BRINGING ICONS TO LIFE

Ronald Baecker,[1] *Ian Small,*[1,2] *and Richard Mander* [2,3]

1. Dynamic Graphics Project
Computer Systems Research Institute and Department of Computer Science, University of Toronto

2. Human Interface Group, Apple Computer, Inc.

3. Department of Educational Psychology, Stanford University

This research completely conducted at the Human Interface Group, Apple Computer, Inc.

ABSTRACT

Icons are used increasingly in interfaces because they are compact "universal" pictographic representations of computer functionality and processing. *Animated icons* can bring to life symbols representing complete applications or functions within an application, thereby clarifying their meaning, demonstrating their capabilities, and even explaining their method of use. To test this hypothesis, we carried out an iterative design of a set of animated painting icons that appear in the HyperCard tool palette. The design discipline restricted the animations to 10 to 20 second sequences of 22x20 pixel bit maps. User testing was carried out on two interfaces — one with the static icons, one with the animated icons. The results showed significant benefit from the animations in clarifying the purpose and functionality of the icons.

KEYWORDS

Icons, interaction techniques, dialogue methods, iterative design, empirical studies, HyperCard.

INTRODUCTION

WIMP (Windows, Icons, Menus, and Pointers) interfaces of the 80's have empowered millions of new computer users. Nonetheless, as systems become more sophisticated, and as functionality continues to expand, interfaces become more complex, sometimes arcane and even bizarre. They are often neither easy to learn nor easy to use.

New developments in interface technology can help deal with the problem of interface comprehensibility. One such development is non-speech audio (Buxton, 1989; Gaver, 1986, 1989). This paper deals with an example of another such development: the *animated interface* (Baecker and Small, 1990).

Why do we believe that animation could help make interfaces more understandable? The reason is that animation is an effective means of portraying complex processes evolving over time. This paper describes ten specific ways in which animation could assist a user. It then focuses on one specific kind of animation: the *animated icon.*

We define animated icons and discuss how animation might solve some of the problems associated with icons while preserving their traditional strengths. We then discuss the design of animated icons and describe our first attempt at a set of animations for the HyperCard tool icons. Weaknesses discovered through user testing led to a redesign which is also described. The second set of animated icons was then user tested in comparison with traditional static icons. The results of these tests are reported and interpreted.

USES OF ANIMATION AT THE INTERFACE

Ten basic ways in which animation could assist a user may be described in terms of questions about a system or interface that animation can help answer (Baecker and Small, 1990, 257-266):

• **Identification**:	What is this?
• **Transition**:	Where have I come from and gone to?
• **Orientation**:	Where am I?
• **Choice**:	What can I do now?
• **Demonstration**:	What can I do with this?
• **Explanation**:	How do I do this?
• **Feedback**:	What is happening?
• **History**:	What have I done?
• **Interpretation**:	Why did that happen?
• **Guidance**:	What should I do now?

For example, many home computer games display animated identifications while the system is loading. The outline zoom (an animated transition) which accompanies the opening (and closing) of an icon on many desktops orients the user to the location of the new window which appears on the desktop. The "Guided Tours" which are distributed with some Apple products are animated explanations. Two well known examples of animated feedback are the animated icons in the Sapphire windowing environment (Myers, 1984) and the now common *percent-done progress indicators* (Myers, 1985).

MOTIVATION

Myers used animated icons to provide feedback about the status of system-level functions and processes. We have begun to explore another kind of animated icon, one that brings to life the symbols representing applications (such as writing, painting, and drawing programs) and the functions within an application (such as the painting tools within HyperCard). Why might one want to do this? What are some of the strengths and weaknesses of icons?

Easterby (1970) stresses the advantages for international use of symbolic displays over those that are language-based. Lodding (1983) asserts that, because people find images "natural," because the human mind has powerful image memory and processing capabilities, because icons can be easily learned and recognized, and because images "can possess more universality than text," iconic interfaces can "reduce the learning curve in both time and effort, and facilitate user performance while reducing errors." Gittens (1986) notes the ease with which graphical attributes of icons such as style and colour can be used to represent common properties of a collection of objects.

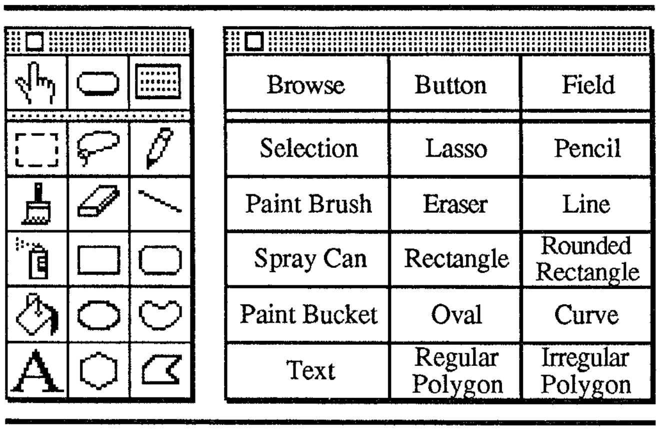
Manes (1985), on the other hand, asserts that icons may be confusing, wasteful of space, and totally ineffective in dealing with large numbers of similar commands, files, or concepts. Gittens (1986) notes the difficulty of finding "obvious pictographic equivalents" of computer system concepts, and of using icons to deal with the specification of large numbers of command parameters. Kolers (1969) notes that the claims for the "immediacy" and "directness" of the understanding of pictograms are exaggerated, and that recognizing even realistic icons requires "a great deal of perceptual learning, abstracting ability, and intelligence."

Despite this debate, there is little experimental evidence documenting the advantages and disadvantages of icons and specifying how they can be used appropriately and optimally (Baecker and Buxton, 1987). In terms of the questions presented above, icons typically serve as *identifications* of applications and functions, answering the "what is this?" question. Ideally, the meaning of an icon should be obvious to experienced users of a system, and also be evocative and self-evident to new users. Many icons fail to meet the former criterion; most fail to meet the latter criterion.

Our hypothesis was that animating icons would reduce this problem: animated icons should be easier to recall for experienced users, and more obvious in their meaning to new users. The work to be described tests this latter conjecture. It also tests the conjecture that animating icons can allow icons to provide a *demonstration* function as well, answering the "what can I do with this?" question.

ITERATIVE DESIGN

We chose the HyperCard tool palette as a suitable domain for testing our conjecture. Each of the 18 icons is a 22 pixel x 20 pixel bit map. The appearance, name, and location of each icon is shown in Figure 1.



Browse	Button	Field
Selection	Lasso	Pencil
Paint Brush	Eraser	Line
Spray Can	Rectangle	Rounded Rectangle
Paint Bucket	Oval	Curve
Text	Regular Polygon	Irregular Polygon

Figure 1: The HyperCard tool palette

Animation describing an icon could occupy an arbitrary portion of the screen. To keep the animation from occluding other icons or obscuring the user's context, we adopted a severe design discipline requiring that each icon should animate in place, within its 22x20 area.

An experienced animator, familiar with the Macintosh and its animation capabilities, was hired to create the prototype animations. The task was extremely laborious: we lacked tools suitable for designing, editing, and playing miniature animated bit maps. Yet the initial results were encouraging, suggesting that miniature animations could be effective. Some key frames from several prototype animations appear in Figure 2.

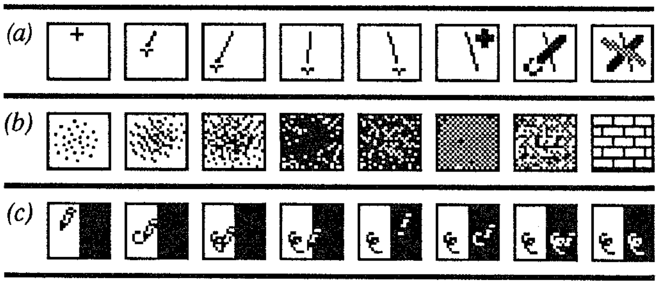


Figure 2: Keyframes from prototype animations (shown actual size). While it is difficult to get a sense of the animation from a static representation, imagine the animation flowing smoothly through the images shown.
(a) the Line tool
(b) the Spray Can tool
(c) the Pencil tool

We conducted a preliminary test by observing the reactions to the animated icons of five experienced HyperCard users from varying backgrounds. We also submitted the animations to detailed scrutiny, looking for cases in which the animations were:

- Wrong, showing something that was incorrect.
- Misleading, easy to misinterpret.
- Confusing, capable of multiple reasonable interpretations.
- Inconsistent, with respect to the other animations.

- Structured poorly, not presenting uses of a tool progressing from the most important and simplest to those less important and more complex.
- Incomplete, omitting something fundamental.
- Too long, trying to show too much.

Our scrutiny revealed several instances of animations that were misleading, either because of visual illusions or because the animator was trying to be too clever. These problems demonstrated that it was clearly important to keep the animations simple, both visually and conceptually. There were also a number of inconsistencies, some having to do with accurate renditions of a simulated cursor in the animations, and some relating to the order of presentation of a tool's capabilities.

Aside from identifying many of these problems, the five subjects were also concerned about the level of detail conveyed by the animations. Some expected complete *explanations*, believing the scope of the animations to be larger than it was. It was clear that this was not possible, as the goals of each animation were constrained by spatial and temporal considerations. In fact, the initial animations of the Browse, Button, and Field tools were clearly not successful in conveying the complex abstract concepts they represent.

We then planned a more detailed study. In order to simplify the subject requirements, and because of the difficulties in animating the Browse, Button, and Field tools in so little time and space, we opted to eliminate these tools from the palette (see Figure 3).

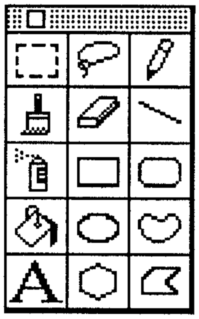


Figure 3: The test palette

Based on the feedback summarized above, we modified or completely redesigned eight paint tool animations (see Figure 4, which shows four of the original animations and their redesigns). The changes ranged from simple ones, such as adding a miniature cursor to the Rectangle tool animation, through more complex modifications, such as reordering the Oval tool animation to depict the simplest case first, to complete conceptual redesign, such as creating new Lasso and Selection tool animations in an attempt to emphasize the similarities and differences between the two.

We left untouched two animations, the Spray Can tool and the Regular Polygon tool, which probably could have been improved. Five painting animations were deemec satisfactory in their original form.

EMPIRICAL EVALUATION

The animated icons used in this study were designed tc represent the functionality of HyperCard's painting tools Our goal was to determine whether or not animations coulc help users understand this functionality.

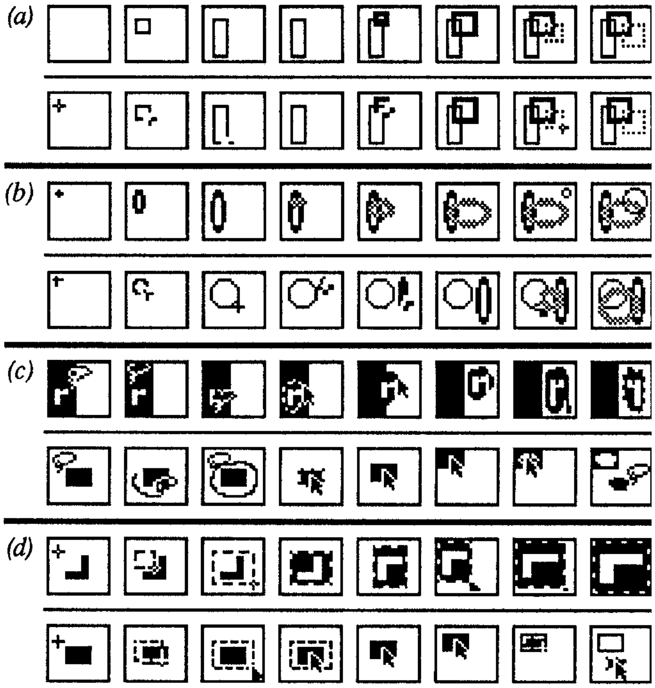


Figure 4: Four of the animations and their redesigns
(a) the Rectangle tool
(b) the Oval tool
(c) the Lasso tool
(d) the Selection tool

Since the animations were designed for novice painters, wє created a simple set of painting tasks in a HyperCard stack This stack consisted of 11 cards, each showing a picture or the left and a partially completed copy on the right, a demonstrated in Figure 5. The artwork was put in the background layer of the card so that the user could paint or the card and erase errors without affecting the preliminary material. The tasks were designed to be progressive beginning with one that required the Pencil tool, ther moving to more complex painting tools, and finally to the Lasso and Selection tools. However, the tasks were alsc designed to provide more than one way to complete the painting. Thus it would not be obvious to users which too to use.

Nine subjects, four having no experience with Macintost paint programs, four having some familiarity, and one having considerable experience with drawing programs, were selected for the study. The results from one novice were noi included in the data because the animations were different but the results were comparable to the others. All subjects completed the entire task set. Test sessions were monitored by two observers, and were audio and videotaped for later analysis.

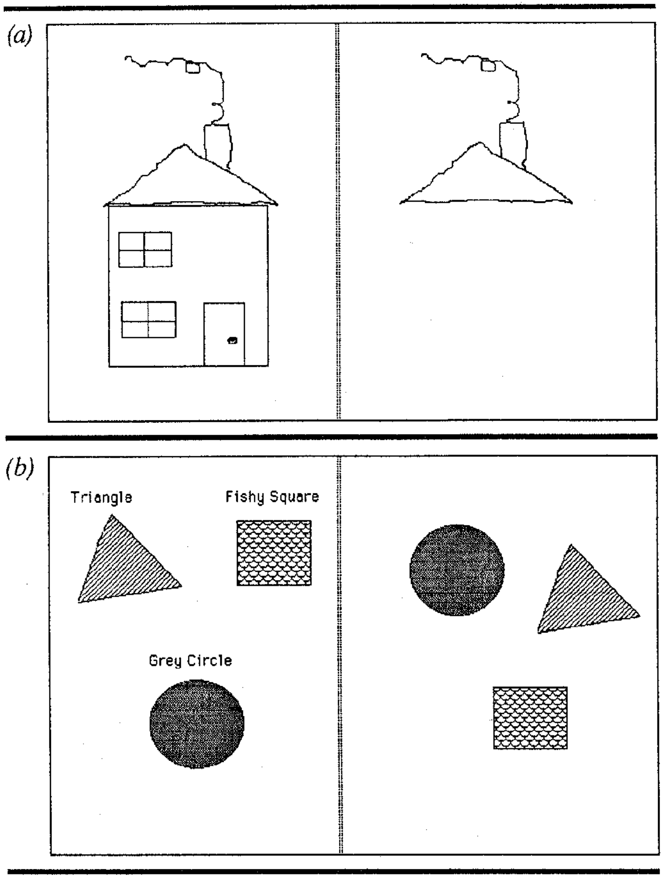


Figure 5: Two of the painting tasks in the user study

We structured the study to provide data describing user comprehension of tool functionality after viewing the static icons, after viewing the animated icons, and finally after using the tools. Before starting the painting tasks, each user was shown the static tool palette and asked to describe what each tool might do. We relied on intrusive questioning to provide this information during the test, questioning subjects after they had viewed a tool's animation but before they had used the tool itself. A final interview provided the required data after users had gained experience with the tools. This test procedure bears some similarity to the first part of the "Naming Test" of the "Icon Shape Test" described in Bewley et al. (1983).

Static Icons

Users' reactions to the Lasso, Pencil, Paint Brush, Eraser, Line, Spray Can, Rectangle, Rounded Rectangle, Oval, and Text tool icons indicate that these static images convey the functionality of the tools they represent. Novice users often found the remaining five tool icons confusing (see Table 1).

While most users thought that the Lasso and Selection tools were for selecting objects, the exact differences between their functionalities were not clear. Users not identifying the Selection tool correctly thought that it drew rectangles with broken lines. The Paint Bucket tool was not recognized as such by any of the novice users. They dubbed it the graduation cap; one user continued to refer to it as a graduation cap that tipped paint even after determining its functionality.

None of the subjects was able to identify the Curve tool properly, although two subjects thought it drew kidney beans. The Regular Polygon tool was often thought to make hexagons and the Irregular Polygon tool to make objects with parallel sides.

Animated Icons

All subjects liked the animations, reporting that they found them very useful. Novice users noticed the animations early in the testing session, typically during the first painting task, and would then look at the animations to find out what the tools did and to confirm their expectations of a tool's functionality. Table 1 demonstrates that in every case the users understood the purpose of each icon after viewing the animations.

One user commented that the animations made selecting an icon less like a memory game in which you had to remember the icon's purpose, since you could view its animation any time as a reminder.

User experiences demonstrated that every detail of an animation needed to be clear and not subject to misinterpretation, which could arise from the strangest of coincidences. Searching for an appropriate tool for adding a chimney to a house, one subject noticed that the Paint Bucket tool's animation featured a chimney-like shape (see Figure 6). Selecting and using the chimney tool resulted in a completely black painting and a very disgruntled user, rather than the elegant chimney desired. Experiences such as this demonstrate the difficulties of creating effective animation which is sufficiently generic and abstract to avoid confusion.

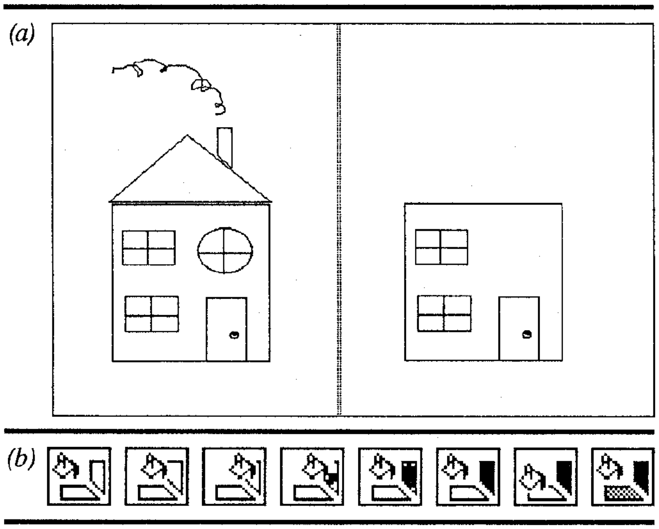


Figure 6: How unexpected confusion can occur
(a) the chimney task
(b) the Paint Bucket tool animation

	Novice (3)		Familiar (4)		Expert (1)	
	Static	Anim.	Static	Anim.	Static	Anim.
⬚	1	All	All	All	All	All
➰	All	All	All	All	All	All
✎	All	All	All	All	All	All
🖌	All	All	All	All	All	All
⌫	All	All	3	All	All	All
╲	2	All	All	All	All	All
🧴	All	All	All	All	All	All
□	2	All	All	All	All	All
▢	2	All	All	All	All	All
🪣	0	All	1	All	All	All
⬭	2	All	All	All	All	All
♡	0	All	1	All	0	All
A	All	All	All	All	All	All
⬡	0	All	0	All	All	All
⏢	1	All	1	All	All	All

Table 1: User comprehension of static and animated icons. The number of users who understood the static icons, and their animated counterparts, is indicated. For example, only 1 novice user understood the static Selection tool, but all 3 understood the animated version.

User context and expectation also affect interpretation of the animations, and must be taken into account during the design process. The Spray Can tool animation depicts several different patterns being sprayed, culminating in the brick pattern. Some users interpreted this to mean that the tool could spray on walls, rather than being able to spray a wall-like pattern.

Similarly, some users interpreted the Selection and Lasso tool animations as meaning that the tools selected objects, rather than regions. While the animations were designed to emphasize the subtle differences between the two tools, users tended to miss details, such as the shrinking nature of Lasso selection, which conflicted with their expectations.

Animated Icons With Sound

A number of the animations, such as the Rectangle tool, conveyed usage as well as functionality. Others, such as the Irregular Polygon tool, were less successful. We believe that this extension to animated explanation would be greatly assisted by the addition of sound effects such as mouse clicks. Multiple media such as animation and sound can be combined to produce compelling help and assistance systems. For example, in the case of the Irregular Polygon tool animation, novice users did not understand how to end a polygon. Adding the sound of mouse clicks to the animation, including the double click required to end the polygon, should clarify this. While we have experimented with prototype implementations that make use of sound, we have not yet subjected them to testing.

INTERFACE ISSUES

While our goal for this study was to address the macro-interface issue of whether animated tool icons provided assistance to novice users, the session recordings enabled us to identify some micro-interface issues as well. Although we had predicted some of the micro-interface behaviour that we observed, other aspects were surprising to us.

We considered allowing all animations to play simultaneously, but rejected this technique as "too busy." After experimenting with a number of methods for accessing the animations, we provided our subjects with a palette that was configured to trigger the animations whenever the mouse cursor passed into a new icon's bounding box. Upon triggering an animation, most users would try to move the mouse out of the way in order to get a clear view of the animation, only to have the animation stop when the mouse was moved outside the icon's bounding box. After a few false starts, users became adept at keeping their mouse just on the edge of the bounding box to maximize their view of the animation. This training process occurred much more rapidly than anticipated.

Two of the more experienced users selected tools so quickly that they completed the painting tasks without ever seeing the animations. This was possible because of a short delay between entering an icon and the noticeable start of its animation. These users liked this feature because of the control it gave them. Unfortunately, some novice users initially found the dissolve associated with this delay to be confusing because it made them think that the tool was unavailable for their particular task.

Users were frequently confused between selecting a tool and animating it. This confusion manifested itself in user behaviour which involved watching the animation of a tool which was not the currently selected tool, and then immediately attempting to use that tool without first selecting it.

Animations which depicted sequences of different tool modes sometimes confused users by making them think that they could select the different modes by carefully timing their selection click with respect to the evolving animation. While this may be a valuable interface technique, animations which caused such user reactions were clearly in need of redesign.

SUMMARY AND CONCLUSIONS

The users found animated icons useful and helpful. In every case in which static icons were not understood, the dynamic icons were successful in conveying the purpose of the tool. In some cases, changes or simplifications to the animations are required to depict tool functionality more accurately.

While the results of this study are positive, there are many issues which are still outstanding. From an authoring standpoint, there are few clear rules regarding length, content, ordering, and visual representation which will guarantee effective animations, particularly for tools with more abstract functionality. Even iterative design can fail to achieve success, as shown by some of our results. Furthermore, the lack of tools for designing miniature bit mapped animations makes prototyping more laborious than it should be.

From an interface standpoint, we have not yet accomplished a completely satisfactory accessing scheme. Neither have we determined how animated icons relate and fit into the context of a much larger and extensive help system. Both these issues are extremely important.

Finally, we need to determine whether we can provide animated icons which can be helpful with more complex functionality and to all levels of users. A different style of animation may be needed to demonstrate effectively complex abstract concepts such as those represented by the Browse, Button, and Field icons. Also, by providing different animations for novice, familiar and expert users, we may be able to extend this help technique to a more general application.

APPENDIX: A NOTE ON THE IMPLEMENTATION

We conducted preliminary evaluations using a prototype implementation which displayed the animations in isolation from the surrounding HyperCard environment. While sufficient for obtaining initial feedback, this implementation lacked the functionality required for supporting novice users attempting real tasks.

A more complete implementation was prepared for the second round of user testing. We replaced the standard HyperCard tool palette with our own abbreviated animated tool palette, thereby allowing users to conduct all their interaction through the new experimental interface. The animations were displayed using a low-level animation playback system developed earlier during our research. The low-level engine ensured that animations played at a constant speed despite system load, by dropping frames from the display list as required. Sound was supported either as a single continuous sound track attached to the animation, or as sound events attached to individual frames within the animation.

ACKNOWLEDGEMENTS

This work was conducted in the Human Interface Group of the Advanced Technology Group of Apple Computer, Inc. We are indebted to Eric Martin, who created the initial animation prototypes; Ruth Ritter, who helped to redesign animations based on user feedback; and to Anne Nicol, Kate Gomoll, Shifteh Karimi, and S. Joy Mountford, for assistance including user testing, animation redesign, research support, and comments on this paper.

REFERENCES

Baecker, R. and Buxton, W. (1987). *Readings in Human-Computer Interaction: A Multidisciplinary Approach.* Morgan Kaufmann.

Baecker, R. and Small, I. (1990). Animation at the Interface. In Laurel, B. (Ed.), *The Art of Human-Computer Interface Design*, Addison-Wesley, 251-267.

Bewley, W., Roberts, T., Schroit, D., and Verplank, W. (1983). Human Factors Testing in the Design of Xerox's 8010 'Star' Office Workstation, *Proceedings of CHI '83*, 72-77.

Buxton, W. (1989). Introduction to this Special Issue on Non-Speech Audio. *Human-Computer Interaction* 4(1).

Easterby, R. (1970). The Perception of Symbols for Machine Displays. *Ergonomics* 13(1), 149-158.

Gaver, W. (1986). Auditory Icons: Using Sound in Computer Interfaces. *Human-Computer Interaction* 2(2), 167-177.

Gaver, W. (1989). The Sonic Finder: An Interface that Uses Auditory Icons. *Human-Computer Interaction* 4(1).

Gittens, D. (1986). Icon-Based Human-Computer Interaction. *International Journal of Man-Machine Studies* 24, 519-543.

Kolers, P. (1969). Some Formal Characteristics of Pictograms. *American Scientist* 57(3), 348-363.

Lodding, K. (1983). Iconic Interfacing. *IEEE Computer Graphics and Applications* 3(2), March/April 1983, 11-20.

Manes, S. (1985). Pushing Picture-Perfect Programs: Smash That Icon! *PC Magazine*, June 1985, 64.

Myers, B. (1984). The User Interface for Sapphire. *IEEE Computer Graphics and Applications* 4(12), 13-23.

Myers, B. (1985). The Importance of Percent-Done Progress Indicators for Computer-Human Interfaces. *Proceedings of CHI '85*, 11-17.