Henry Lieberman

Intelligent Graphics

What's a new paradigm?

A NEW paradigm is not just something that's a good idea. There are plenty of merely good ideas, but a new paradigm must go beyond simple innovation. A new paradigm is often introduced to solve a particular problem, but it must do more than that. It must fundamentally change the way we look at problems we have seen in the past. It must give us a new framework for thinking about problems in the future. It changes our priorities and values, changes our ideas about what to pay attention to and what to consider important.

Thomas Kuhn wrote most eloquently about the impact of new paradigms in science in *The Structure of Scientific Revolutions* [7]. He traced the history of new paradigms such as quantum mechanics in physics. Such new paradigms are introduced in response to perceived problems with current practice, are advocated by enthusiastic iconoclasts, usually meet entrenched opposition, are accepted only slowly, and eventually become standard practice themselves.

How does a new paradigm come about? Of course, the appearance of new paradigms is inherently unpredictable, but one thing that can be said is that the synergy between seemingly different disciplines is often a fertile ground for the growth of new paradigms. Here, I don't mean the kind of shallow interdisciplinary activity in which an expert in one field blithely assumes he or she can make pronouncements in another. Rather, when experts in different fields look with admiration and curiosity at each other's domains, and search for commonalities and fresh perspectives on their own, truly new paradigms can result.

A New Paradigm of AI and Graphics

In my career, I've always felt caught between the fields of artificial intelligence (AI), beginning my career at the MIT AI Lab, and computer graphics, or more generally, human-computer interaction (HCI). I've always strongly felt that the goal of computing should be to enable collaborative problem-solving between people and machines, and that to place all the emphasis on the intelligence of the machine (as AI does) or all the emphasis on interfaces that couple the computer to the human's intelligence (as HCI does) would only be part of the picture. A new paradigm, what could perhaps be called *intelligent graphics*, is needed.

Certainly, many events in the history of computer science point toward this synthesis. As far back as 1962, Ivan Sutherland's Sketchpad [16], introduced both the precursors of today's interactive graphical editors and CAD systems, and the AI technique of constraint networks. Early animation systems, such as those of Ron Baecker [1], Craig Reynolds, and others, tried to move animation beyond successions of static pictures to modeling underlying processes and characters, and offering animators assistance with the semantics of their imagined worlds. David Canfield Smith's Pygmalion brought learning-by-example techniques to an interactive visual programming language. As graphical computing became more widespread, some forward-looking graphic designers such as Aaron Marcus [13] helped educate the computer industry that visual design could play the role of a dynamic knowledge representation language for human-computer communication. Artists like Harold Cohen [14] and Myron Krueger [6] welcomed the participation of semi-autonomous programs in the creation of their art.

For me personally, the development of a new paradigm of intelligent graphics was strongly influenced by my collaboration with graphic designer, Muriel Cooper, and her research group, the Visible Language Workshop at the MIT Media Lab. Sadly, Muriel died in 1994, but the work of the group continues. Though Muriel was not a technical person, nor I a professional graphic designer, we shared a common vision for a new paradigm of intelligent graphics. In the remainder of this article, I will explain a bit about this collaboration and some of the research of the group that contributed to a synthesis between artificial intelligence and graphics.

A Pioneer in Intelligent Graphics

For many graphic designers and artists, the idea of computers entering the design process was viewed as threatening. The exploration of visual forms and sense of communication in visual design was viewed as essentially and uniquely human, and the incursion of the computer into the process was viewed by many as demeaning the role of the creative professional. Of all the branches of computer science that proposed relevance to visual design and communication, artificial intelligence was perhaps the most threatening of them all, since it proposed to model aesthetic judgment, which, in a pessimistic view, might leave little role for a human designer.

A SMALL handful of visual designers, however, saw it very differently. A lot had to do with the designer's personal view of the design process itself. The prevailing culture often viewed design as magic, designers as magicians. But some viewed the design process as a constant search for the relation between form and meaning, a continuing exploration of possibilities for expression. They welcomed tools that could assist a designer in relating form and meaning, in exploring and evaluating alternatives. They saw the possibility for the computer to become an intelligent partner in the design process, to extend the reach and grasp of the human designer. From that point of view, the concerns of artificial intelligence in representing meaning, inference, and learning are not foreign to the design process. They are central.

And it was not always the most technologically oriented among designers that had the greatest enthusiasm for the potentials of the new technology. It was the vision of new, dynamic, intelligent, image-making tools that drove a few designers, educated in a culture that was artistic, not technological, to put up with learning the often frustrating details of computers, then as now.

Muriel Cooper was one of those rare pioneers, one who had a vision of computer-based tools that could actively participate in the process of dynamic visual expression. She saw no inherent boundary between the visual design process and the software design process, only one imposed on us by the imperfections of our current technology. She was often bold and iconoclastic, challenging the unstated assumptions of both visual designers and computer scientists.

In Muriel's view, the computer is a new design medium, a new kind of brush, a new kind of canvas, a new kind of palette. There is no more reason for a designer to fear computers than for any artist to fear their tools. You learn to use them and to work creatively within their strengths and limitations. The particular strength of the computer as a design tool is its ability to respond dynamically to the user, to remove the burden of tedious, precise or repetitive operations, to become personalized to its users' needs and styles of working. Muriel was attracted by Nicholas Negroponte's original vision of the Architecture machine as an intelligent assistant for architectural design, and envisioned a similar role for an intelligent assistant for visual design tasks.

The Visible Language Workshop

Muriel's concern for supporting the design process with intelligent tools led her to make artificial intelligence one of the major themes of her research group, the Visible Language Workshop (VLW) [3]. Certainly, the AI community paid little attention to visual design problems during the 1970s and 1980s. It preferred to concentrate its attention on applications in such areas as engineering problem solving and medicine, which were felt to be "more well understood" and have a more immediate real-world payoff. The computer graphics community looked mainly toward involvement of the computer in image-production efforts and not toward the computer as a tool for conceiving new designs.

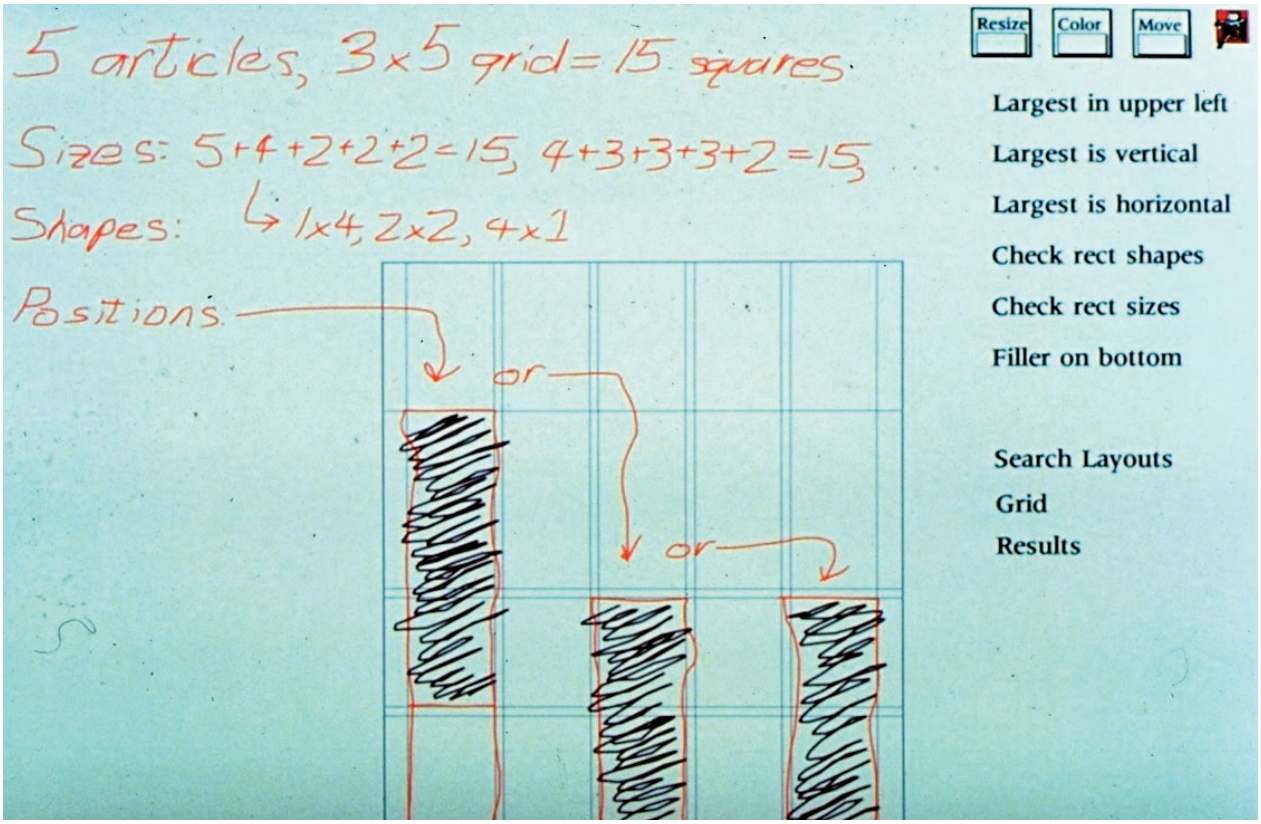
Muriel made the Visible Language Workshop a meeting ground for people from diverse backgrounds who shared a fascination for the computer as a place where technology and design could converge. Her collaborator in starting the VLW was Ron MacNeil, who came from a photography and printing background, and who also became a strong proponent of the AI approach to design. Designers who had little familiarity with the technology encountered computer scientists who hadn't yet given serious thought to problems of visual design and expression, and both sides learned enormously from one another. The VLW embodied the ideal of interdisciplinary spirit in which the Media Laboratory was founded.

A Personal Note

I had started out at the MIT Artificial Intelligence Laboratory with Seymour Papert's Logo group, which had the then-radical notion of using ideas and computer systems developed for artificial intelligence work to teach children about problem solving. The most popular activity for the kids was the turtle (first a robot, then an imaginary agent on a graphics screen) that could be commanded to draw pictures. My job was to program the computer graphics, and I

Figure 1. Generating design alternatives

Figure 2. Layout from sketches



had written the first Logo system for then-new raster graphics screens, and the first color Logo system. I was struck by how engaging the computer graphics domain was for the kids and by the power of combining both visual and abstract problem solving. It was this spirit of interdisciplinary collaboration that brought me to the Visible Language Workshop.

The kids programmed math and physical simulations, but art was also a major activity. I had exhibited some computer art at the SIGGRAPH conference, and some of my students and I won prizes in *Byte* magazine's first computer-art contest. My first official VLW event in the early 1980s was when I was invited by Muriel to participate in a panel on computer art, along with Russell Kirsch, who constructed AI simulations of famous artists' styles [5]. Russell, in his sweet and gentle manner, humbled me about my naiveté about artistic traditions and concerns. There was something exciting about the interplay between artistic and computational concerns that I hadn't encountered elsewhere. The panel was part of Muriel's annual summer course, *The New Graphics*, which introduced people from a variety of backgrounds to computer image making. The activities in this course represented some of Muriel's pioneering efforts in the field of what is now called electronic publishing. The course used the sometimes cranky, sometimes miraculous VLW image editing system, called Sys. The encounter between people of design and computer backgrounds was, for me, exhilarating.

Continuing my contact with Muriel and Ron through the incorporation of the VLW of the Media Lab, I started to teach in the introductory VLW course, and participate in thesis committees for VLW students. The course bore the rather nondescript title, *Computer Graphics Workshop*, but it was much more than that. Taken mostly by students immediately after joining the VLW, it was "boot camp" in the VLW philosophy. While much of the course was conducted in the manner of a traditional design course, with design exercises, projects and critiques, Muriel also wanted the students to gain some background and experience in AI issues, and I supplied that perspective, along with Ron and Patrick Purcell, who had done important work in studies of design process in architecture. I worked part-time between the Media Lab and the AI Lab for a while; then, feeling increasingly that the exciting directions for the future lay in the convergence of artificial intelligence with interactive interface design issues, I joined the VLW full-time in 1987.

Threads of VLW Research

AI techniques played a central role in many areas of the VLW's research. We'll explore some of these threads in the remainder of this article, surveying some student and staff projects that address these

themes. Among the themes are automatic layout, intelligent assistance for the construction of visual designs, learning by example, and visual representation of programs and of knowledge.

Unfortunately, much of the research performed by the students, especially in the early days, appeared only in unpublished Masters' theses. Professional standards in the visual design community place less emphasis on scholarly publication than in the computer science community. Where no explicit references appear to the work cited, interested readers can obtain the original theses by writing to the MIT Media Lab, and soon through our Web server, www.media.mit.edu.

Automatic Layout

Even in the days of mechanical typesetting, automatic letter spacing was accomplished mechanically, using wedges called "shivs" that pressed between the metal letters. In this age of electronic typesetting, no matter how good electronic tools for image editing, image composition, and screen design for interactive interfaces get, it is unreasonable to expect that a human designer's participation would be required for each and every screen that the user sees, or each and every page of hard copy. Digital data simply comes too fast, and is too dynamic to have every visual presentation designed by hand. An electronic news service cannot have a human hand-design each screen when stories are coming in minute-by-minute. Some sort of automatic layout is required.

Automatic layout seemed like a good candidate for an expert systems approach. Figure 1 shows a set of automatically generated design alternatives for a magazine cover, which were actually used for an issue of the magazine *IBM Perspectives*. The expert system took into account notions of "visual balance" among the small photographs arranged on a grid. Students Timothy Shea, Tom Amari and Alka Badshah were early explorers of rule-based systems for design, with business graphics and packaging being domains of application. Russell Greenlee used sketches of principal elements of layout as a guide, and his system generated sequences of possible layouts for the user to choose from that were consistent with expressed constraints. Figure 2 shows how the user inputs a "back of the napkin" sketch of positions and text flow for articles, and selects constraints from a menu on the right. I designed an automatic layout system that used best-first search to deal with the common problem of overconstrained space allocation. More recently, Grace Colby [2] applied the techniques of case-based reasoning and constraints to the automatic layout problem. B.C. Krishna looked at the problem of automatically detecting layout constraints from scanned images. Louis Weitzman [18] pursued a linguistic grammar-based approach, with the aid of a parser that can enforce geometric and temporal constraints during automatic layout. Figure 3 shows a table of contents automatically converted between two differ-

Figure 3. Automatic conversion of layouts

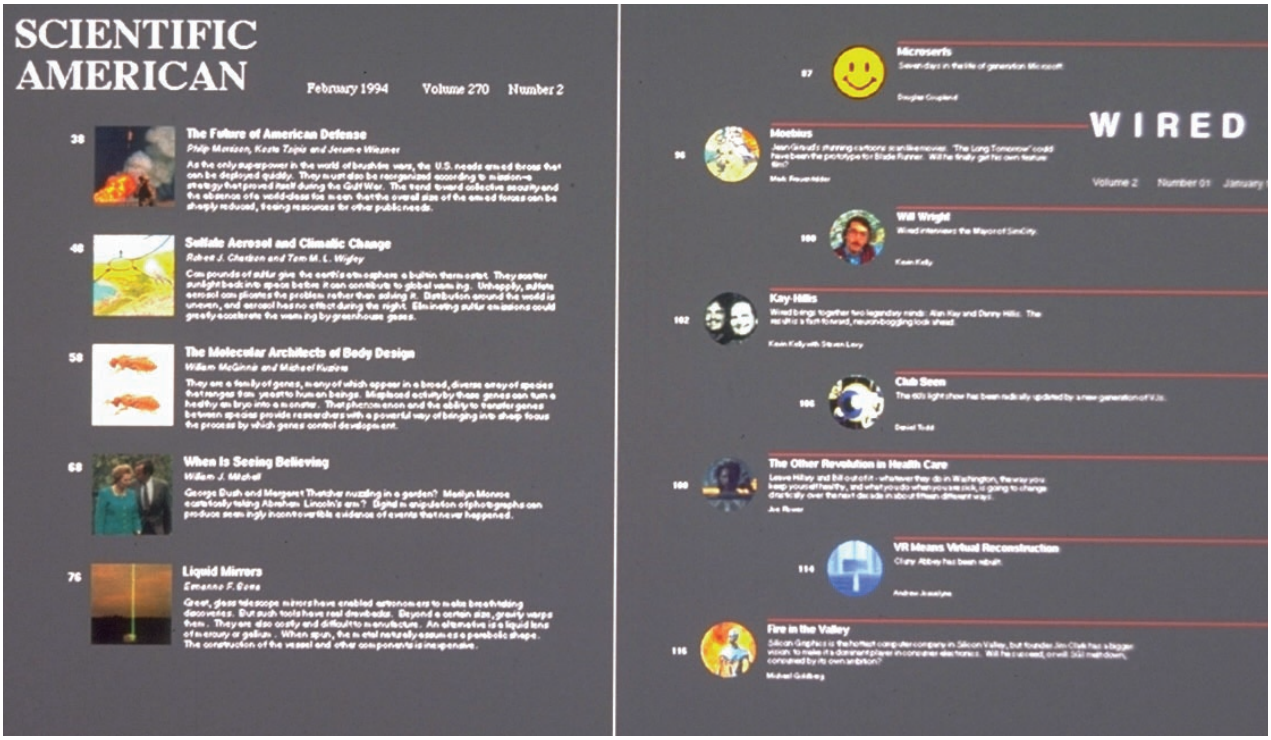
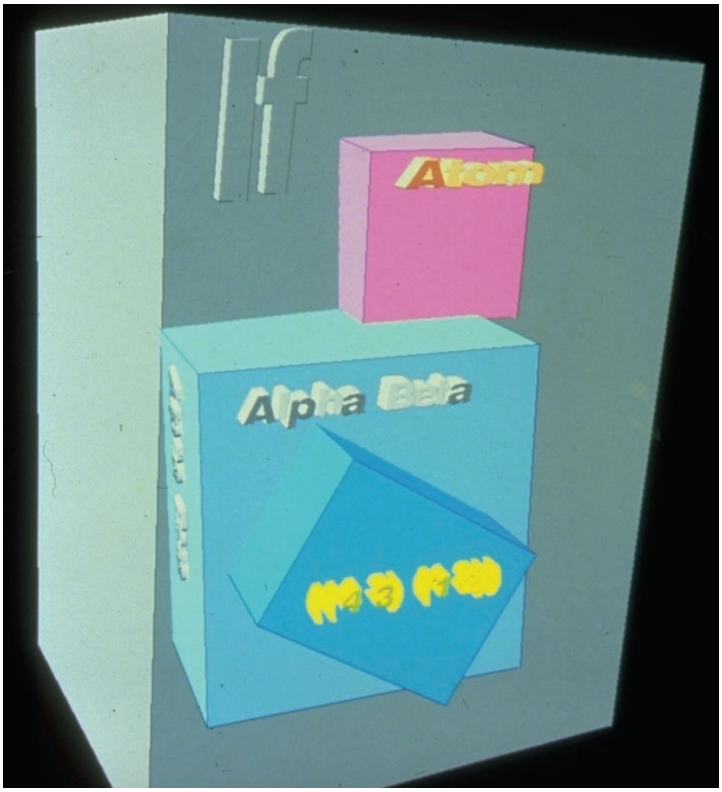


Figure 4. A 3D program representation



ent magazine styles: one for *Scientific American*, the other for *Wired* magazine.

Visual Representation of Knowledge

Applying concepts from AI programming to visual design problems such as layout was not the only concern of the VLW. Equally important was going in the other direction: applying visual design ideas to the process of constructing AI programs.

Muriel expressed constant frustration at the lack of visual imagery in the programming process itself, even when the subjects of the programs themselves were computer graphic images, or when representing knowledge about visual design. Conventional textual programming languages and environments put up a barrier for people who thought of themselves as visual thinkers against expressing themselves in the computational medium. So the VLW also pursued some work in visual programming and visual representation of knowledge in AI.

I led a project in investigating alternative visual representations, including 3D programming languages [8] and use of color and translucency. Figure 4 shows a three-dimensional representation of a computer program using nested boxes, which are animated as the program runs. Ron MacNeil's early Tyro system used a representation of visual design elements connected by "spider webs" representing constraints. Projects by students Dorothy Shamonsky, Ming Chen, and Michelle Fineblum also explored issues of visual representation in programming. Storyboards containing visual examples of states of a program appear in Fineblum's work, my Mondrian system and in Louis Weitzman's VIA. Visualization of rules, constraints and graphic relationships continue to play an important role in several VLW projects.

AI tools for interactive design AI techniques can be used to provide intelligent assistance for visual designers in the process of constructing both static images and interactive media. Designers who use interactive image editors and multimedia editors can benefit from having tools that explicitly support the design problem-solving process. This involves building some representation and understanding of the design process into the computational tools.

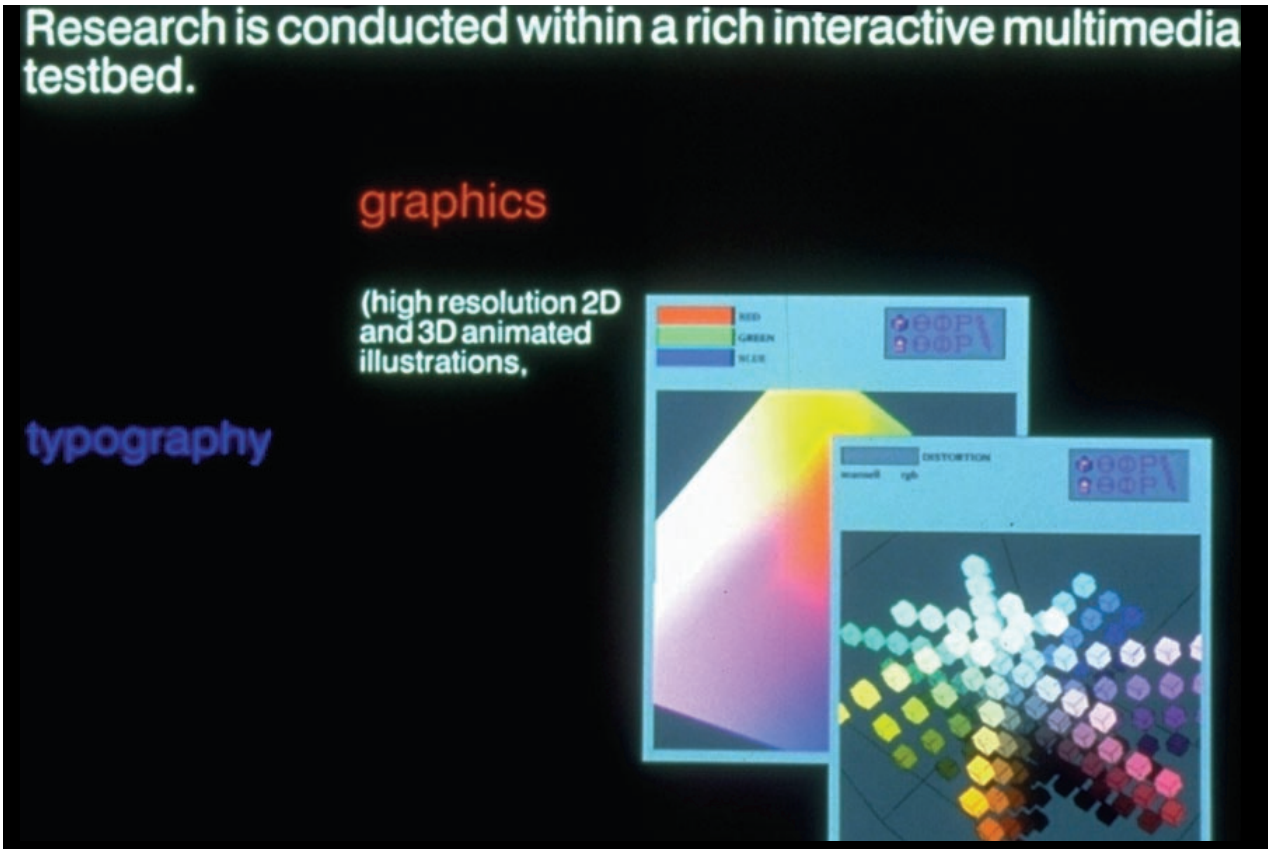
Muriel provided the insight of a top design professional into the process of design. A task for me and for her VLW collaborators was to "knowledge engineer" some of Muriel's design expertise, and the design expertise she brought to VLW in the form of visitors from the design world and reference materials. The knowledge engineering was trying to understand enough about the design process to understand what aspects of it could be embodied in intelligent interactive tools. We still have not come very far, relative to the creative powers of an expert designer, but the dream of intelligent design tools is on its way in many forms.

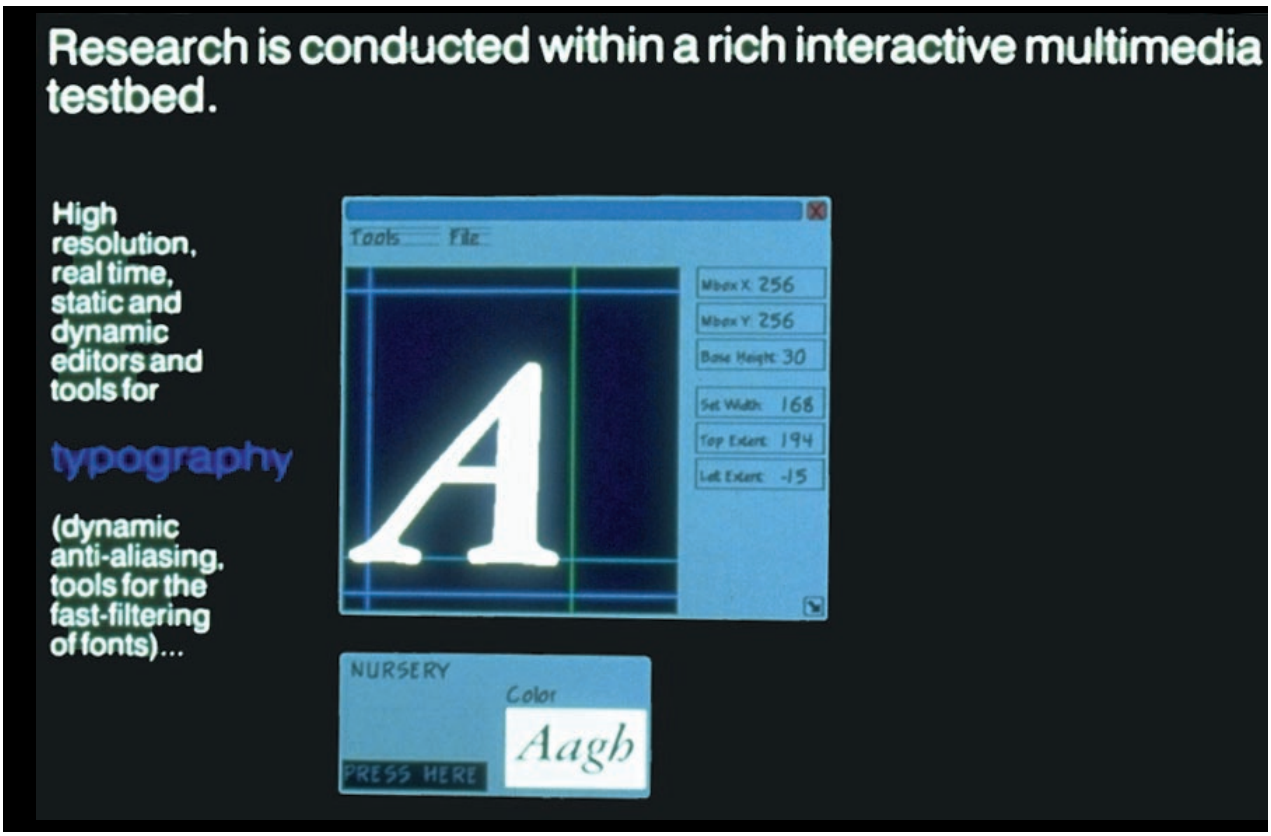
Debra Adams used the observation that type designers often base their designs on a few prototypical letterforms to construct a system that would automatically design a complete font after having been shown designs for the prototype letters. Craig Kanarick built knowledge about the design of charts and spatial data into a case-based presentation tool. Laura Robin built a best-first search engine into a hypermedia browser, so that the time and level of detail of a presentation could be automatically adjusted to produce the best possible presentation in a given amount of time and subject to expressed interest in a subset of the topics. Figure 5 presents two views of a single adaptive multimedia presentation, which show different levels of detail according to the user's interest and time limitations.

Learning from Visual Examples

A particular interest of mine in both artificial intelligence work and in studying the design process has been learning from examples. As a teacher, I have always been firmly convinced that the best way to learn is by example, and as a learner, my personal style has always been to concentrate on learning in

Figure 5. Two views of an adaptive presentation





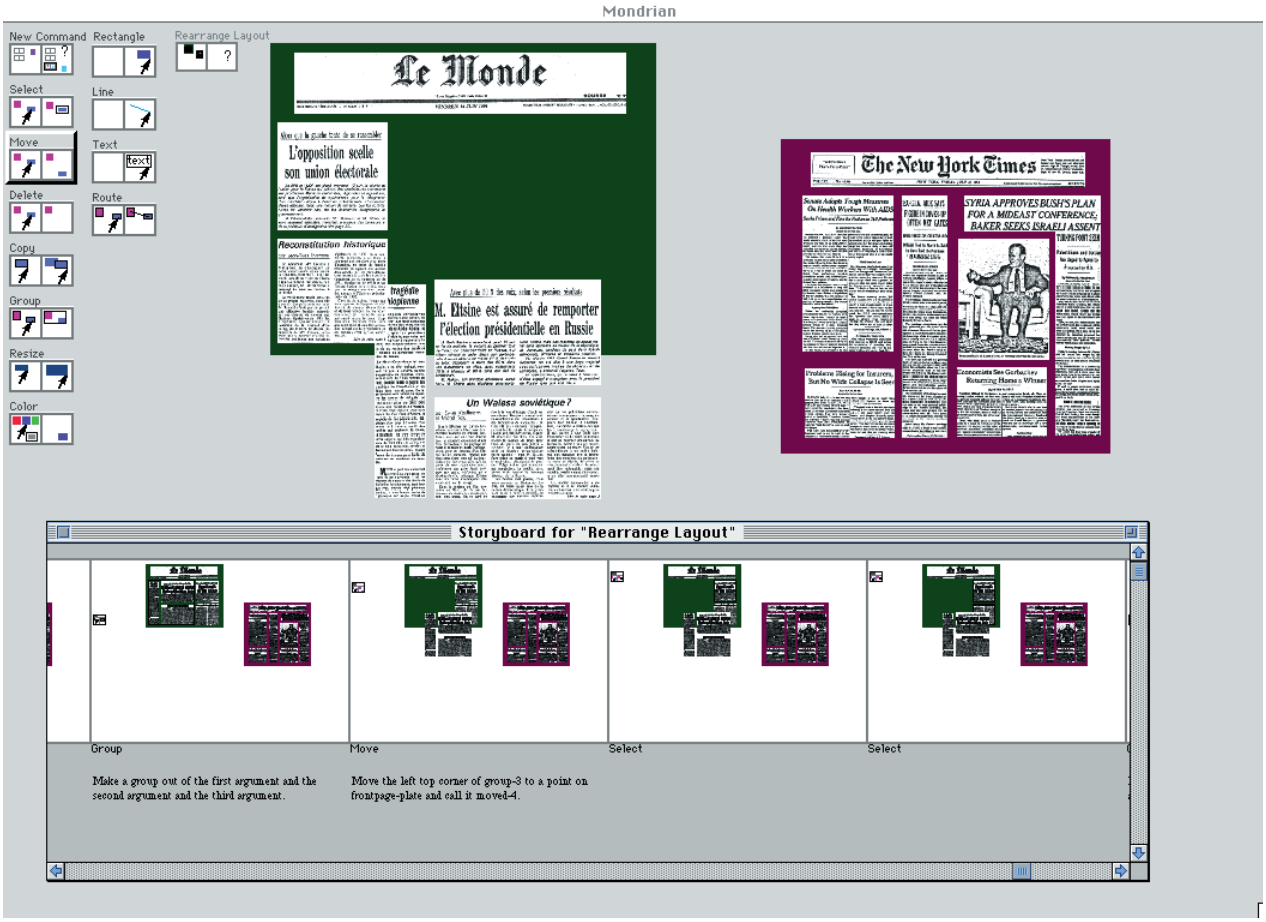


Figure 6. Learning from examples

this way. One of the most important lessons about visual design and communication that I learned from Muriel was the extent to which design is based on the generation and critique of concrete visual examples.

Open any book on graphic design, either one intended for teaching beginning design or for communication between seasoned professionals. Unlike books in science and engineering, few principles will be explicitly stated in the text; there will be no equations, no axioms. So how can design students use these books to effectively solve problems? What the books will provide is an abundance of examples: examples of exemplary designs, examples of design variants (or "near misses" in AI terminology) [11]. Good design students are experts in generalizing examples and making analogies from examples to new problems they encounter.

But why can't we use concrete visual examples in teaching our computers how to do designs, instead of incomprehensible programming languages and rule languages? It is this question that led me to work on the idea of programming by example. In this approach, a designer can use an interactive interface, and a software agent records the actions performed by the user, and can generalize them so that they can be applied to analogous problems in the future.

I built several systems that incorporated this approach. Mondrian [10] is an object-oriented graphical editor that can learn new graphical procedures by example. Its base is a MacDraw-like graphical editor, a familiar tool for design professionals. Its learning agent could also be applied to many other sorts of image editing and media editing tools. Mondrian incorporates a learning agent that records and generalizes procedures presented as sequences of interface commands. The user selects objects to represent the examples, and demonstrates a procedure which depends on those objects. The user interface is extended with a new operation that can be applied to different objects in the future. In Figure 6, we teach the computer how to rearrange articles on the front page of a newspaper, using a specific example from the newspaper *Le Monde*. On the right, the machine performs an analogous rearrangement on a layout from the *New York Times*.

Mondrian uses a consistent visual language for communicating with the user. Operations are represented with dominoes, before-and-after pictures representing a visual example of the operation, and storyboards, a comic-strip like sequence of frames. It

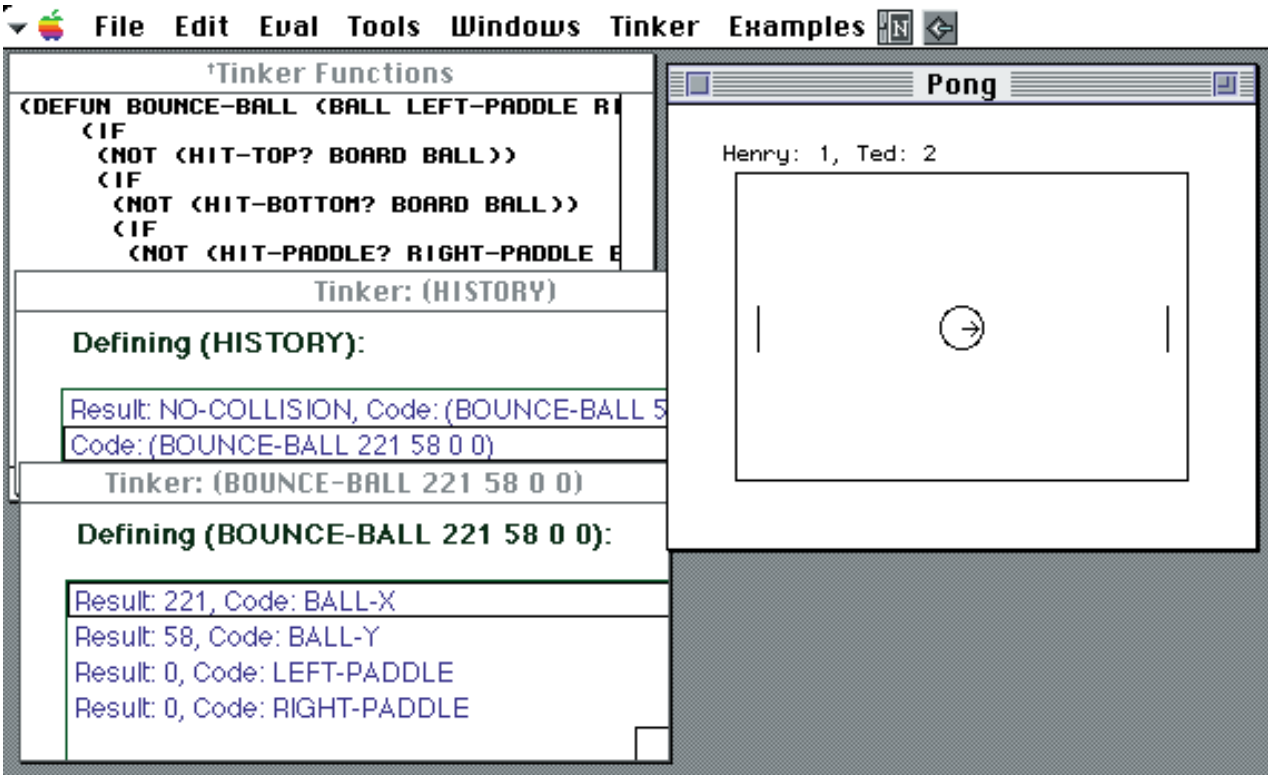


Figure 7. Programming a Pong game by example

automatically produces natural language descriptions, both written and spoken, and accepts advice from the user via speech recognition. Mondrian can also use digitized video as a source for the demonstration, to capture real-world procedures such as assembly and disassembly tasks.

An outstanding question about programming systems by example has always been "How general are they? What are their limits?". Many have expressed skepticism that programming by example won't work universally, and will always be limited to specialized domains. My system Tinker [9], was motivated in part by trying to answer this question by implementing a general-purpose programming system, capable of producing any program expressible in its underlying programming language, Lisp. It was the first to handle conditionals and recursion through the presentation of multiple examples. In one demonstration, shown in Figure 7, I showed how Tinker could be used to develop the video game Pong, by showing examples of configurations of the game, and demonstrating how the rules were to be applied by example. Thus, example-based techniques could be used to demonstrate dynamic graphical behavior. In this section, see the article by Ken Kahn [4], which takes the radical step of using dynamic video-game actions to represent the program itself.

Suguru Ishizaki continued exploring the theme of programming by example in a dynamic graphic environment by specifying temporal behavior in a geographic information system. In Figure 8, graphical presentation of dynamic icons showing animal migrations are automatically linked to the underlying map data as their locations change.

Sketched examples are natural input for design. They played an important role in Russell Greenlee's layout-from-sketch system, and some of the other systems mentioned earlier. Steve Librande wrote a system that automatically interpolated between sketched examples, based on Tomaso Poggio's radial basis functions, motivated by properties of the human visual system. Shown in Figure 9, the system can automatically produce drawings intermediate between a sketched face and profile, even without a 3D model. Case-based reasoning, another example oriented technique, was central to Ron MacNeil's Tyro [12], which constructed new multimedia presentations based on previously presented examples. In Figure 10, frames from videotaped examples illustrate the steps of a repair procedure for a circuit board, which can then be used to describe analogous procedures performed on different examples.

Intelligent Graphics as a New Paradigm

What makes the synergy between artificial intelligence and graphics a new paradigm, rather than just a collection of techniques, is that it fundamentally changes the way we view interaction between people

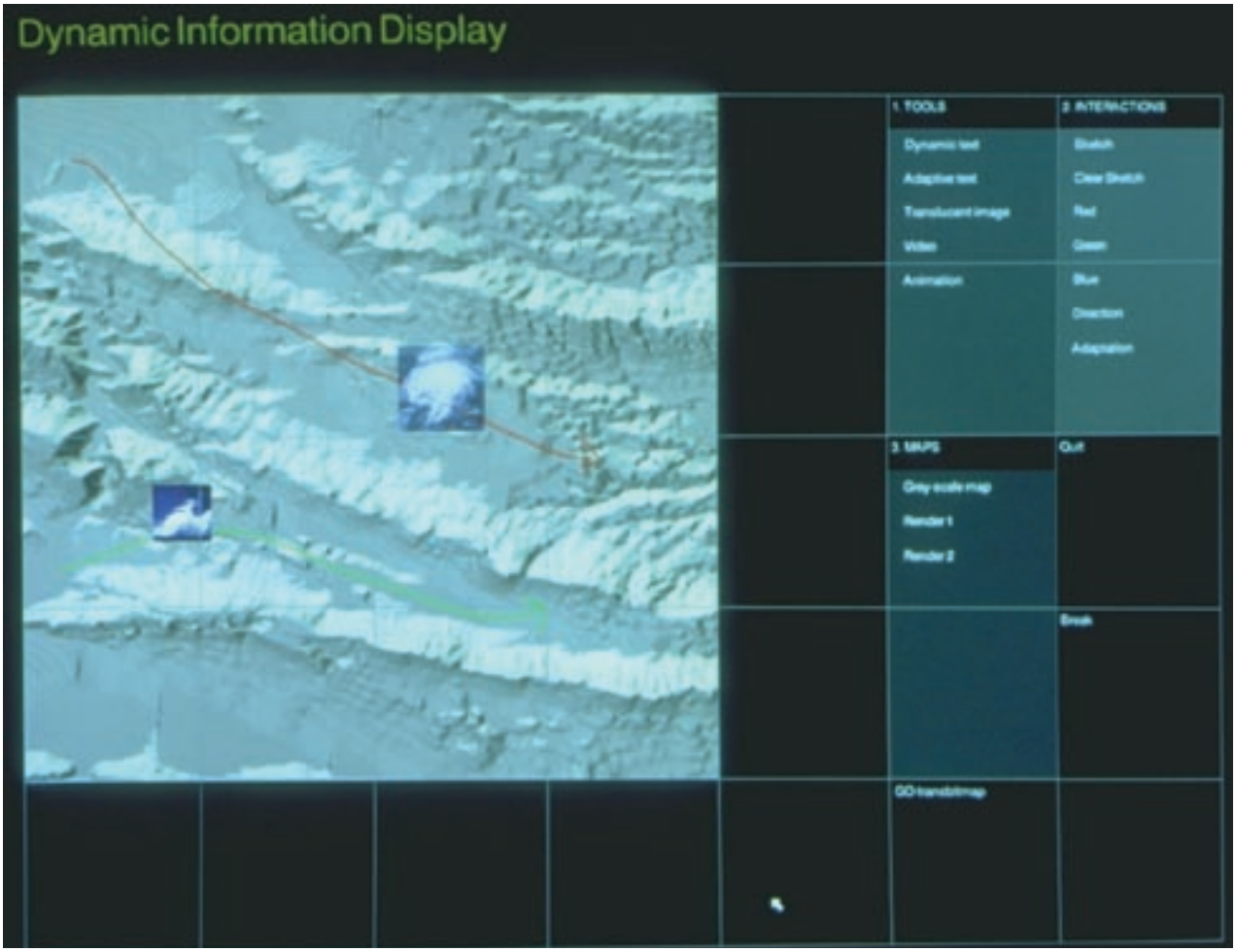


Figure 8. Learning by examples from maps

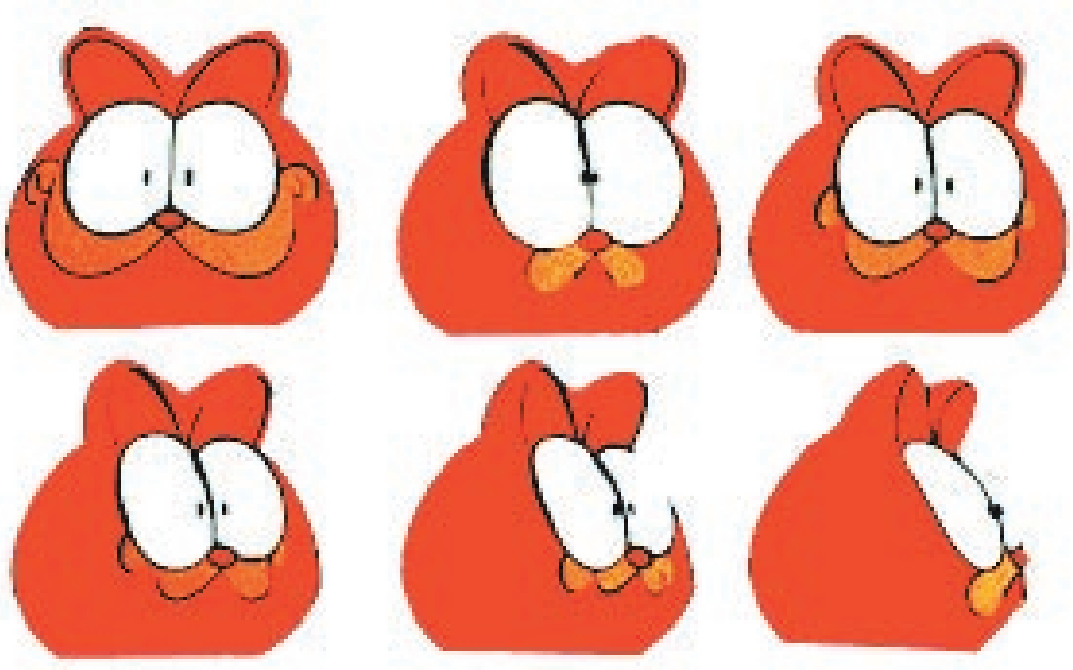


Figure 9. Intelligent interpolation

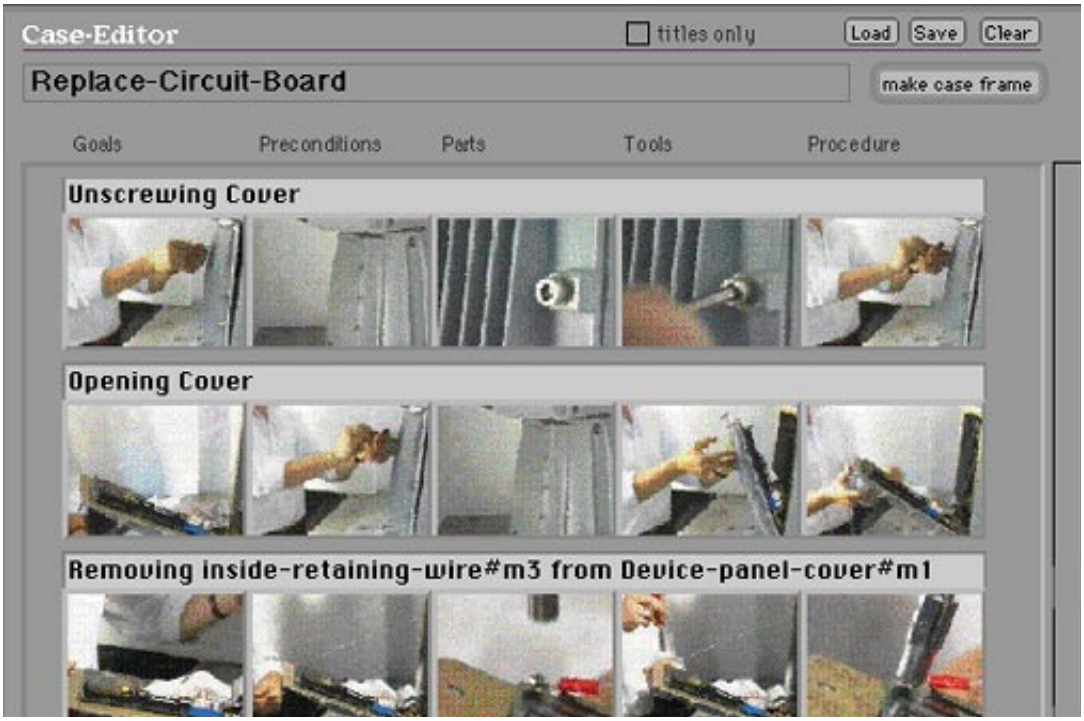


Figure 10. Case-based reasoning for presentations

and computers. Graphics is about visually representing the world and visually representing our ideas. Artificial intelligence is about symbolically representing the world, and symbolically representing our ideas. And between the visual and the symbolic, between the concrete and the abstract, there should be no boundary.

Acknowledgments

Thanks to Ted Selker, Wendy Richmond, and Ken Kahn for comments on the manuscript. I would also like to thank the many sponsors of research at the Visible Language Workshop, including Alenia, Apple, ARPA, DEC, Linotype-Hell, HP, IBM, JNIDS, the News in the Future Consortium, NSF, U.S. Dept. of Transportation, and other sponsors of the MIT Media Laboratory.

References

1. Baecker, R.M. *Interactive Computer-Mediated Animation*. Ph.D. dissertation, MIT, 1969.
2. Colby, G. Maintaining legibility, structure, and style of information layout in dynamic display environment. In *Proceedings of CHI '92* (Monterey, Calif., May 1992).
3. Cooper, M. Computers and design. *Design Quarterly 142* (1989), 22–31.
4. Kahn, K. Drawing on napkins, video-game animation, and other ways to program computers. *Commun. ACM 39*, 8 (Aug. 1996).
5. Kirsch, R. The anatomy of painting style: Description with computer rules. *Leonardo 21*, 4 (Dec.1988).
6. Krueger, M. *Artificial Reality*. Addison-Wesley, Reading, Mass., 1983.
7. Kuhn, T.S. *The Structure of Scientific Revolutions*. University of Chicago Press, 1970.
8. Lieberman, H.A. Three-dimensional representation for program execution. In E.P. Glinert, Ed., *Visual Programming Environments: Applications and Issues*. IEEE Press, 1991.
9. Lieberman, H. Tinker: A programming by demonstration system for beginning programmers. In A. Cypher, Ed., *Watch What I Do: Programming by Demonstration*, MIT Press, 1993.
10. Lieberman, H. Mondrian: A teachable graphical editor. In A. Cypher, Ed., *Watch What I Do: Programming by Demonstration*, MIT Press, 1993.
11. Lieberman, H. The visual language of experts in graphic design. In *Proceedings of the IEEE Symposium on Visual Languages* (Darmstadt, Germany, September 1995).
12. MacNeil, R. Generating multimedia presentations automatically using Tyro, the constraint, case-based designer's apprentice. In *Proceedings of the IEEE Workshop on Visual Languages* (Kobe, Japan, October 1991).
13. Marcus, A. *Graphic Design for Electronic Documents and User Interfaces*. Addison-Wesley, Reading, Mass., 1992.
14. McCorduck, P. *Aaron's Code: Meta-art, Artificial Intelligence, and the Work of Harold Cohen*. W.H. Freeman, 1991.
15. Strausfeld, L. Financial viewpoints: Using point-of-view to enable understanding of information. In *Proceedings of CHI'95* (Denver, CO, May 7–11, 1995).
16. Sutherland, I.E. *Sketchpad: A Man-Machine Graphical Communication System*. SJCC, AFIPS, 1963.
17. Vertelney, L., Arent, M., and Lieberman, H. Two disciplines in search of an interface: Reflections on a design problem. In Brenda Laurel, Ed., *The Art of Human-Computer Interface Design*. Addison-Wesley, Reading, Mass., August 1989.
18. Weitzman, L. and Wittenburg, K. Automatic presentation of multimedia documents using relational grammars. In *Proceedings of ACM Multimedia '94* (San Francisco, Calif., Oct. 15–20, 1994).
19. Wurman, R.S. *Information Architects*. Graphis Press, Zurich, 1996.

About the Author:

HENRY LIEBERMAN is a research scientist at MIT's Media Laboratory. His work lies at the intersection of human-computer interface, artificial intelligence, and computer graphics. His current projects involve interactive interfaces that can learn from the user by example, and more recently, intelligent agent interfaces for browsing the Web. **Author's Present Address:** Media Laboratory, Massachusetts Institute of Technology, 20 Ames Street, Cambridge, MA 02139. email: lieber@media.mit.edu

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copying is by permission of ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.

© ACM 0002-0782/96/0800 $3.50