GAME PROGRAMMING Gems 3

Charles River Media

EDITED BY DANTE TREGLIA

1.14

Natural Selection: The Evolution of Pie Menus

Don Hopkins

don@DonHopkins.com

Pie menus are a naturally efficient user-interface technique—directional selection of pie slice-shaped targets. The cursor starts out in the inactive center region of a pie, and all target slices are large, nearby, and in different directions. Pie menus are quite easy for new users. You simply follow the pop-up directions to use them. They are also extremely efficient for experienced users. Once you know the directions, you can quickly and reliably 'mouse ahead' without looking. Fitts' Law [Fitts54] explains the pie menu advantage—their fast selection speed and low error rate is due to their large target size and the small distance between each item.

The evolution of user interface design is driven not only by theory, but also by practice. We'll examine the successes and failures of a few real-world examples, not only to avoid re-inventing the square wheel, but also to encourage further creativity. The examples presented here are intended to inspire you to think outside the box and design new kinds of fun, efficient, and reliable user interfaces.

The Feng GUI of Pie Menus

User interface design is not just a process of raw artistic creation nor a legalistic application of interface guidelines and theories. It's the exploration and discovery of naturally efficient ways of solving problems, given competing sets of constraints. The outcome is always different, because the trade-offs and constraints always vary, but many of the underlying principles are universal.

'Feng GUI' seeks to understand the dynamic flow of mental and physical energy. It orchestrates the flow of attention and gesture throughout the interface as a whole. Fitts' Law is useful for scientifically analyzing performance speed and error rate, but it doesn't capture the human side of the equation. Feng GUI tries to prevent unfortunate accidents (like the 2000 Florida presidential election 'Butterfly Ballot') before they happen.

When designing a pie menu, think of Martha Stewart arranging a bunch of flowers into a beautiful bouquet. You must work with what you're given, try to play off the

visual and semantic symmetries and relationships, and arrive at a pleasing pattern that's both enjoyable and easy to remember.

To construct a memorable pie menu tree of submenus, you should emulate Alexander Calder's creating a hanging mobile sculpture. The task not only requires a sound understanding of scientific engineering principles, but also aesthetic judgment calls and acrobatic balancing acts.

Doug Engelbart, who invented the mouse and pioneered interactive user interfaces, strongly believes that the human-tool co-evolution should be based on rigorous exploratory use in a wide variety of real-world applications. So don't just talk about pie menus—use them, evaluate their performance, and improve upon them!

Researching and Evaluating Pie Menus

The essential idea of directional menu selection has been around for a long time in various forms and with different names. Many examples of implementations exist, and a detailed history can be found at [PieMenu02].

Many studies have also been done on their effectiveness compared to other UI approaches. Gordon Kurtenbach and Bill Buxton (University of Toronto) have demonstrated many interesting results with their empirical research and controlled experiments with marking menus and various input devices. At Alias|Wavefront, they have successfully applied them to Maya, a high-end 3D animation environment, so users can design their own marking menus to customize their environment. In their research, they studied the learning curve from novice to expert user. They found that there are three stages of behavior along the learning curve:

1. Novice users click up the menu, wait for it to display, look for the desired label, move the mouse, and click to select the highlighted item.
2. Intermediate users remember the direction, click up the menu, move in a desired direction, wait for the menu to pop up and highlight the desired item, and release the button to confirm the selection.
3. Expert users simply press down the button, move in a desired direction, and release the button without hesitating.

Because the physical motions of novice, intermediate, and expert users are the same, pie menus transparently train you to become an expert. Each time you make a selection, you're rehearsing the expert mouse-ahead gesture. The intermediate stage is like an escalator along the learning curve. It helps novice users become experts by exercising their skills and increasing their confidence to mouse-ahead. Your muscles quickly and unconsciously learn to mouse-ahead without looking.

Jaron Lanier (VPL Research) put it well: "The mind may forget, but the body remembers." Pie menus exploit your body's ability to remember muscle motion and direction, even when your mind has forgotten the names of the corresponding items.

The nature of the input device used has a significant effect on the selection speed and error rate. Mice have been found to be faster and more accurate than trackballs, and pens are faster and more accurate than mice.

The maximum usable breadth (number of items) and depth (submenu nesting level) is limited by the maximum error rate the application can tolerate. Nuclear power-plant interfaces should stick to single level-two and four-item pie menus, which are extremely reliable. A game like *SimCity* or an editor like Maya can get away with using deeper menus with more items because it's easier to recover from selecting the wrong item.

Experienced users perceive single-level pie menus with two, four, and six items to be error-free, and eight items to be very reliable. Kurtenbach and Buxton measured the error rate at less than 10% with four items four levels deep as well as with eight items two levels deep.

Increasing the number of items in a pie menu has an obvious detrimental effect on the selection speed and error rate, but the relationship is not simply linear. Even numbers of items are easier to use and remember because more of the items are on-axis and symmetrical. On-axis items are easier to select than off-axis items, so it's good to put commonly used items to the North, South, East, and West, and less-common items along the diagonals.

This even/odd effect is most pronounced when comparing 7 versus 8 items, and 11 versus 12 items. Eight and 12 items are especially easy to use, because the directions are mentally more familiar and physically more on-axis. As the number of items increases, the negative effect of adding another item decreases. So, it's often helpful to add an extra item to 11-, 7-, and even 3-item menus, just to make them nice and even.

When designing nested pie menus, the depth versus breadth trade-off seems to be about even. So, it's best to let the semantics of the items determine how they should be arranged: shallow menus with many items, or deep menus with few items.

It's worth noting that some menus still work better as linear menus. Most linear menus and submenus aren't arranged to take advantage of the pie menu directions, and pie menus with too many items are huge and unwieldy. To solve those problems, modifiable pie menus have been developed that the user could customize, and scrolling and paging pie menus can handle any numbers of items.

Plugging in Pie Menus

Component technologies, like ActiveX and Dynamic HTML behaviors, make it possible to implement general-purpose, easily reusable plug-in user-interface components. Pie menus can provide configuration languages, property sheets, and special-purpose editors, which enable designers and users to create and customize their own menus without programming.

ActiveX (also known as COM and OLE) is a component technology developed by Microsoft. We developed an open-source ActiveX pie menu component that can

be plugged into any OLE control container, including those used by Internet Explorer, Visual BASIC, Visual C++, and many other tools and applications. They're easily created and customized through scripting languages like Visual BASIC or JavaScript, and they have property sheets to configure their many options, for editing, and to preview the pie menus (see Figure 1.14.1).

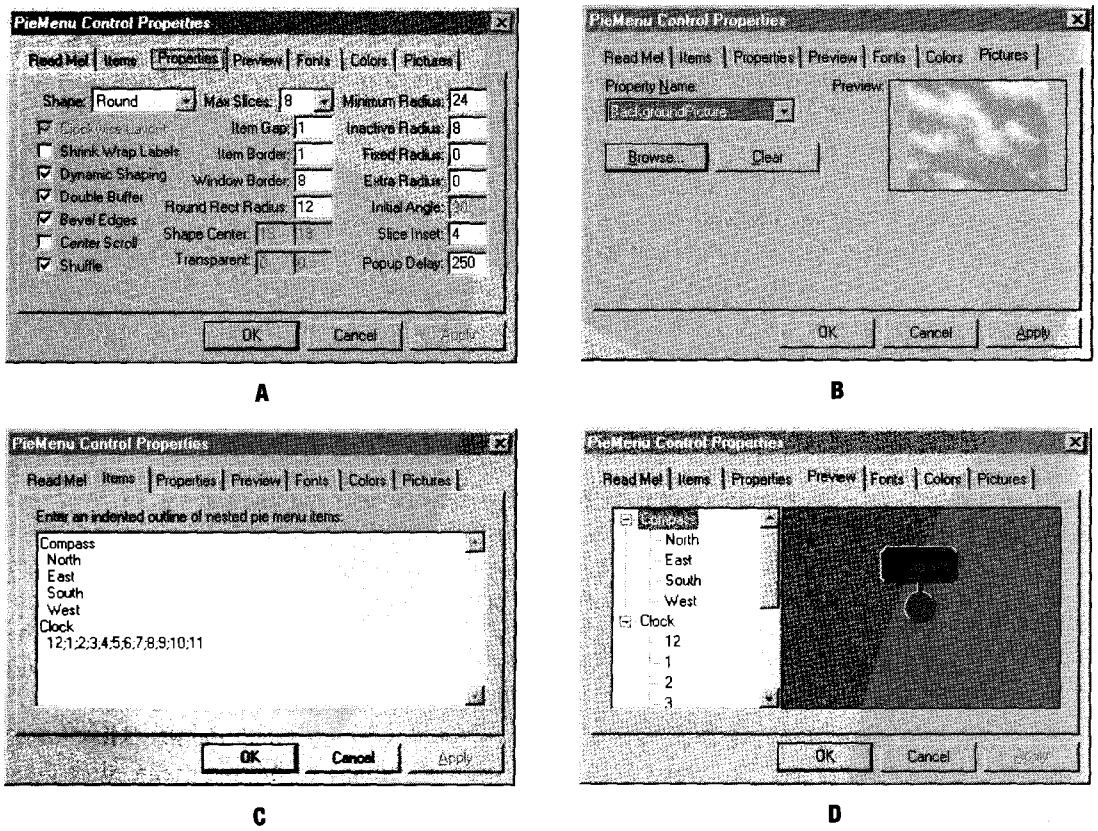


FIGURE 1.14.1 (A-D) *Editing control properties using an example program.*

ActiveX pie menus support many properties and methods to control their appearance and behavior. You can customize pie menus by writing scripts that manipulate their properties, call methods, and handle callback events signaled during tracking. However, their graphical abilities are quite limited when compared to Dynamic HTML (see Figure 1.14.2).

The open-source JavaScript pie menus for Internet Explorer solve this problem nicely [JavaScript02]. They're tightly integrated with the Web browser and can take advantage of all of its features. They're easily and completely configured in XML as well as being extremely flexible, because you define their appearance with Dynamic

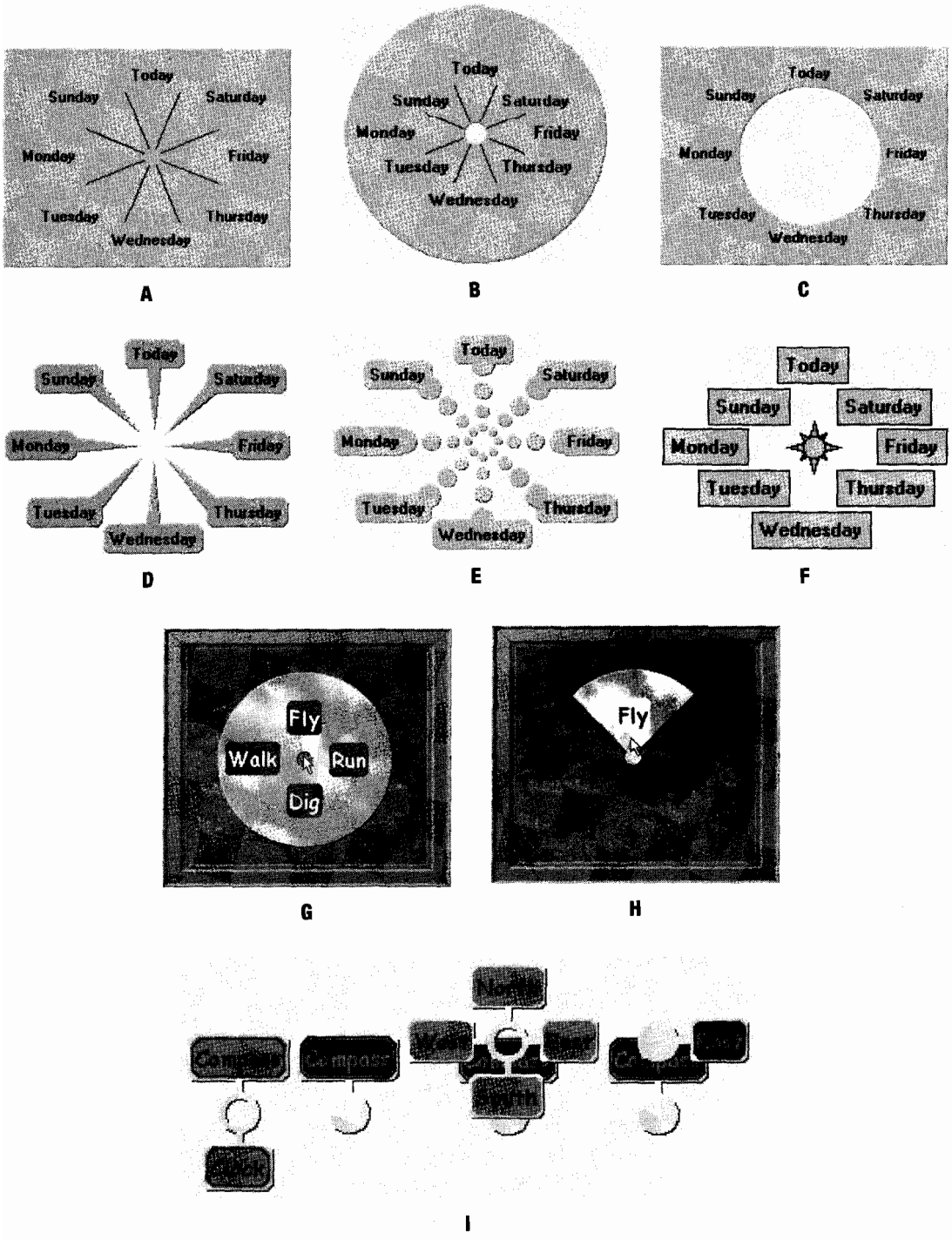


FIGURE 1.14.2 (A-I) *Pie menus implemented using Dynamic HTML.*

HTML. These menus a easy for Web page designers to use for static pages and for Web server programmers to use for dynamic online services because they're implemented as modular 'Dynamic HTML Behavior Components.'

JavaScript pie menus are specified in XML, so it's possible for people to manually write them with a text editor. It is also possible for programs to dynamically generate them from a database. The JavaScript pie-menu component code is cleanly distinct from the Web page and XML pie-menu specification. You can customize their behavior by writing event handlers on the Web page in JavaScript, VBScript, or other languages. They can provide rich, dynamic graphical feedback, because scripts can reach into the pie menus and Web pages, and actually modify the Dynamic HTML on the fly.

Using XML to specify pie menus has many advantages. The format is independent of the implementation, so the same pie menus can be used across many different platforms. Web servers and browsers can automatically transform application-specific XML formats into pie menus by using standard XML-processing tools, like XSTL and distributed XML databases. For example, an XSLT style sheet can dynamically generate a Web page with 'Punkemon' pie menus, based on an XML database of trading-card attributes and links to animations [Punkemon02] (see Figure 1.14.3).

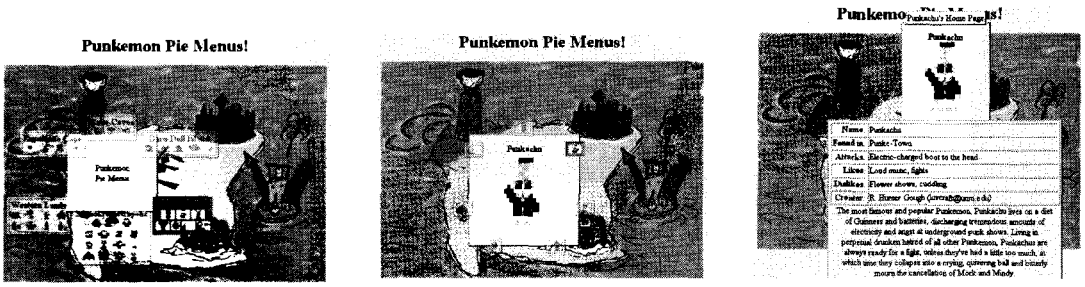
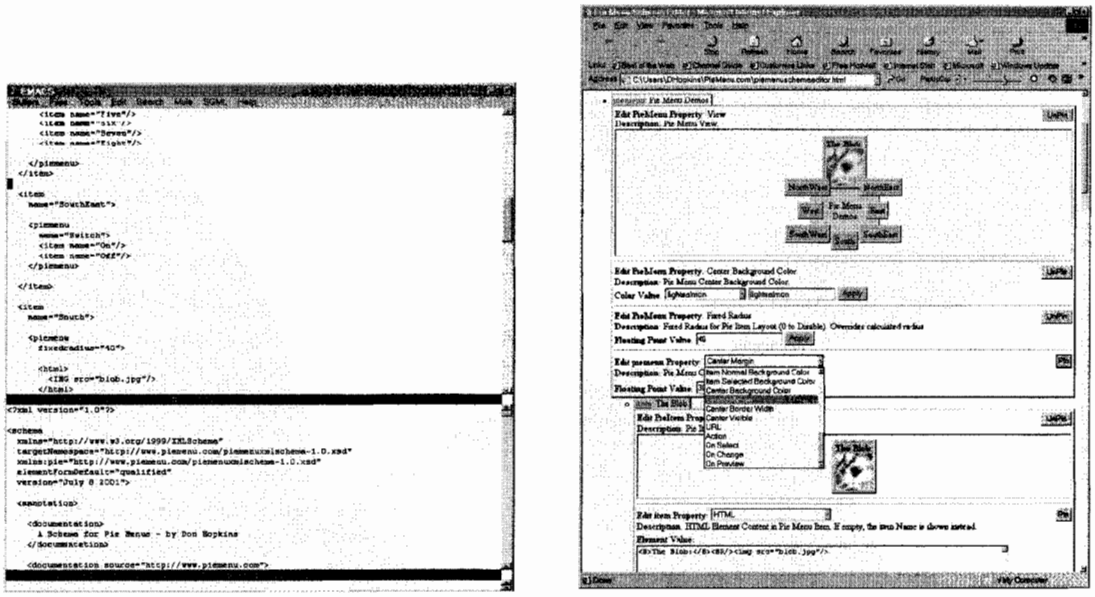


FIGURE 1.14.3 (A-C) *The 'Punkemon' example.*

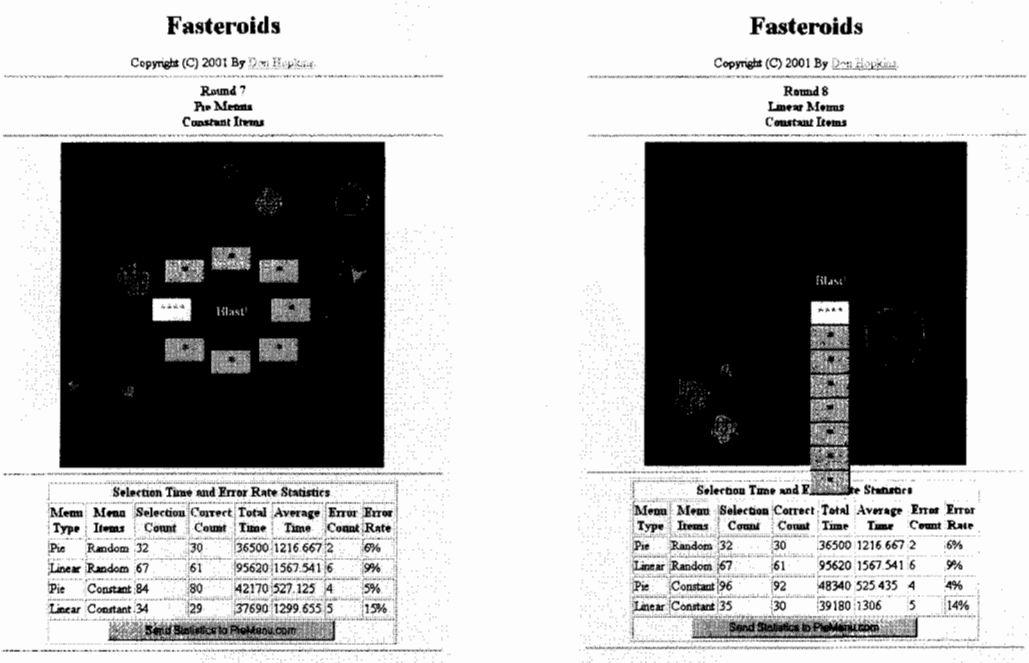
The XML pie menu schema enables editors to automatically validate, construct, and edit pie menus. The pie menu schema is on the CD-ROM as well as [PieSchema02]. An example editor can be seen in Figure 1.14.4, which is available online [PieEditor02].

Fasteroids [Fasteroids01] is both a real-time video game and an empirical user interface experiment; it enables you to compare linear menus and pie menus. The JavaScript pie menus also support the old-fashioned linear menu style, and they can be instrumented to record the selection time for experimental purposes. *Fasteroids* alternates between pie menus and linear menus (as shown in Figure 1.14.5), and prompts you to select a certain item to blow up the asteroids. It records and displays



A B

FIGURE 1.14.4 (A-B) *Pie menu schema and editor.*



A B

FIGURE 1.14.5 (A-B) *The Fasteroids game and experiment.*

the average selection time and error rate, so you can compare pie menus and linear menus for yourself.

Future Directions

Pie menus work well with touchscreens on handheld devices like the Palm Pilot and the Pocket PC. 'Finger Pies' are easy enough to use with your finger, so no pen is required. A product we have developed called ConnectedTV makes your handheld into a customizable entertainment guide and remote control that's designed to be held in one hand and operated with the thumb and finger. ConnectedTV lets you use finger pies to make your own personalized television schedule, filter, and program-search guide; you can flip back and forth through show descriptions and movie reviews, and send infrared remote-control commands to change the TV channel and operate other equipment.

Fast, inexpensive motion detectors that are sensitive enough to detect the direction of gravity will soon be built into consumer electronic equipment like cell phones, handheld computers, remote controls, and games. Motion detectors will enable convenient one-handed scrolling, dialing, panning maps, tilting pie menus, continuous gesture recognition, and many other exciting interaction techniques.

Going to Town with *SimCity*

In 1991, we first ported *SimCity* to Unix. It featured pie menus (see Figure 1.14.6) for quickly selecting *SimCity* editing tools, which was a useful shortcut for the original *SimCity* command palette.

Static pie menus whose items don't change can be carefully designed for ease of use and nicely illustrated for aesthetic appeal. We translated the *SimCity* tool palette into a convenient set of static pie menus. Icons in the pie menus are arranged in the same pattern as the palette, so they're easy to learn and quick to use.

Pop-up pie menus let you quickly switch tools without moving back and forth between the map and the tool palette. You soon learn to mouse-ahead through the pie menus and submenus just by flicking in the appropriate directions. Thanks to Moore's Law, you can now run *SimCity* so fast, it's a strategy twitch game, running at decades per second. Thanks to Fitts' Law, pie menus help you keep up with accelerated *SimCity* time by mousing-ahead without looking, and without wasting centuries dragging though linear menus.

The icons of the *SimCity* tool palette and pie menus are different sizes and shapes than the original, square *SimCity* tool icons. Their sizes and shapes are related to the prices and functions of the tools. Small icons stand for inexpensive tools, like parks or bulldozers. Large icons stand for expensive tools, like power plants or airports. Long icons suggest linear tools, like roads or railroads. And square icons are for square buildings, like residential zones or fire stations. The purpose behind this oddball icon design is to make remembering and differentiating between them easier (see Figure 1.14.6).

FIGURE 1.14.6 (A-C) *SimCity's pie menus.*

Living at Home with *The Sims*

The pie menus in *The Sims* use a combination of desaturation, darkening, and alpha blending to feather the edges of the menu (see Figure 1.14.7 and Color Plate 1). This was done because we didn't want the pie menus to obscure too much of the scene behind them. You can see through the pie menu as the animation continues on in real-time behind it. The head of the currently selected person is drawn in the center of the pie menu and follows the cursor by looking at the currently selected item.

It was necessary to somehow separate the head from the rest of the scene. Otherwise, it looked like a giant head was floating in a room of the house, which was somewhat disconcerting and violated the 'Principle of Least Astonishment.' Simply drawing a solid menu background would obscure too much of the scene behind the menu. Using a partially transparent menu background still did not visually separate the head from the background scene enough. It looked muddy and cluttered, instead of crisp and bright.

Instead of simply alpha-blending the menu background, we actually lowered the contrast, which darkens the image and desaturates the background. The effect was to cast a colorless shadow with soft, feathered edges over the animated background, against which you can easily see the head and menu item labels.

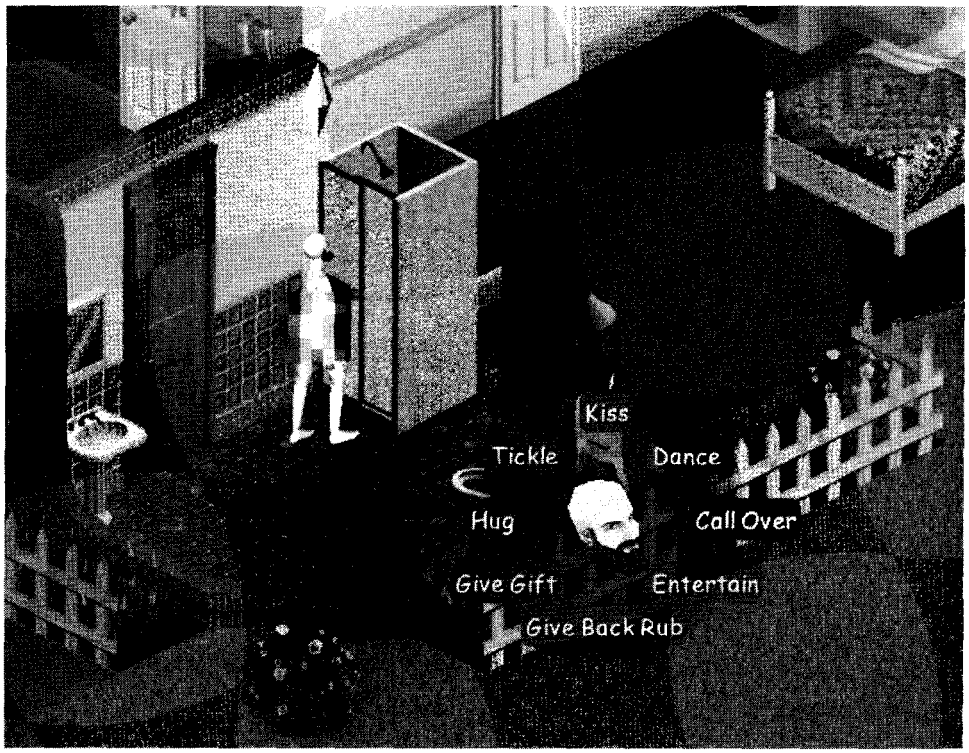


FIGURE 1.14.7 *Pie menus in The Sims.*

Instead of drawing a circular edge around the pie menu, the gray shadow gradually tapers off, suggesting that the active pie menu target area is not confined to a small circle. The labels are drawn around the pie menu center with high-contrast drop shadows, so they're easy to read.

The animated head in the center needed to look sharp and bright against the pie menu background. So, the shadow effect looks at the *Z* buffer to clip around the head in the menu center, keeping it crisp and bright. That gives it visual 'pop,' which clearly separates the user interface from the world, without drawing dividing lines or unnecessary visual clutter.

Conclusion

Pie menus benefit from the natural consequences of Fitts' Law. They're neither the only nor the best user interface technique to take advantage of this effect. However, they're a great improvement over today's standard linear menus and a stepping stone to developing even better user-interface techniques.

The proven advantages that pie menus have over linear menus are their higher speed and lower error rate. They also have the potential to be carefully designed and conveniently automated in many different ways, which increases their usefulness for many applications.

Computer games and handheld consumer electronic devices are reshaping the way user interfaces are designed because of their new and unusual demands. Real-time games require quick, responsive, engaging user interfaces. Handheld computers and phones must be useful for a wide range of people in real-world conditions, so they demand high reliability and ease-of-use.

Designing a good user interface requires balancing many competing demands and guidelines. It's extremely important not to squander the user's time or attention—consider it your most rare and precious resource. Don't get tripped up on metaphors—take a step back and look at what's really going on. Think in terms of the user's goals, mental models, and physical actions.

Be prepared to throw away your first design. Use your own system on an everyday basis. Continuously iterate the design, based on feedback from empirical testing and the users themselves. Every application and user has different requirements that demand different trade-offs at different times.

Designing good pie menus takes thought and effort, like writing Haiku. Limit the number of items in each menu, and group them together into memorable, balanced submenus. Arrange the items in natural directions to exploit their semantic relationships and physical associations. Don't exclusively use pie menus when other techniques are more appropriate, like sliders, scrolling lists, keyboards, or handwriting recognition. It's a good idea to provide multiple ways of accomplishing the same task when it makes the application easier to use.

Feng GUI seeks to integrate the lessons of real life, empirical research, and theoretical principles, and apply them to the enlightened design of efficient, reliable user

interfaces. The "Butterfly Ballot" debacle demonstrated how badly designed user interfaces can have enormously consequential effects on the real world. By striving to design user interfaces with good Feng GUI, you can improve people's lives and affect the world in many positive ways.

References

[Fasteroids01] Hopkins, Don, *Fasteroids*, available online at http://www.PieMenu.com/fasteroids.html, March 2002.

[Fineman01] Fineman, Howard, "Unsettled Scores," *Newsweek*, September 17, 2001.

[Fitts54] Fitts, P. M., "The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement," *Journal of Experimental Psychology*, 1954: Vol. 47, pp. 381–391.

[JavaScript02] Hopkins, Don, "Open Source JavaScript Pie Menus," available online at http://www.PieMenu.com/JavaScriptPieMenus.html, March 2002.

[PieEditor02] Hopkins, Don, "Pie Menu Schema Editor," available online at http://www.PieMenu.com/piemenuschemaeditor.html, March 2002.

[PieMenu02] Hopkins, Don, "Pie Menu Central," available online at http://www.piemenus.com/, March 2002.

[PieSchema02] Hopkins, Don, "Pie Menu XML Schema," available online at http://www.PieMenu.com/piemenuxmlschema-1.0.xsd, March 2002.

[Punkemon02] Hopkins, Don, "Punkemon Pie Menus," available online at http://www.PieMenu.com/punkemon.xml, March 2002.