FRANK McKIEL, JR.

AUDIO-ENABLED GRAPHICAL USER INTERFACE FOR THE BLIND OR VISUALLY IMPAIRED

An audible feedback scheme has been developed to allow a blind computer user to fully comprehend and use a windowed graphical user interface in the same manner as a sighted user. The user manipulates a mouse or similar input device to move a pointer around on the display. As the pointer passes over windows, controls, and other graphical features, stereo sounds are generated by special software to convey the identity of the features.

INTRODUCTION

In the early 1980s, both the traditional large computer and the rapidly proliferating personal computer were used primarily through text-based interfaces. Just like anyone else, blind users could type in commands or data and, with the help of a speech synthesizer, read the alphanumeric output on the computer screen. Because of the decreasing cost and aesthetic appeal of high-resolution graphics, color screen images have quickly advanced from being a luxury reserved for special graphical applications to being a bare necessity for any workstation.

At first, many applications took advantage of graphical capabilities but with little intuitiveness and consistency across applications. In the mid-1980s, the paradigm of a window-based interface emerged, exploiting the graphical environment as well as pointing devices such as the mouse. This type of interface offers the user several major advantages: The user can have several applications presented on the screen at one time, each one implemented so as to coexist as a window on the screen. The user can also arrange, size, and terminate each application through a standardized mouse or keyboard interaction. In addition, the user can employ a mouse to activate controls and icons within the windows to run an application, rather than having to memorize and type cryptic commands. Thus, a window-based graphical user interface (GUI) offers consistency and intuitiveness through the use of visual metaphors for real-world objects (push buttons and icons) and hand movements (mouse pointer).

Consequently, window-based GUI's are the current and foreseeable trend in applications software for computer workstations. Unfortunately, they tend to limit access by blind and visually impaired users who otherwise have well-developed means for accessing the traditional text-based applications.

GOALS AND DESIGN CONSIDERATIONS

The goal of this work is to provide a means for the blind to access GUI-operated applications. To accomplish this goal, special sounds are generated as the user moves the pointer over graphical features. Sound offers a very rich vehicle for feedback to the user because of the wide variety and complexity of sounds that can be produced electronically and deciphered by the human ear. In this work, stereo effects are particularly important for dramatically enhancing the user's sensation of the position and size of objects on the display.

The general concept of using sound to help the blind user sense a graphical environment is not new.[1] Several authors have promoted the idea of video-game–like sound effects emanating from icons.[2,3] The multisensory aspects are also appealing to sighted users and can reinforce the visual data to aid intuition and to reduce mistakes.

Another common idea is to use the pitch of musical notes or chords to convey information. At first, pitch alone would seem to be a most readily distinguishable sound attribute, but in practice it has limited dimensionality and causes a considerable fatigue effect.[4] Tone and rhythm sequences, however, may be useful to represent objects or ideas that have no obvious analogy among sampled sound effects.[5] In the current design, the pitch of pure tones is used sparingly for certain types of controls rather than being a mainstream feedback channel for navigation.

Very little work has focused on an overall approach for handling all aspects of a windowed environment that allows the blind user to navigate with full awareness of everything on the screen. In pursuit of such an overall approach, the present work includes a subtle, persistent feedback signal to let the user sense his or her screen position within open rectangular areas and to repeatably locate iconic objects and controls rather than just stumble across them by chance. This theme is applied more or less consistently to the screen background, within the interior of windows, and even within some rectangular controls.

In addition to making an interface that is merely operable, considerable human engineering has been applied to help user performance and promote usefulness in a day-to-day work environment. For example, the operation of the audio feedback has been streamlined so that the user can move as quickly as desired. The current implementation exhibits fast response and preemptive function so that the user is not slowed down by waiting for sound effects to occur or to reach completion. As needed, new sounds may interrupt other previously triggered sounds that have not yet completed. Further attention has been aimed toward minimizing contribution to fatigue and making the feedback unobtrusive so that a sighted user can share the same workstation without being hindered by the added effects.

To reduce fatigue, more complex sounds such as filtered noise and multiple oscillator phase-shifting chords are preferred over single-frequency tones. The sounds also diminish automatically within a few seconds if the mouse is not moved, thus making the system more pleasant in an open office environment.

The design philosophy emphasizes building an environment that is uniquely usable in allowing the sighted and nonsighted user to share the same experience of the graphical aspects of the interface; that is, the blind user accesses an application through its appearance on the screen rather than through some alternative representation. This approach largely eliminates the need for existing applications to be modified or rewritten to accommodate the blind. It also attempts to encompass future developments in graphical user interactions along the lines of direct manipulation of icons.

GENERAL OPERATION

To use the interface for the visually impaired, the user manipulates a mouse or equivalent input device to move a pointer around on the display. The blind interface software, along with readily available sound-generating hardware installed in the workstation, produces special stereo sounds as the pointer passes over windows, controls, and other graphical features. These sounds, which may combine musical tones, filtered noise, prerecorded sound effects, and synthesized speech, allow the user to probe and mentally map the graphical environment in a manner similar to using tactile feedback in the physical environment. In experiments to date, the technique has enabled the blind user to find and manipulate windows and standard controls on the display with the same understanding of the windowed environment and at nearly the same pace as the sighted user.

DETAILS OF OPERATION

The following list provides some examples of how the blind interface functions as the user moves the pointer around the screen:

1. Any time the pointer enters a new window, the title of the window is announced using speech synthesis.

2. When encountered by the pointer, the window that is currently active and "in front of" the other windows has noticeably louder overall sound effects than the other windows.

3. Each of the standard window features, such as title bar, sizing border, system menu bar, and minimize/maximize buttons, has a characteristic sound. For instance, the minimize button is represented by a tone that cyclically diminishes in amplitude every second or so. This sound reminds the user that pressing this button will cause the window to diminish in size to a minimized icon. Where possible, sounds are chosen so as to be metaphorically intuitive. A pull-down menu makes the sound of a window shade being drawn downward. When the pull-down menu is dismissed, the user hears the window shade roll up and slap against the roller.

4. Stereo balance and filtering effects are superimposed to indicate constantly the relative position of the pointer within a particular window (Fig. 1). As the pointer moves from left to right within a window, the sound effects "move" accordingly by adjusting the relative volumes of the left and right stereo channels. Vertical motion is accompanied by stepwise changes in filtering of the ongoing background sounds. This sound effect divides the space into horizontal bands to aid the user in locating objects at fixed vertical locations within a window. To be more specific, the window background sound, which consists of a weak nondescript chord along with some white noise, is passed through a single-pole filter whose center frequency can be varied to signify vertical movement. The user is sensing an average virtual pitch rather than listening to a pure tone.

5. Icons are identified to the user by prerecorded sounds or speech synthesis. For example, a metaphorical file-drawer icon labeled "Tax records" might appear within a window (Fig. 2). As the pointer passes over this icon, the sound of a file drawer unlatching and rolling out is produced, followed by the verbal announcement, "Tax

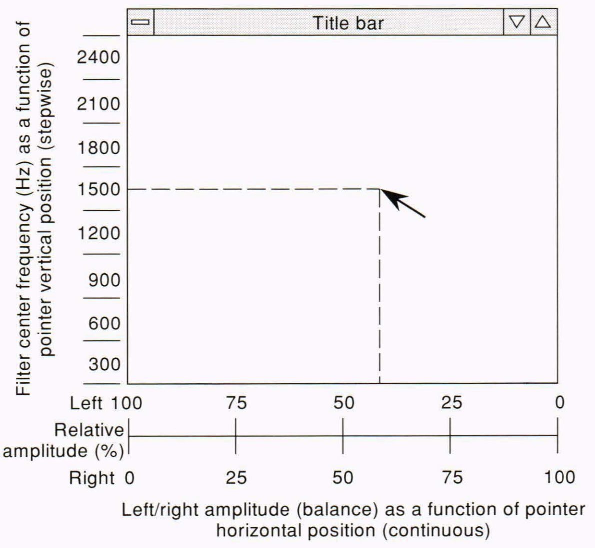

Title bar
Filter center frequency (Hz) as a function of pointer vertical position (stepwise)
2400
2100
1800
1500
1200
900
600
300
Left 100
75
50
25
0
Relative amplitude (%)
Right 0
25
50
75
100
Left/right amplitude (balance) as a function of pointer horizontal position (continuous)

Figure 1. Filtering and balance effects as a function of pointer position within a window.

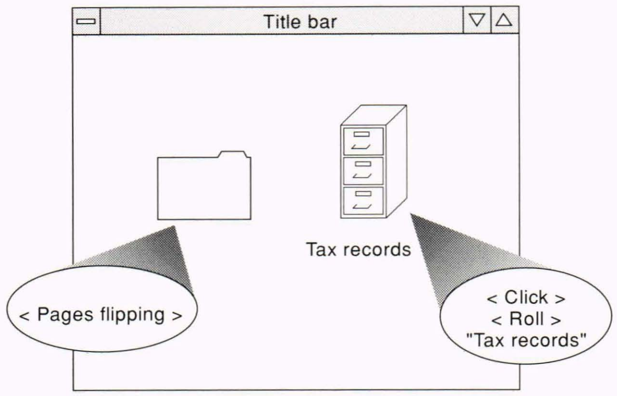


Figure 2. Examples of sound effects for file folder and file cabinet icons.

records." The sound-effect "prefix" of the file drawer sounds allows the user to quickly identify and pass over the icon when searching for something other than a file drawer. The preemptive sound generation will continue the sound only so long as the user keeps the pointer on the icon. Thus, the user is not slowed down by having to listen to the entire announcement.

6. Controls such as push buttons and check boxes within the window are handled in the same manner as icons. Additional sound or speech is used to indicate toggle status or to identify default conditions.

7. The basic sound-generation schemes are extended to encompass user handling of message boxes, pull-down menus, scroll bars, and select-drag-and-drop.

INITIAL USER TRAINING

The blind user is introduced to the concept of windows by use of stiff cardboard cutouts. Beads of glue or thick paint are applied to the surfaces of the cutouts to depict, in relief, the various borders and controls of a standard window, including the outer border, title bar, minimize/maximize buttons, and system menu bar. The user quickly learns the relative locations of these features. Two or more of the cutouts are then used to explain how windows may be relocated and lapped over one another. Other pieces are added to the basic cutout to depict the operation of scroll bars and controls within the interior of a window. This explanation takes about forty-five minutes. After another hour of explanation of the associated sounds and hands-on work with the audio-enabled viroment, the user is able to perform elementary functions of locating, activating, and arranging windows and icons.

Limited exposure to a cross section of blind users, ranging from those who have never been sighted to those who have had sight but have never worked with window-based computer programs, indicates that this initial training is effective but, like most skills, must be followed by practice before the user becomes comfortably proficient.

IMPLEMENTATION ENVIRONMENT

The hardware used for development so far includes an IBM PS/2 Model 70 computer with an IBM audio capture and playback adapter and a compact DECtalk (Digital Equipment Corp.) MultiVoice text-to-speech synthesizer (Children's Hospital Institute on Applied Technology, Boston, Mass.). The monaural output of the speech synthesizer can be mixed into both channels of the stereo signal from the audio card either through the card itself or through an external amplifier. The earliest work also included a digitizing tablet, but the development of stereo techniques made an absolute positioning input device unnecessary.

The original proof-of-concept version was implemented under DOS (actually the OS/2 "DOS-box") using discrete graphics function calls to create a simulated window-type graphical interface. This method of implementation was used to get to a proof-of-concept demonstration in a matter of a few weeks. Once the concept was proved, the more extensive task was undertaken of implementing the scheme on an actual target environment—OS/2 Presentation Manager (version 1.3). This work now surpasses the function of the original demonstration and is already providing some operability for useful applications.

The OS/2 Presentation Manager is a multitasking operating system that supports multiple concurrent processes and uses the 386 processor's built-in memory-protection scheme. Under OS/2, the blind interface exists as a separate application that runs simultaneously with the applications that the user wishes to run.

Approximately twenty times per second, the blind interface determines the identity of the graphical feature under the mouse pointer and then communicates with software running on the audio card processor to control the sound generation. The blind interface also sends data to the text-to-speech unit via the serial communications port.

At this point and without extensive investigation, the concept of mapping sounds to graphical features as presented in this work seems to be extensible to graphical environments other than OS/2 Presentation Manager, such as AIX Motif or DOS Windows. The implementation of the blind interface application under other operating systems, however, may require a different approach to obtain the screen information in a continuous real-time manner, or may even require modification of the operating system itself.[6]

The hardware for other systems should not be a problem, since sound-generating cards are available for all major personal computers, and most text-to-speech systems are controlled through the commonly existing serial port.

CONCLUSION

The main goal of this project is to enable the blind or visually impaired user to work with the increasingly pervasive GUI. Our work is unique in that it allows the user to experience the graphical aspects of the interface in addition to the textual information. A secondary goal is to create a workstation that can be readily shared with a sighted coworker. In fact, sighted users appreciate audible feedback as a multimedia enhancement of the GUI. Other potential applications for some of the concepts presented here include assisting those with cognitive disorders by providing multisensory reinforcement, enhancing the friendliness and appeal of computers to children and the elderly, and augmenting the accessibility of computer-driven information kiosks in public places.

Continuing efforts will be toward increasing the scope of graphical interactions that can be represented by sound; enhancing the quality, richness, and dimensionality of generated sounds; providing for user customization of sound generation; and enabling more productive use of color-to-sound mapping.

Some issues will continue as underlying pursuits in future work, such as finding the simplest correlations of sound with graphical meaning, which can then be standardized and applied widely. Another design issue is that of allowing the present scheme to coexist with and to complement text-based screen-interpreting aids (e.g., IBM ScreenReader/2), which may be preferable in some instances.

At this point, the blind interface has been used only in an attempt to provide access to standard windows and controls that form the bulk of application appearance. Graphically drawn entities such as line drawings, curves, and charts are not specifically handled in the current system. The proposed sound-generation scheme should at least be amenable to, and should not preclude, dealing with these items.[7] A further long-term goal is to allow a blind user to create graphical material and present it at meetings.

REFERENCES

[1]Drumm, A. D., "Audible Cursor Positioning and Pixel Status Identification Mechanism," *IBM Technical Disclosure Bulletin* **09-84**, 2528 (1984).

[2]Gaver, W. W., "The SonicFinder: An Interface That Uses Auditory Icons," *Human-Computer Interaction* **4**(1), 67-94 (1989).

[3]Lazzaro, J., "Windows of Vulnerability," *BYTE,* 416 (Jun 1991).

[4]Edwards, A. D. N., "Soundtrack: An Auditory Interface for Blind Users," *Human-Computer Interaction* **4**(1), 45-66 (1989).

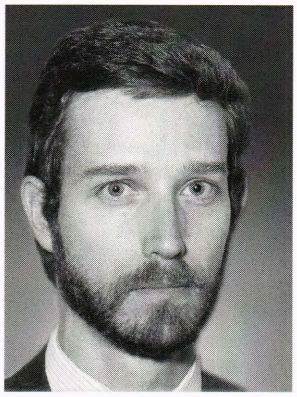
[5]Blattner, M. M., Sumikawa, D. A., and Greenberg, R. M., , "Earcons and Icons: Their Structure and Common Design Principles," *Human-Computer Interaction* **4**(1), 11-44 (1989).

[6]Schwerdtfeger, R. S., "Making the GUI Talk," *BYTE*, 118-128 (Dec 1991).

[7]Mostow, M. A., "Converter from Visual Curves to Auditory Cues for Blind Persons," *IBM Technical Disclosure Bulletin* **05-89**, 381-383 (1989).

ACKNOWLEDGMENTS: The author wishes to acknowledge the assistance and patience of Chris Bowman, a blind student attending the University of Texas at Arlington. Further acknowledgments go to IBM for supporting this work and those within IBM who have provided input and guidance. My immediate colleagues and management are particularly appreciated for their wholehearted encouragement of this pursuit.

THE AUTHOR



FRANK McKIEL, Jr., received his B.S. degree in chemistry from the University of Texas at Dallas in 1981. His career started at Rockwell International, where he served diverse roles as a microwave telecommunications technician, a chemist, and a microelectronics process engineer specializing in wire bonding and fiber optics technology. He obtained four patents during that employment and also taught undergraduate electronics as an adjunct instructor at a local college. In 1989, he joined an IBM division that makes office workstation software so as to apply his broad background and interests to developing neural networks, speech-recognition systems, signal processing algorithms, and advanced user interfaces. The audio-enabled graphical user interface is an offshoot of these endeavors.