Andries van Dam

Post-WIMP User Interfaces

the human connection

UNLIKE THE STEADY, PREDICTABLE HARDWARE PRICE/PERFORMANCE improvement described by Moore's law, there has been no such year-to-year progress in user interface design. Instead, the history of user interfaces can be characterized using evolutionary biologist Steven Jay Gould's notion of punctuated equilibrium: long periods of stability interrupted by rapid change. We can identify four separate generations characterized by four distinguishable interface styles, each lasting for many years and optimized to the hardware available at the time. In the first period, the early 1950s and 1960s, computers were used in batch mode with punched-card input and line-printer output; there were essentially no user interfaces because there were no interactive users (although some of us were privileged to be able to do console debugging using switches and lights as our "user interface"). The second period in the evolution of interfaces (early 1960s through early 1980s) was the era of timesharing on mainframes and minicomputers using mechanical or "glass" teletypes (alphanumeric displays), when for the first time users could interact with the computer by typing in commands with parameters. Note that this era persisted even through the age of personal microcomputers with such operating systems as DOS and Unix with their command line shells.

During the 1970s, timesharing and manual command lines remained deeply entrenched, but at Xerox PARC the third age of user interfaces dawned. Raster graphics-based networked workstations and "point-and-click" WIMP GUIs (graphical user interfaces based on windows, icons, menus, and a pointing device, typically a mouse) are the legacy of Xerox PARC that we're still using today. WIMP GUIs were popularized by the Macintosh in 1984 and later copied by Windows on the PC and Motif on Unix workstations. Applications today have much the same look and feel as the early desktop applications (except for the increased "realism" achieved through the use of drop shadows for buttons and other UI widgets); the main advance lies in the shift from monochrome displays to color and in a large set of software-engineering tools for building WIMP interfaces. I find it rather surprising that the third generation of WIMP user interfaces has been so dominant for more than two decades; they are apparently sufficiently good for conventional desktop tasks that the field is stuck comfortably in a rut.

I argue in this essay that the status quo does not suffice—that the newer forms of computing and computing devices available today necessitate new thinking

about fourth-generation UIs, what I call post-WIMP user interfaces. These don't use menus, forms, or toolbars, but rely on, for example, gesture and speech recognition for operand and operation specification. They started coming out of research labs in the early 1990s but haven't yet reached widespread deployment. In this essay, I focus on the user's side of the interface—interaction devices and techniques. Space does not permit discussion of the equally important computer-output or information-presentation side of interaction. Information visualization is an important area of study in its own right [5] and deals with such issues as spatial metaphors (e.g., the 3D cityscape vs. the 2D desktop metaphor for organizing information.)

I've been interested in 3D graphics since the late 1960s. I also worked on the second-oldest hypertext system (Doug Engelbart's NLS was the first) with Ted Nelson in 1967 and 1968. From that collaboration I gained a keen interest in user interfaces, and since then have guided my 3D graphics research group progressively more into user interface concerns, focusing on post-WIMP interfaces since the early 1990s.

Advantages of WIMP GUIs

Let's look at the pros and cons of WIMP GUIs to see why a new generation of UIs is needed. In the first few decades of computing we concentrated on functionality and performance of applications. But during the 1980s, when desktop productivity tools began to be used by millions of people, we recognized that, above some minimal threshold of functionality and performance, the most important predictor of an application's success was its ease of use (how "user-friendly" it is), both for novices and experienced users. For novices, ease of use is dictated primarily by how easy to learn and remember the interface is, while for power users the concern is less with the learning curve than with the effort required to be highly productive. The same interface is rarely optimal for both classes of users.

However user-friendly, an interface is still an intermediary between the user's intent and execution of that intent. As such, it should be considered at best a necessary evil because no matter how fluid, it still imposes a layer of cognitive processing between the user and the computer's execution of the user's intent. The ideal interface is no interface—"I think, therefore the computer gives me what I thought about (and what I should think about)". A more feasible goal to strive for is the computer as perfect butler, à la Jeeves, who knows my context, tastes, and idiosyncrasies and discreetly does my bidding by anticipating my needs without needing explicit direction. When I do direct this butler I communicate primarily by speaking, gesturing, facial expression, and other forms of human communication. In addition, I might still want to make a sketch to indicate a design or a concept or select from a menu of options, even enter information via a keyboard (still an efficient means of communication). The goal we strive for with today's user interfaces is to minimize the mechanics of manipulation and the cognitive distance between intent and the execution of that intent. To state the obvious, the user wants to focus on the task, not the technology for specifying the task. Thus I see WIMP and post-WIMP GUIs as successive waystations toward a much more powerful and natural interface that will ultimately take decades to develop.

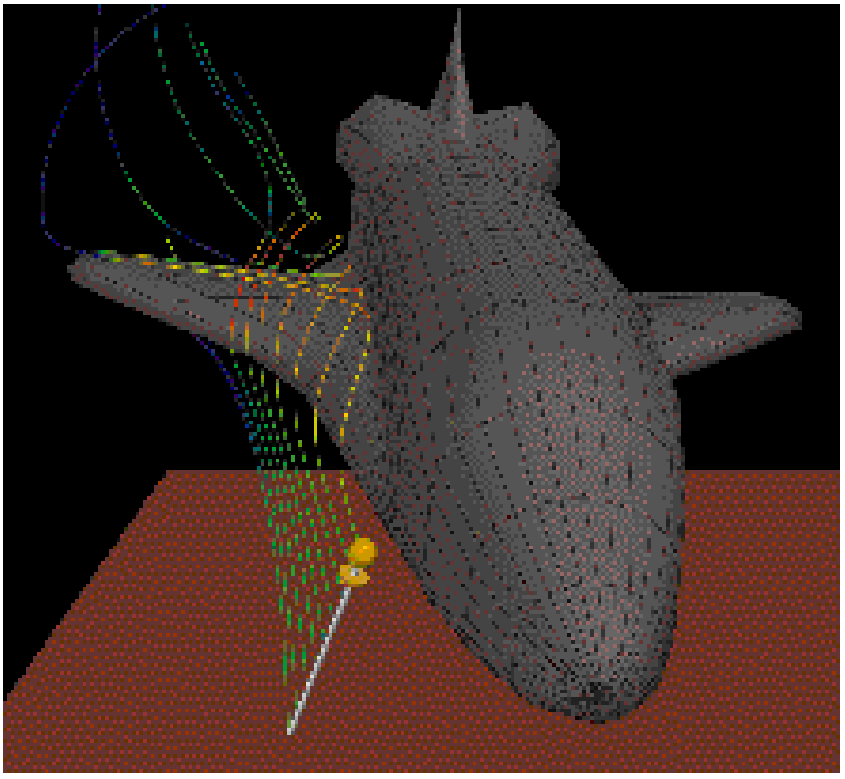

Figure 1. A 3D widget is used to manipulate elements of a 3D scene

While WIMP GUIs are a long way from a butler-style interface, let me list some measures of their success and importance. They have enabled three classes of users to be comfortable with computers who in general couldn't use them with earlier interfaces: young children who cannot yet read or write, managers, and non-professional home users. "Point and click," the hallmark of WIMP GUIs, has become part of modern culture. User interface design has become a specialty and user interface designers are highly sought after. In fact, testing in usability labs that "shakes down" the user interface is now a necessary component of the application development process. What WIMP GUIs have made possible is a de facto standard for the application interface that, compared to command line interfaces, gives us (relative) ease of learning, ease of use, and ease of transfer of knowledge gained from using one application to another because of consistencies in look and feel. "No one reads manuals anymore" because by and large they don't have to.

Drawbacks of WIMP GUIs

Nonetheless, significant problems remain. First, the more complex the application, the nonlinearly harder the interface is to learn because of the profusion of widgets and features, each of which is individually easy to learn but in the aggregate creates complexity. Many modern desktop applications are so large that users drown in the functionality and don't want to upgrade to newer releases, preferring to stick with the small subset they know, as described by the classic 90/10 rule. Second, users spend too much time manipulating the interface, not the application. Expert users are often frustrated by too many layers of "point and click" and screen clutter due to too many widgets, and prefer keyboard shortcuts. Third, WIMP GUIs, with their 2D widgets, were designed for and are well suited to 2D applications such as word processing, document layout, and spreadsheets. When the information is three-dimensional, the mapping between 3D tasks and 2D control widgets is much less natural. The WIMP interface for 3D applications today typically consists of control panels of 2D buttons and sliders that surround the 3D world (what I call the "TV model") and are used to control 3D cursor and viewpoint manipulation and object editing, thereby introducing significant indirection and "cognitive distance." In general, 3D applications tend to have greater visual complexity than 2D applications, another strain on WIMP GUIs. Fourth, "mousing" and keyboarding are not suited to all users, either because they don't find it natural or because they develop repetitive stress injuries, not to mention the special needs of users with disabilities.

A deficiency more serious than any of these that affects all users is that, unlike the ideal butler-style interface, WIMP interfaces do not take advantage of speech, hearing, and touch. While a large percentage of our neurons lie in the visual cortex and vision is our highest-bandwidth information channel, it is difficult for us to communicate in the physical world without speech, sound, and touch as well. As Bill Buxton (of Alias/Wavefront) points out, WIMP GUIs based on the keyboard and the mouse are the perfect interface only for creatures with a single eye, one or more single-jointed fingers, and no other sensory organs.

Yet another limitation of WIMP GUIs is that they are designed for a single desktop user who controls objects that have no autonomy and at most react to mouse manipulation. The interaction is one channel at a time, half-duplex; the system responds to each discrete input event and these events can be easily decoded, consisting of simply key presses and mouse selections. The most complicated input is a sequence of mouse positions that represents, for example, the trace of a paint brush.

Consider the following scenario at the opposite of extreme from the desktop productivity application, one that we'll see increasingly in applications ranging from games to virtual prototyping and training in immersive ("virtual reality") environments. Multiple participants are interacting on a shared task, each having some kind of wide-field-of-view stereo immersive display controlled via head and hand tracking, voice and gesture recognition and the manipulation of a variety of interaction devices with more than the two degrees of freedom provided by a mouse. The interaction here involves multiple parallel high-bandwidth input and output channels operating full-duplex on continuous (not discrete) signals that are decoded and probabilistically disambiguated in real time. WIMP interfaces simply cannot handle this much more demanding form of interaction; while they won't disappear, they need at least to be augmented.

Post-WIMP Interfaces

A post-WIMP interface to me is one containing at least one interaction technique not dependent on classical 2D widgets such as menus and icons. Ultimately it will involve all senses in parallel, natural language communication and multiple users. The most common examples of post-WIMP interaction commercially

available today are pen-based gesture recognizers used in hand-held PDAs such as the Apple Newton or the U.S. Robotics Pilot. These devices more or less successfully meld both WIMP and post-WIMP techniques for 2D tasks. Another instructive example of natural human-computer interaction that uses no WIMP devices or techniques is arcade games such as driving simulators with a steering wheel and gear shift and golf simulations in which the player swings a real club to hit a real ball, the trajectory of which is then simulated and displayed.

For 3D modeling, 3D widgets are increasingly used that are part of the 3D scene and obviate some of the conventional 2D widgets that otherwise would surround the 3D scene. As an example, the 3D components of the "rake" widget [1] are used to control the rake length, the number of streamlines, and their position in a scalar field derived from a numerical simulation of air flow around a space shuttle model (Figure 1). Among the now-common 3D widgets are scale/rotation boxes and handles and navigation controllers (used in VRML and other 3D browsers). Another technique that combines WIMP and post-WIMP to "get the interface out of your face" (as Bill Buxton calls his crusade) is the use of "marking menus," a modern form of multilevel radial menus for which the user can exploit muscle memory and perform menu selection gesturally with a mouse or stylus without having the menu actually appear [3]. Buxton is also a proponent of two-handed input in which the non-dominant hand controls coarse movement (e.g., placement of a tool) and the dominant hand fine adjustment (the manipulation of the tool) [2].

While speech recognition has been used for providing commands and even unrestricted textual input, the technology is still not sufficiently robust for wide usage. Part of both the attraction and difficulty of implementing continuous gesture and speech input is that such input is much harder to tokenize and disambiguate into verb, noun, and modifier components. Bob Zeleznik and colleagues at Brown University have recently demonstrated a 3D modeling interface that relies purely on gestural sketching of geometry and commands and has a vocabulary of over 20 gestures [9]; they have extended this type of interface to two-handed interaction, a combination of techniques that shows much promise [8]. This system, Sketch [9], was used to create a 3D model of a baseball field in a matter of mere minutes (Figure 2). Sketch uses unconventional, non-realistic rendering techniques to make the 3D model appear like an informal line drawing. The two car images depicted in Figure 3 show how an exhaust plume can be positioned in 3D using both hands simultaneously (to control the red and yellow cursors) to manipulate the plume and its shadow. While the learning curve is initially steeper than with a WIMP interface, the practiced user experiences reduced cognitive effort.

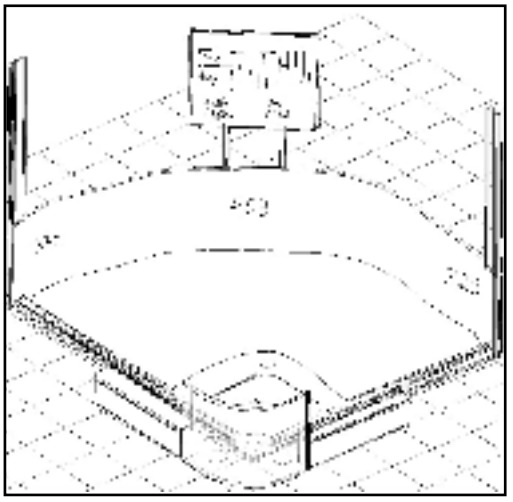

Figure 2. A 3D model appears like a line drawing in the Sketch system [9]

Haptic user interfaces are another, even more unexplored area, since haptic hardware has only recently become affordable. Haptic devices, unlike other interaction devices, can both sense and send information. Thus, most haptic interface designers have two different yet important points to consider: the tactile sense (i.e., feeling by touch) and the kinesthetic sense (the body's sense of where it is). The most common of these involves a force-feedback device such as the PHANToM (by SensAble Devices—see http://www.sensable.com) that receives position and gestural information and returns point-force feedback. Thus a user can feel the shape of a rigid object or even push through layers of different resistance (useful in, say, surgical simulation).

The Future

What may we expect in the future? Because of Moore's law (which shows no signs of letting up for the next decade) and continuing advances in panel display and projector technology, we may expect hugely powerful ubiquitous computers [7] in many different form factors. Examples include wearable computers (the CMU wearable computers [6]), PDAs, whiteboard-sized or even wall-sized displays, as well as lightweight, minimally intrusive head-mounted displays for virtual and augmented reality. In augmented reality, the user sees computer-gener-

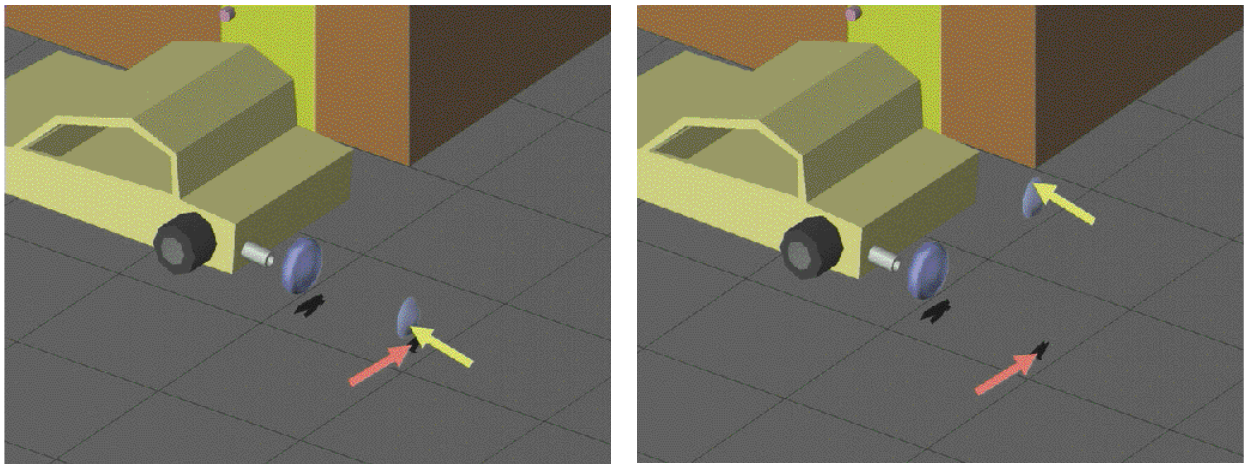

Figure 3. An example of two-handed 3D manipulation

ated information superimposed on real-world images via optical or video merging to provide, for example, annotation or x-ray vision.

Display resolution will have increased well beyond the inadequate 70–100dpi of today to make online reading less fatiguing and more pleasant. We will finally have what we need even more urgently for post-WIMP interaction: non-invasive, accurate sensors with good spatial and temporal resolution for head, body, and eye tracking. These will make possible fast and robust gesture recognition and possibly even non-invasive or minimally invasive biofeedback, especially important to disabled users. Voice recognition based on (limited) natural language understanding will be a dominant form of user-computer interaction. Raj Reddy of CMU has described SILK interfaces that will support speech, image, and language understanding, all driven by knowledge bases [4].

We will finally have what we desperately need: non-invasive, accurate sensors with good spatial and temporal resolution for head, body, and eye tracking. These will make possible fast and robust gesture recognition and possibly even non-invasive or minimally invasive biofeedback, especially important to disabled users. Voice recognition based on (limited) natural language understanding will be a dominant form of user-computer interaction. Haptic display will augment our ability to perceive computer-generated information. Computing power will finally be able to handle large amounts of continuous information from many input channels simultaneously while running real-time simulations of autonomous, reactive objects. And we will learn how to combine the best of user-empowering direct user control via WIMP and post-WIMP interfaces with indirect control provided by certified, trusted agents that can anticipate needs and work offline on our behalf with other such agents distributed over the net. Needless to say, such agent technology must be based on powerful (as yet non-existent) knowledge bases. This combination will finally allow us to approach the ideal situation in which user-computer interaction is at least as natural and powerful as human interaction. ◻

Acknowledgments

Thanks to Bob Zeleznik for critical reading and helpful suggestions.

References

1. Herndon, K.P. and Meyer, T. 3D widgets for exploratory scientific visualization. In *Proceedings of UIST '94, ACM SIGGRAPH*, (November 1994), pp. 69–70.
2. Kabbash, P., Buxton, W., Sellen, A. Two-handed input in a compound task. In *Proceedings of CHI '94*, 417–423.
3. Kurtenbach, G. and Buxton, W. The limits of expert performance using hierarchic marking menus. In *Proceedings of InterCHI '93*, 482–487.
4. Reddy, R. Turing Award Lecture: To dream the possible dream. *Commun. ACM 39*, 5 (May 1996), 105–112.
5. Robertson, G., Mackinlay, J. and Card, S. Information visualization using 3D interactive animation. *Commun. ACM 36*, 4 (Apr. 1993), 57–71.
6. Smailagic, A., and Siewiorek, D.P. The CMU mobile computers: A new generation of computer systems. In *Proceedings of COMPCON'94*. IEEE Computer Society Press, February 1994.
7. Weiser, M. Some computer science problems in ubiquitous computing. *Commun. ACM 36*, 7 (July 1993), 74–84.
8. Zeleznik, R.C., Forsberg, A.S., and Strauss, P.S. Two-pointer input for 3D interaction. To appear in *Proceedings of 1997 Symposium on Interactive 3D Graphics* (Providence, Rhode Island, April 27–30, 1997).
9. Zeleznik, R.C., Herndon, K.P., and Hughes, J.F. Sketch: An interface for sketching 3D scenes. *Computer Graphics (Proceedings of SIGGRAPH '96)*, August, 1996

Andries van Dam (avd@cs.brown.edu) has a been member of Brown University's faculty since 1965, and was one of the Computer Science Department's founders and its first Chairman, from 1979 to 1985.