical arrangement of the displays in the environment, and how the user gets feedback about her actions, we can classify techniques into spatial/non-spatial, planar/perspective/literal and closed-loop/open-loop/intermittent (see Figure 2).

This framework helps designers understand the differences and similarities between techniques, and therefore can predict more efficiently which technique is more appropriate for a particular environment. The classification also provides a common language for researchers to refer to important concepts of cross-display interaction.

UIST'08, October 19-22, 2008, Monterey, California, USA ACM 978-1-59593-975-3/08/10.

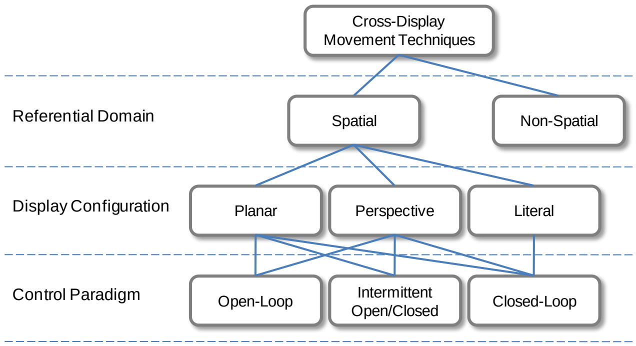

Figure 2. Diagram of the classification scheme

THE CROSS-DISPLAY OBJECT MOVEMENT PROCESS

Figure 3 represents the model of cross-display object movement upon which this analysis is based. The model distinguishes four sub-processes: first, the demands of the task are transformed into the intention of moving an object, which includes determination of the correct destination display; second, an adequate response (i.e., an action plan for moving the object to the destination) has to be formulated; third, the movement must be executed; and fourth, in some cases the user must monitor and adjust movement action through a feedback loop.

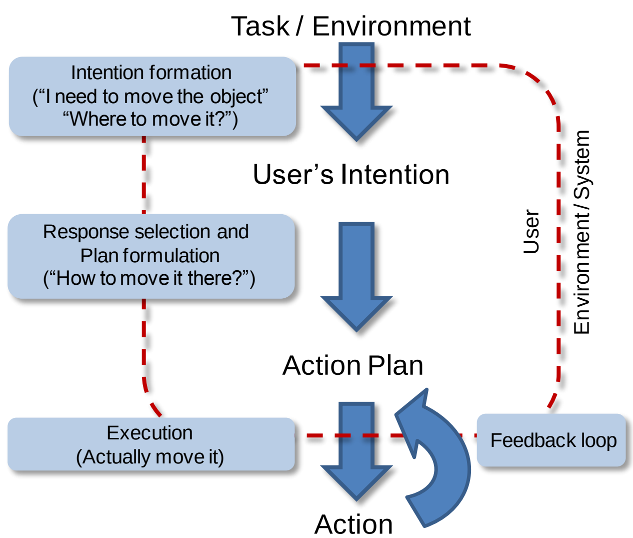

Figure 3. A model of cross-display object movement processes

The structure of the model can be illustrated with a simple example: a user is editing a document in a three-monitor desktop system (task/environment); at some point during the process, she realizes that she has to copy some text that is located on a different monitor, which requires moving the cursor to that display (user intention: to move the cursor to monitor B); the user then selects which of the possible actions has to be performed to achieve the goal, in this case, to move the mouse in a particular direction (action plan: move the mouse to the right); finally, the user performs the movement (action), which she adjusts along the way by looking at the position of the cursor (feedback loop).

This simple model represents the cross-display object movement process with elements that are already well known; nevertheless, the division into sub-processes allows us to separately understand different characteristics of different techniques and to classify them.

A FRAMEWORK IN THREE LAYERS

The framework I propose is divided into three levels: the referential domain, the display configuration and the control paradigm. Each of the following sub-sections describes one of the dimensions with its associated categories, presents example techniques from each category, and summarizes evaluations that compare techniques across categories.

Referential domain

Cross-display object movement can be seen as the transformation of an intention (the user wants an object to be in a particular display) into a system´s change of state (the object appears in its new position). In order to achieve this change of state the user needs to communicate to the system where to move the object through an interaction technique.

The intention of the user and the display specification required by the interaction technique can be expressed in a variety of ways. For example, the user might want to move the object to a display that is called 'John's display' or to the display that is to her right – two different ways of referring to the same display. Similarly, an interaction technique might require typing the name of the destination display, or may allow the user to indicate the destination by touching the display. The type of representation of the displays is what I call the referential domain.

I divide techniques into spatial and non-spatial depending on the way that the user has to specify the destination. Spatial techniques require gestures that relate spatially to the destination; for example, with pointing [4, 11] we indicate the destination by orienting a device or the finger in the direction of the destination display.

Non-spatial techniques indicate destination through actions that are independent of the physical location of the destination display. For example, we can use Instant Messaging to transfer a file to another person's laptop in a room, but the selection of the laptop's owner from the buddy list does not relate to the position of the people in the room.

These two different categories of techniques will perform differently depending on the task. If the user's intention is formulated in terms of a spatial location (Figure 4.A), spatial techniques will be a better match (e.g. the cursor movement technique of Figure 4.C), if the intention is not dependent on the location of the display (Figure 4.C), non-spatial techniques seem more appropriate (Figure 4.D).

Not surprisingly, existing evidence indicates that using spatial techniques for spatially formulated tasks is more efficient [16]. Our own research corroborates this, but also suggests that the physical location of the displays can interfere with the process, even for non-spatial tasks with non-spatial techniques [7].

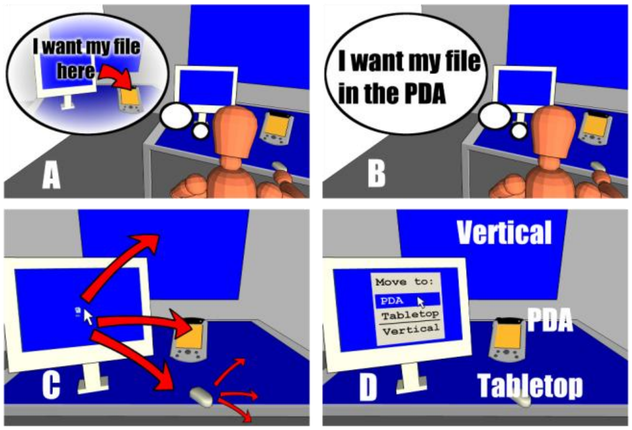

Figure 4. Match between the referential domains of task and technique.

Display configuration

In spatial terms, an MDE is defined by its display configuration. The display configuration depends on both the physical arrangement of the MDE (the positions and physical properties of its displays) and the input model of its interaction techniques (how spatial input commands are transformed into object movement within and between displays).

While the physical arrangement of an MDE is usually determined by the application, there is freedom for designers to choose different input models for any given MDE. Some of the obvious choices (e.g. the standard multi-monitor settings of current operating systems – Figure 5.A) are adequate for simple physical configurations, but do not scale for more complex arrangements (Figure 5.B).

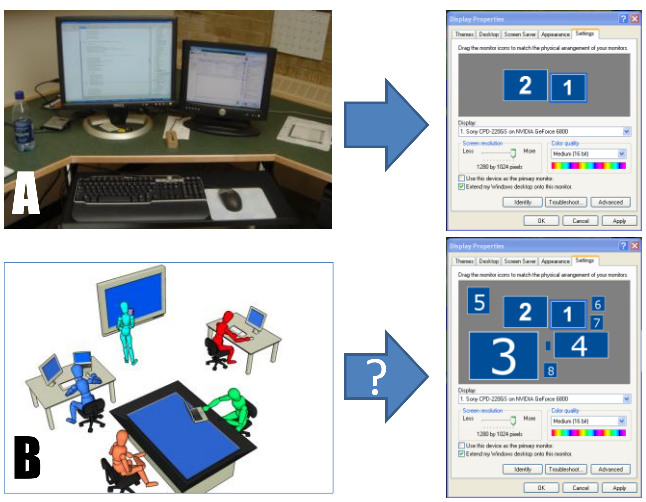

Figure 5. Physical arrangements and planar input models.

At this level I distinguish three types of input model: planar, perspective and literal. Planar models represent the displays in a two dimensional plane (e.g. the models in Figure 5, or the system described in [3]). Perspective models are those in which the behavior of the controlled objects depends on the position of the user, such as in our own Perspective Cursor [11], laser or finger pointing [4, 12] or techniques based on head pose (e.g. [2]). Literal techniques are those where the input and output spaces coincide (e.g. Pick-and-drop [15], Passage [13]).

In MDEs with complex physical arrangements (e.g. many displays of many sizes and in different orientations) we can expect literal and perspective techniques to show performance advantages because there is a better match between what users see and the gestures required by the technique. The advantage of perspective techniques over planar techniques is corroborated in a study that compared two perspective techniques with a standard planar technique [11]. We can expect the same kind of performance advantage for literal techniques, but these require physical access to the destination display and are not feasible for applications in which distances between displays are large or some parts of the display are not physically accessible [8].

Control paradigm

The lowest level of the framework concerns the execution of the cross-display action. Techniques can be classified into three different groups according to whether a feedback loop is present: closed-loop, open-loop and intermittent.

Closed-loop techniques are those that enable the correction of the gesture before it is finished by providing feedback as the action unfolds. Standard manipulation of a cursor within a display is closed-loop because the user can see the cursor at all times and adjust the gesture while it is being performed.

Open-loop techniques lack a feedback loop either because the system does not provide real-time feedback or because the gesture is too fast. For example, Flick [14] is an open loop technique because the gesture is fast, and feedback on the result is only provided after the action is finished.

Intermittent techniques may be open-loop or closed-loop during different parts of the gesture. Intermittent techniques arise in MDEs because displays are typically not perfectly contiguous and the space between displays cannot provide feedback, creating a discontinuity in the feedback loop. The clearest example of this category is Baudisch et al.'s Mouse Ether [1]. In this technique motor space is made to match the physical space; the assumption is that a correspondence between the cursor movement and the mouse movement is preferable to warping the cursor from the end of one display to the start of the next (i.e. motor-visual consistency is more important than uninterrupted feedback).

Open-loop techniques can be faster than closed-loop techniques, although at the cost of accuracy [14]. We have also investigated the trade-offs between closed-loop and intermittent feedback in dual-monitor desktops with different degrees of separation between displays, and found that closed-loop is superior to intermittent, even though the cursor warps from one display to the next [9].

CONCLUSION

MDEs need to implement interaction techniques that allow the movement of elements from one display to another. Although many techniques have been proposed, it is difficult to select the appropriate techniques for a particular system.

In this work I addressed this problem by creating a framework that analyzes the object movement process and categorizes existing techniques according to three levels: how the destination is represented by the task and the technique, how the input model of the technique maps to the physical arrangement of displays, and whether a feedback loop is kept during all, none, or parts of the execution of the cross-display action.

Designers of MDEs can use this work to learn about the range of options and to make informed design decisions with regard to cross-display object movement techniques. Researchers can use the framework as a common language to describe the design space of cross-display interaction, to better understand similarities and differences between existing techniques, and to find new types of techniques that have not yet been explored.

REFERENCES

1. Baudisch, P., Cutrell, E., Hinckley, K., and Gruen, R. 2004. Mouse ether: accelerating the acquisition of targets across multi-monitor displays. In *CHI '04 Extended Abstracts.I* ACM, New York, NY, 1379-1382.
2. Benko, H. and Feiner, S. 2005. Multi-monitor mouse. In *CHI '05 Extended Abstracts.* ACM, New York, NY, 1208-1211.
3. Biehl, J. T. and Bailey, B. P. 2004. ARIS: an interface for application relocation in an interactive space. In *Proceedings of Graphics interface 2004*. ACM International Conference Proceeding Series, vol. 62. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, 107-116.
4. Bolt, R. A. 1980. "Put-that-there": Voice and gesture at the graphics interface. In *Proceedings of the 7th Annual Conference on Computer Graphics and interactive Techniques*. SIGGRAPH '80. ACM, New York, NY, 262-270.
5. Grudin, J. 2001. Partitioning digital worlds: focal and peripheral awareness in multiple monitor use. In *Proceedings of CHI'01.* ACM, New York, NY, 458-465.
6. Johanson, B., Ponnekanti, S., Sengupta, C., and Fox, A. 2001. Multibrowsing: Moving Web Content across Multiple Displays. In *Proceedings of the 3rd international Conference on Ubiquitous Computing*. Springer-Verlag, London, 346-353.
7. Nacenta, M., Aliakseyeu, D., Stach, T., Subramanian, S. and Gutwin, C. (2007) Two Experiments on Co-Located Mobile Groupware, Technical Report HCI-TR-07-01. *http://hci.usask.ca/publications/2007/TR-1.pdf* Last accessed 11/07/2008.
8. Nacenta, M. A., Aliakseyeu, D., Subramanian, S., and Gutwin, C. 2005. A comparison of techniques for multi-display reaching. In *Proceedings of the CHI '05.* ACM, New York, NY, 371-380.
9. Nacenta, M. A., Mandryk, R. L., and Gutwin, C. 2008. Targeting across displayless space. In *Proceedings of CHI '08*. ACM, New York, NY, 777-786.
10. Nacenta, M. A., Sakurai, S., Yamaguchi, T., Miki, Y., Itoh, Y., Kitamura, Y., Subramanian, S., and Gutwin, C. 2007. E-conic: a perspective-aware interface for multi-display environments. In *Proceedings of UIST '07*. ACM, New York, NY, 279-288.
11. Nacenta, M. A., Sallam, S., Champoux, B., Subramanian, S., and Gutwin, C. 2006. Perspective cursor: perspective-based interaction for multi-display environments. In *Proceedings of CHI'05.* ACM, New York, NY, 289-298.
12. Parker, J. K., Mandryk, R. L., and Inkpen, K. M. 2005. TractorBeam: seamless integration of local and remote pointing for tabletop displays. In *Proceedings of Graphics interface 2005*. ACM International Conference Proceeding Series, vol. 112. Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, 33-40.
13. Prante, T., Streitz, N., and Tandler, P. 2004. Roomware: Computers Disappear and Interaction Evolves. *Computer* 37, 12 (Dec. 2004), 47-54.
14. Reetz, A., Gutwin, C., Stach, T., Nacenta, M., and Subramanian, S. 2006. Superflick: a natural and efficient technique for long-distance object placement on digital tables. In *Proceedings of Graphics interface 2006*. ACM International Conference Proceeding Series, vol. 137. Canadian Information Processing Society, Toronto, Ont., Canada, 163-170.
15. Rekimoto, J. 1997. Pick-and-drop: a direct manipulation technique for multiple computer environments. In *Proceedings of UIST '97*. ACM, New York, NY, 31-39.
16. Swindells, C., Inkpen, K. M., Dill, J. C., and Tory, M. 2002. That one there! Pointing to establish device identity. In *Proceedings of UIST '02*. ACM, New York, NY, 151-160.