Novel Interaction Techniques for Overlapping Windows

Michel Beaudouin-Lafon
Laboratoire de Recherche en Informatique
Bâtiment 490 - Université Paris-Sud
91405 Orsay - France
mbl@lri.fr - http://www-ihm.lri.fr

ABSTRACT
This note presents several techniques to improve window management with overlapping windows: tabbed windows, turning and peeling back windows, and snapping and zipping windows.

KEYWORDS: window management, interaction technique

INTRODUCTION
The dominant model for window management in desktop interfaces is overlapping windows. As the number of windows on the screen increases, the task of flipping between windows becomes more and more tedious and time-consuming. Previous work, e.g. [2,5], has addressed this issue with tiled windows. This note presents techniques to improve this situation with overlapping windows.

TABBED WINDOWS
Many commercial systems already use dialog boxes with several pages accessible through tabs. We introduced the concept of tabbed windows in the CPN2000 application [1]. It raises the tab technique to the level of document windows and gives the user the additional flexibility of moving pages from one window to another by dragging them via their tabs (Fig. 1). A page can be dragged to an existing window, which adds a tab to it, or to the background, which creates a new window with a single page and tab. Dragging the last page out of its window deletes the window.

Our experience with this technique in CPN2000 shows that tabbed windows dramatically reduce the number of windows on the screen while still providing quick access to a large number of pages that would otherwise be separate windows. Users thus control their working sets of pages and windows without sacrificing efficiency.

ROTATING AND PEELING BACK WINDOWS
Windows in desktop environments are almost always rectangular, with the sides parallel to the sides of the screen. When two windows of the same size overlap, at most two sides of the rear window can be seen. When many windows of similar sizes overlap, many of them become invisible, making them difficult to access.

By contrast, while physical books and sheets of paper on a desk are also rectangular, they are rarely perfectly aligned. This gives the user clues for locating them and an easy way to access them even when they are stacked.
SNAPPING AND ZIPPING WINDOWS

Users often work with windows that are strongly related, such as an outline and a page layout of the same document, or a document window and some tool palettes. Some applications support tiling, placing several panes together inside a window: Microsoft Word can split a window to display two views of a document, Netscape Mail can split a window into a list of folders, a list of messages and the current message. However, the user has little flexibility in organizing these panes.

**Peeling windows** allows the user to assemble several windows into a single entity. When a window is moved to that one of its sides is close to the side of another window, the two windows are snapped together and stay snapped unless the user moves the window away before releasing the mouse button. The interaction is similar to snap dragging [1] or magnetic guidelines [2]. Note that a window can be snapped inside, overlapping the other window.

A snapped window is slaved to its master window: moving the master window moves its slave, while moving the slave unsnaps it from the master window. This works well when the snapped window is small compared to the main one, e.g., when it is a tool palette. When windows are of similar sizes along the side being snapped, snapping becomes *zipping*: the two windows are given the same size along their common side, which becomes a divider line that can be moved. The windows are then moved and resized together, and they are unzipped by double-clicking the divider. Finally, snapped windows can be collapsed and reopened by clicking their tabs. MacOS 8.x/9.x has similar pop-up windows except that they can only be at the bottom of the screen.

CONCLUSION

We have presented several techniques that improve window management by extending the metaphor of overlapping windows. All these techniques have been implemented, and preliminary results are encouraging. Our future work will create a full window manager based on these techniques and we plan to conduct more formal usability studies.