

Floating Pie Menus: Enhancing the functionality of Contextual Tools

Jose Moreno Rubio and Paul Janecek

Database Laboratory

Swiss Federal Institute of Technology (EPFL)

1015 Lausanne, Switzerland

+41 21 693 6656

<jose.morenorubio, paul.janecek>@epfl.ch

ABSTRACT

This paper describes several design enhancements we have added to pie menus: (1) transparency and repositioning to allow a user to more easily see the model they are working on, (2) display of “contextual information” in the menu to show the current state of the object, (3) “lock-mode” to apply multiple commands to the same object without closing the menu, and (4) “dynamic target” to apply commands to multiple objects. The result is a new contextual interaction tool that mixes the dynamic aspect of pie menus with the persistence of floating palettes.

Keywords

Pie Menu, contextual menus, interaction techniques.

INTRODUCTION

Graphical interfaces often rely on some combination of toolbars, contextual menus and dialog boxes to manage access to commands. Our lab recently developed a graphical schema editor for modeling spatio-temporal databases. Objects and relationships in these models are often complex with many properties to specify. Navigating through the interface to find and select one of the hundreds of commands is a complex and difficult task. Contextual menus, such as pie menus, potentially simplify this navigation by using the type of the object of interest as a filter to show only the set of relevant commands.

In this paper we describe several extensions we have made to pie menus to overcome some of the weaknesses of traditional menus. First, the menus are transparent and can be repositioned to allow the user to more easily see the model they are working on. Second, information about the current state of the object is shown in the menu next to related commands. Third, the menu can be locked open allowing the user to apply multiple commands to the same object without repetitively navigating the menu hierarchy. And finally, once the menus are locked open, they can be dynamically attached to another object of the same type.

PIE MENUS

Pie menus [1] are radial menus with the same functionality as contextual menus. When the pie menu is shown, the pointer is positioned in the center of the circle and all items are at an equal distance. Similar to other menus, they can be structured hierarchically. Users are able to remember the angular position of items in the menu more easily than linear menus, and inverse positions can be used to pair actions with opposite effects, such as copy and paste.

Pie menus have the same functionality as pop-up menus: the user opens the menu and then makes a selection that causes an action. The action can directly activate a command, or open a dialog box where the user can change several properties. When the user makes a selection, the menu disappears. “Marking menus” [3] [4] [5] are an extension of pie menus that allow the user to specify a command with simply the gesture they would normally trace to navigate to the command.

Pie menus have several disadvantages. They use a large amount of screen space, and the number of elements in a single menu is limited to eight [2]. To apply multiple commands, the user must repeatedly navigate the menu hierarchy. In a complex interface with hundreds of commands, this can become an extremely repetitive and tedious task. We have added several extensions to pie menus that address these problems.

EXTENDED FUNCTIONALITY

Repositioning

First, the “floating” pie menus are transparent and can be repositioned. Transparency allows the user to see through the menu to the underlying object. We have observed that when the level of transparency is too high, the user can confuse commands between submenus, and when there are more than three levels of pie menus the background is no longer visible. Repositioning the menu allows the user to see the model even when there are multiple overlapping submenus. The pie menu, along with any submenus, is moved with the left mouse button.

Contextual information

The properties of an element in a model are not always displayed in the graphical representation. For example, properties can be hidden to reduce visual complexity. We have added “contextual information” to the pie menus to show the value of properties next to the commands that allow the user to set them. This information is shown in a second line in the menu slice with a different color than the label, and is updated dynamically.

Lock-mode

When a user makes a selection in a menu, the entire menu hierarchy is closed. If a user wants to change several properties of an object that are located in the third level of the hierarchy, they must navigate through all three levels each time they apply a command. We have added a “lock-mode” to lock the menu open at any level in the hierarchy.

The small circle in the center of the pie menu is divided in two: the upper-semicircle cancels the pie menu and the lower-semicircle locks the pie menu. The user can also lock more than one level of the hierarchy to improve the navigation within the menu structure.

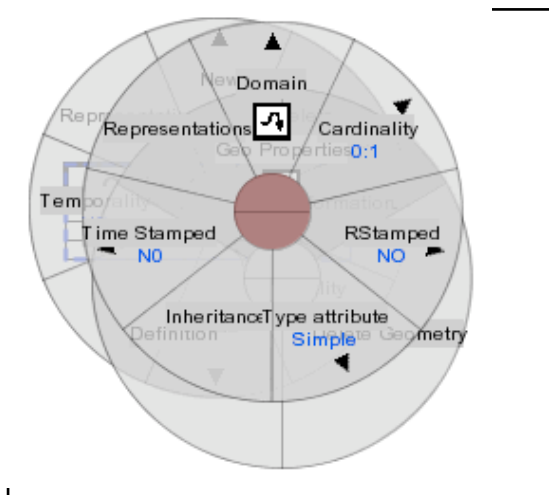


Figure 1: The floating pie menu. The contextual information is shown in blue. The red color in the center indicates the menu is in “lock-mode.”

Dynamic target

With the “dynamic target” functionality the user is able to attach a locked menu to another object, allowing them to change properties on different elements without closing

the pie menu. For example, to apply commands to three different objects, it would be necessary to open a contextual menu for each object. With this solution, the user can open the pie menu of an object, lock it, and apply a command. To work with another object, the user selects it with the right button of the mouse. The contextual information in the pie menu is updated to reflect the state of the new object.

CONCLUSIONS

The “Floating” Pie Menu is a tool that combines the object-centric properties of contextual menus and floating palettes. Transparency and repositioning allow the user to more easily see the model while they access the menu, and contextual information allows the user to inspect the state of the object, even for properties that may not be apparent in the model. Furthermore, the lock-mode and dynamic target allow the user to apply multiple commands to one or more objects without re-navigating the menu hierarchy. The result is a hybrid tool that offers a novel combination of interactivity.

ACKNOWLEDGMENTS

We want to thank Jason Hong for letting us use the java implementation of pie menus developed in the SATIN project of the “Group for User Interface Research” at the University of California, Berkeley [6].

REFERENCES

1. D. Hopkins. The design and implementation of pie menus. In *Dr. Dobb's Journal* 1, 1991, 6:12 pp, 16-26
2. J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An Empirical Comparison of Pie vs. Linear Menus. In *Proceedings of CHI'88*, pages 95-100, 1988
3. G. Kurtenbach. *The Design and Evaluation of Marking Menus*. PhD thesis, University of Toronto, 1993
4. G. Kurtenbach and W. Buxton. User Learning and Performance with Marking Menus. In *Proceedings of CHI'94*, pages 258-264, 1994
5. Mark A. Tapia, Gordon Kurtenbach. Some design refinements and principles on the appearance and behavior of marking menus. In *Proceedings of UIST'95*, pages 189-195, 1995
6. Pie Menus, the SATIN project. <http://www.cs.berkeley.edu/~jasonh/download/software/piemenu>