Characterizing tool use in an interactive drawing environment

Robert St. Amant and Thomas E. Horton
Department of Computer Science
North Carolina State University
EGRC-CSC Box 7534
Raleigh, NC 27695-7534
stamant@csc.ncsu.edu, tehorton@eos.ncsu.edu

ABSTRACT

The metaphor of tool use for describing the interaction between a human and a computer is pervasive in user interface design. The basic concept of tool use, however, is difficult to define precisely, for HCI purposes or in general. In this paper we argue that a close examination of physical tool use can improve the design of interactive software. We describe a drawing application, HabilisDraw, that incorporates some of the properties we associate with physical tools but are not commonly found in software: persistent tool objects that encapsulate behavior and information, that can be used in conjunction with one another, and that embody rich cues about their appropriate usage. Initial results from formative evaluation suggest that the approach has some promise.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces; I.2.0 [**Artificial Intelligence**]: General—*Cognitive simulation*

General Terms

Human Factors

Keywords

Metaphors, tool use, drawing, interface design

1. INTRODUCTION

The metaphor of tool use for describing the interaction between a human and a computer is pervasive in user interface design. It is surprisingly difficult, however, to define the concept of tool use precisely, especially in cognitive terms. Examples of tools in software vary widely. At one extreme, a tool can be as simple as an icon in a drawing application that sets a specific mode (e.g., "Hold the mouse down on the ellipse tool. . . ") At the other extreme we find hundreds of systems in the HCI literature characterized as "tools" that have little more in common than being interactive software. The goal of developing a detailed definition of tool use is not simply for linguistic reasons, or to separate interactive software arbitrarily into tools and non-tools; rather, we believe that a better understanding of the cognitive characteristics tool-using behavior will lead to the design of more effective software.

The tool metaphor, even loosely applied, has been extremely successful in HCI, and has influenced the design of interactive systems in many ways. The tool metaphor has driven the development of cognitive artifacts, which act analogously to simple tools in the physical world, to amplify capabilities (e.g., written lists as memory extensions), or to translate problems into more manageable form [10, 17]. Software environments have also taken lessons from environments for physical tools—for example, a blacksmith's shop [12], a carpenter's workbench [8], or a chef's kitchen [1]—by incorporating some of the same design properties: effective use of space [13], conceptual organization that reflects functional relationships, and exploitation of conventions [19]. It is easy to see how spatial and functional organization of tools in a physical environment, such as the carpenter's workbench, translates to comparable structure in a software environment, in the arrangement of menu options, floating palettes, icons on toolbars, and multiple window placement. It is also easy to see at least some parallels between cognitive and physical tools.

As user interface technology has matured, however, the distance between the source of the metaphor (tools) and its target (interactive software) has widened. More abstract, cognitively oriented tools make different demands on our tool-using software abilities than physical tools. For example, an expert draftsperson may have a great deal of facility with drawing implements, straightedges, compasses, and related tools; how much of this expertise translates to the common activities in a drawing application of clicking an object type and then outlining its bounding box, or dragging objects into different locations and orientations, or raising a dialog box to edit object properties? The benefits of the more abstract software interface (e.g., predictability, consistency, and generalizability) are well understood. One important tradeoff, less often considered, is that other elements, such as the directness of action, physical affordances, and the transfer of motor skills, may be lost.

We are developing a drawing application, called HabilisDraw, to help us explore the possible benefits and drawbacks of a more detailed correspondence between physical tool use and software tool use [25]. HabilisDraw includes persistent tools that encapsulate behavior and information, that can be used in conjunction with one another, and that embody richer cues about their appropriate usage than is common in other interfaces. Tools in HabilisDraw have some similarities to past work, such as "local tools" in the KidPad

Int. Symp. on Smart Graphics, June 11-13, 2002, Hawthorne, NY, USA.

drawing environment [3], but our research examines the conceptual foundations of tools and their use in closer detail.

In this paper we present a preliminary, general approach to interpreting tool use in interactive software. The first section below describes tool use in the physical world. In the sections that follow we discuss the implications for software tools, looking in particular at influences on the HabilisDraw system. The paper ends with a brief description of our early evaluation efforts.

2. PHYSICAL TOOL USE

To understand how people might apply their tool-using skills in a software environment, it would be helpful to have a model of the use of physical tools. Unfortunately, human tool use has received limited attention in artificial intelligence, human-computer interaction, cognitive modeling, and related fields, and we are aware of no formal representations that we can exploit as a foundation for improving software environments. We must therefore develop our own foundation. This section organizes some of the basic concepts of tool use; our account is based on research in anthropology [12], animal cognition [11, 23], experimental psychology [28], cognitive psychology [10], social psychology [22], situated cognition [1], and design disciplines [16, 19]. Figure 1 summarizes the discussion points we will make in this section.

We first observe that physical tools are persistent artifacts. This point has been made most strongly by Beck in research on non-human primate cognition [2]: "Thus tool use is the external employment of an unattached environmental object to alter more efficiently the form, position or condition of another object, another organism, or the user itself when the user holds or carries the tool during or just prior to use and is responsible for the proper and effective orientation of the tool." In software environments, "tools" are often global modes (as in many drawing applications) or transient dialogs (as in many word processing applications.) In physical environments, tools are persistent, structured objects with specialized behaviors and properties that can be manipulated. One can adjust a wrench or a T-square, for example, to a particular application; one chooses a particular type of hammer or saw for a job. The persistence of physical tools means that they can be reused [1], they can encapsulate information as well as behavior, and they can be combined with one another or interact for their effect.

In addition to characterizing tools as objects, we can make general observations about procedural structure in their use. What distinguishes tool-using behavior from other types of activity? The most colorful examples of procedural structure come from research in non-human primate cognition, where accounting for behavior in tool-using terms is an important issue.

- *Tool use involves direct action* [29]. A striking action with a stone, with the goal of cracking open a nut, is an example of tool use. In contrast, some primates show uncanny accuracy in dropping objects onto researchers [11]; this is considered tool-related behavior rather than actual tool use.

- *Tool use often amplifies existing behavior* [10]. Using a stick to extend one's reach (e.g., through a narrow opening, or to touch a moderately distant object) is a common aspect of tool use in experimental settings and in the wild [18].

- *Tool use is goal-directed activity* [11]. Sometimes desirable ends are achieved through the incidental or even accidental use of an object, which is not considered a tool in that case. Inferred intention is an important part of judging whether an activity constitutes tool use.

- *Tool use involves effective behavior.* One influential study involved monkeys given the task of pushing a reward out of a narrow, transparent tube; one monkey unwrapped a thick bundle of reeds held together with masking tape, but then tried to push with the tape instead of a reed [30].

In many domains, tools are not used in isolation; rather, we find tool-using environments, such as a carpenter's or mechanic's workshop, which suggest that sets of tools can be organized by their intended use. We can also identify abstract classes that give broad coverage of tools in general:

- *Tools that produce a persistent effect on materials or the environment.* We call these *effective* tools. Examples in the physical world include hammers, saws, screwdrivers, and so forth. Although effective tools are those that first come to mind when we think of tools, this category does not encompass all types.

- *Tools that provide information about materials or the environment.* These have been called *instruments* in the tool use literature [10]. Instrumentation may be built into an effective tool, as when a table saw indicates the angle at which it is cutting a board. Tools that act alone as instruments include measuring tapes, calipers, microscopes and magnifying glasses, and so forth.

- *Tools that constrain or stabilize materials or the environment for the further application of effective tools.* We call these *constraining* tools. Examples include clamps, rulers, and other devices that limit movement or flexibility. (Notice that screws and nails are not reusable and are commonly thought as materials rather than tools, in that they remain in the finished product.) We often find artifacts that are simultaneously constraining and effective tools. For example, a handsaw is effective, in that the blade makes a cut in a piece of wood, but it is also constraining, in that the breadth of the blade forces (or at least facilitates) a straight-line cut. That is, because the breadth of the blade must follow the toothed edge through the groove as the wood is cut, it is easier to cut in a straight line than not. We notice that jigsaws and keyhole saws have a very narrow blade just to relax this constraint.

- *Tools that demarcate the environment or materials.* The goal of demarcation is to distinguish similar areas or pieces of the environment so that they can be treated differently. Examples include the carpenter's pencil, pushpins, and working surfaces inscribed with fixed markings.

This taxonomy has the property that all of its categories describe relationships between a tool and the environment in which it is used. This point is worth making explicit: the applicability of a tool is determined by its ecological properties. For example, a car mechanic who lacks a hammer can use any sufficiently solid, heavy object that comes to hand. The ecological nature of tool use leads to other significant properties:

- *Tool use can be opportunistic.* Tools can be used for purposes not intended by their designers, as above.

- *Tools provide rich cues about their appropriate use.* The affordances of a tool become obvious in its use; the hammer is almost a canonical example.

- *Tool use involves establishing and exploiting constraints between the user and the tool, the user and the environment,*

Tools are persistent artifacts:
Tools are replicatable and reusable; tools encapsulate information as well as behavior;
tools can combine or interact with one another for effect.
Tool use exhibits significant procedural structure:
Direct rather than indirect action; amplification; goal-directed activity; effective behavior.
Tools fall into a natural taxonomy:
Effective tools produce a persistent effect on materials; *instruments* gather information;
constraining tools constrain or stabilize materials or the environment; demarcating tools spatially structure the environment.
Tool applicability is determined by ecological properties:
Tool use can be opportunistic; tools provide rich cues about their appropriate use;
effective use of space is important; tool use exploits constraints between the user, the tool, and the environment.

Figure 1: A summary of the properties of tools and tool use

and the tool and the environment [28]. The issue of interacting constraints becomes immediately obvious in the example of hammer use, if one is working in a tightly enclosed space, from an awkward position, with a sprained wrist, etc. From a more positive perspective, establishing appropriate constraints will enhance tool use, e.g., bracing oneself for a stronger hammer blow, or clamping materials for appropriate resistance. An important specialization of this property is that tool use is associated with effective use of space [13, 29]. Many experienced tool users lay out their tools before beginning a task, on the assumption that some common tools are almost always eventually needed and should be ready to hand.

These observations suggest that tool use is a rich area for exploration and research from a variety of viewpoints. Two areas that we have not touched on include the role of tools as mechanisms for communication in social interaction, and the visual/motor execution aspects of tool use. The use of tools for communication raises a new set of issues: for example, recognizing tool use sometimes depends on culture or convention. The determination whether an artifact is a tool or not can be difficult; a fist-sized piece of flint may be stone tool to one archaeologist, for example, but only a leftover artifact from the production of flint arrowheads to another archaeologist [19]. At the level of visual and motor activities, it is possible to see tool use as a form of visually guided activity [4]. This viewpoint provides some significant insights into tool use that we have only begun to explore [21, 24].

We are aware of no interactive software systems that incorporate a significant portion of the concepts outlined above. In our current research on the HabilisDraw system, described in the next section, we examine the potential benefits of incorporating some of the properties of physical tools into a drawing application.

In taking this direction an important open issue concerns the possible tradeoffs. We run several dangers. Software tools that more closely resemble physical tools may be less efficient, in a task analysis sense. They may be more difficult to use, by being overly constrained due to irrelevant physical considerations. They may be more difficult to learn in the first place; for example, in a conventional drawing application, many different types of objects are created by selecting a mode and drawing a bounding box. This kind of consistency is missing from drawing tools in physical environments; their uses must be learned separately. It might even be argued that we are inappropriately conflating different types of tools: physical tools are associated with amplification, while cognitive tools are concerned with problem transformation [10]. Finally, we face a practical limitation in working with conventional input devices, the keyboard and mouse—some types of physical affordances simply will not transfer from richer, physical tool-using environments.

These and related issues have been addressed to some extent by others. Gentner and Nielsen argue that richer cues can improve learnability and usability, offsetting the loss of consistency in the use of different tools [7]. Constraints are often viewed as opportunities for learning [6]. Other questions remain open, but in the end these can best be answered by the description and analysis of the design of specific systems.

3. HABILISDRAW

HabilisDraw[1] is a prototype drawing application for exploring the use of software-based tools in an intereactive system. In typical drawing applications, tool use generally refers to the selection and application of one of several independent modes—the cursor changes shape and a mouse click takes on a new function. The differences between drawing a picture on a computer and drawing with pen and paper are striking; in the real world, tool use is a far richer and more complex process, one that provides for multiple degrees of freedom in the interaction, with users creatively applying cognitive, visual, and motor skills to solving problems. If a mode-based model limits a user to a subset of his or her abilities as a tool user, then the integration of real world styles of tool use into the interaction might plausibly result in a more powerful and more intuitive interface. Such an interface might also lead designers to a greater understanding of intelligent tool-using behavior.

Figure 2 shows a diagram of the familiar "shoot the monkey" physics demonstration, created using the current version of HabilisDraw. The tree and the monkey's body and trajectory were drawn using a pen tool by itself. The circles forming the monkey's head and eye, and the arc's forming his arms, ear, and tail were all created using a pen tool in combination with a compass, as was the arc showing the trajectory of the arrow. The arrow itself, and the hunter's cap, head, and body were created using a pen and a pen combined with the compass. The angles forming the hunter's arms and legs were drawn using a pen tool with two rulers set-up as jigs. Finally, the bow was drawn using a pen tool, two rulers and a compass, all in combination. (The point of shooting the monkey, by the way, is to show that if the monkey lets go of the branch at the same instant the hunter fires directly at the the monkey, then the arrow—which, like the monkey, is accelerated by the Earth's gravity—will

[1]The name is based on that of *Homo habilis* ("handy man"), the first hominid species known to have made and used stone tools.

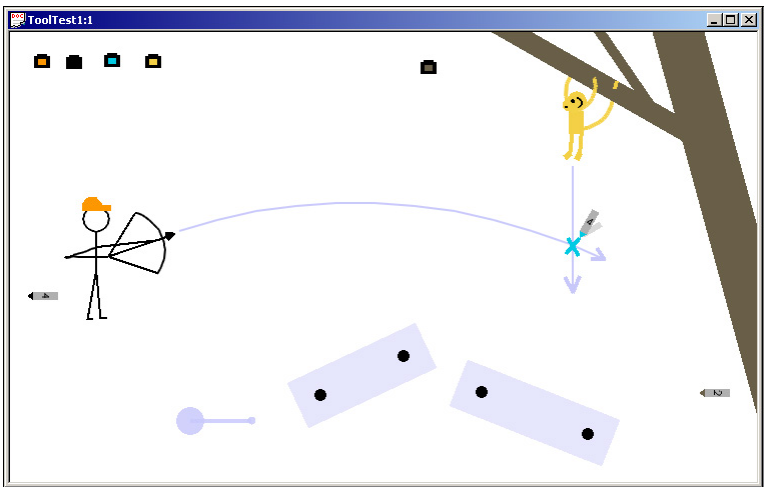


Figure 2: A drawing generated in HabilisDraw

strike the monkey as it falls; the actual trajectory is approximated here as a circular arc.)

While most of the user's interaction occurs within the boundaries of the drawing canvas, several functions are accessed by menu selections.

View→Show Tools toggles the display of tools on the page. Hiding the tools allows the user to see how the drawing appears without any additional clutter and without having to put all the tools away.

View→Show Lens toggles the visibility of the magnifying lens. Since the lens is only useful for detailed work, most of the time it is kept out of the user's way.

Window→Show Detail toggles the detail window. When the lens is displayed, the detail window shows the contents of the lens, but with added scalability. By increasing the size of the Detail Window, the user can zoom-in on the page further than with the lens alone.

Window→Show Properties brings up the properties window. When a drawn object or a tool is selected, its properties (length, line width, angle, color, etc.) are displayed in the properties window and may be directly edited by the user.

Window→New Palette, in the current version of HabilisDraw, does nothing more than bring up a new empty window. In future versions, however, each palette window might be capable of holding a customized set of tools for quick access. Often used sets (e.g., favorite ink well colors) could be saved for use with later documents.

In addition, a command line allows for scripting of the interface and for the generation and execution of macros. Our original intention was to provide for programmatic access to HabilisDraw functionality, but macro facility also suggests other directions, such as the approach taken by Eager in programming by example [5], or the possibility of end-user programming. Both of these possibilities are discussed further in Section 5.

A few general rules are followed throughout the HabilisDraw interface. To move or adjust any tool or drawn object on the page, the user right-clicks on the object and drags (drawn objects may also be moved with a left-click.) To activate a tool's function, the user left-clicks and drags. Clicking on a tool or drawn object will select it (currently, only one object may be selected at a time.) A selected object is displayed with the corners of a bounding box drawn around its perimeter. The properties of the currently selected object may be edited in the properties window.

3.1 Tools in HabilisDraw

HabilisDraw implements six types of tools so far: rulers, compasses, pens, ink wells, pushpins, and lenses, as shown in Figure 3. We do not claim that all these individual tools are unique (e.g., some CAD systems for architecture offer comparable functionality, if we take a task analysis view [15], and KidPad has obvious similarities [3].) The novelty of this work lies instead in our focus on tools as first-class artifacts and our explication of their conceptual foundations.

3.1.1 Pens

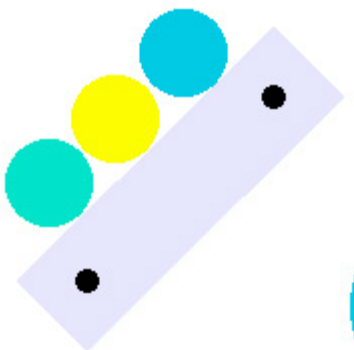
The most basic tool in HabilisDraw is the pen tool. Currently, this is the only tool capable of marking the page. The color of the pen tip indicates the color of the line that the pen will draw. The number on the body of the pen is the width (in pixels) of the line that will be drawn.

When inactive, a pen tool lies horizontally on the page. Right clicking on an inactive pen allows the user to drag the pen around the page without affecting any other tools or objects. Left clicking over an inactive pen will activate ("pick up") the pen. While multiple pens may be on the page, only one pen may be active at a given time. When active, the pen tool tracks the cursor position. A left click while a pen is active and not over another tool initiates free-hand drawing at the pen tip. The pen continues to mark the page until the user releases the left button. A left click while the tip is over an ink well will cause the pen to adopt the color of the ink in that well. Right clicking while the pen is not over another tool will "drop" the pen back to the page and it returns to an inactive state. To minimize the time users must spend switching tools, right clicking over another tool while a pen is still active, allows the user to move the other tool around the page, just as though the pen tool wasn't there.

A pen tool has five properties that may be set through the prop-

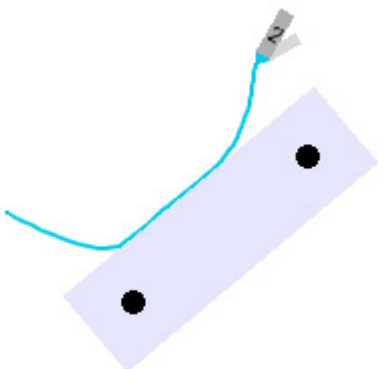
Pen Pushpin Ink wells Lens

Drawing with a pen

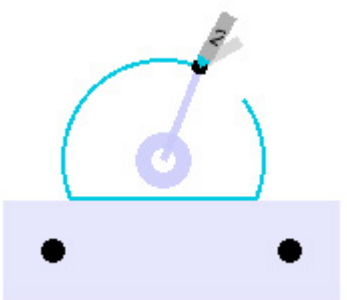


Dynamic alignment with ruler

Pen and compass



Pen and ruler interaction



Pen, compass, and ruler interaction

Pen, compass picture

Figure 3: Tools in HabilisDraw

erties window. Changing the width property sets the width of the lines drawn with the pen. Changing the R, G, and B properties sets the red, green, and blue components of the pen's drawing color respectively. Changing the A property sets the alpha value for lines draw with the pen.

Some classes of tools, including the ruler and compass tools, as described below, are designed to impose constraints upon the pen tool. To use a ruler or compass in combination with a pen, the user first picks up the pen by left clicking, then moves the pen to the helper tool. When the pen's tip is properly positioned, the helper tool's surface will be highlighted. Left clicking and dragging will cause the pen to draw a line constrained by the helper tool.

3.1.2 Ink Wells

Ink wells are some of the simplest tools in HabilisDraw. The only property of an ink well is the color of the "ink" it contains. This color is displayed on a label in the middle of the ink well. Usually, the ink well is a passive tool. Its primary function is to store a color for later use. Right clicking on the ink well and dragging allows a user to position the ink well on the page. Left clicking while dragging an ink well over a pen, another well, or a drawn object causes the well to "tip over" and assign its color to the object. Placing an active pen over an ink well and left clicking also transfers the color of the ink from the well to the pen. By left clicking on an ink well and dragging, a user can pull out a "siphon." If the user drags the end of the siphon to any point on the page and right clicks, the ink well will adopt the color immediately under the end of the siphon, be it the color of the background, a drawn object, or another tool.

The RGB and alpha components of an ink well may also be set directly through the properties window. In addition, by selecting the replace option, changing the color of the ink well will replace the color of all pens and drawn objects on the page currently colored with the ink well's old color with the ink well's new color.

3.1.3 Pushpins

While it seems that pushpins may hold great potential, they are perhaps under-used in the current version of HabilisDraw. Pushpins may be "stuck" anywhere in the page. If placed over a drawn object, the object is "pinned" to the page and cannot be repositioned until the pin is removed. A pin may also be used to constrain the movement of other tools, acting as a stop and blocking the drag of a ruler or the rotation of a compass.

3.1.4 Compasses

Both compasses and rulers may be used to constrain the motion of a pen tool as it draws. A compass tool constrains a pen's motion such that it follows a circular path around the center of the compass's base. The compass tool consists of a circular base and a circular activation site connected by a narrow arm. A right click and drag on the base allows a user to position the compass on the page. A right click and drag on the activation site allows a user to set the angle and distance between the base and the activation site. When an active pen tool is brought into range of the activation site, the site is highlighted. When in range, a left click starts the pen tool drawing in the center of the activation site. Dragging moves the pen and the site around the base, drawing a circular path with a radius equal to the distance between the centers of the base and activation site. Releasing the left mouse button stops drawing. If the button is released before the circle was closed, the resulting drawn object is only part of a circle—thus it is very easy to create arcs (often a complex task in conventional drawing programs.)

When moving the activation site, whether drawing or not, a label

appears near the base, displaying the distance and angle between the centers of the base and the activation site. These parameters may also be set explicitly using the properties window. Also in the properties window is a checkbox marked "Fill." When this box is checked, circles and partial circles drawn using the compass will be automatically filled in.

3.1.5 Rulers

Like compasses, rulers may be used to constrain the movement of a pen tool. During a free-hand draw, running the pen along the edge of a ruler creates a straight segment in the line. In addition, when an active pen is against the edge of a ruler, the edge becomes highlighted. Left clicking at this point will initiate drawing and bind the pen to the edge of the ruler, restricting drawing to a straight line of a length no greater than the length of the ruler. As the pen tool is moved from side to side, the end points of the line are extended, as though a real pen and straight edge were in use. Where rulers overlap, the pen tool may "cross over" from the edge of one ruler to the edge of another, forming a corner.

There are two ways to move a ruler tool. Right clicking and dragging a ruler will "pick up" the ruler from the page. The ruler becomes more transparent and it will not interact with other objects on the page as it is moved. Left clicking and dragging "pushes" the ruler along the surface of the page. Moving this way, a ruler tool behaves in a manner similar to an alignment stick [20], pushing around drawn objects, and aligning them along the edge of the ruler.

At each end of the ruler is a circular handle. Dragging one handle allows the user to adjust the length of the ruler and simultaneously rotate the ruler about the opposite handle. When dragging a handle, a label appears, displaying the length and angle of the ruler. The length and angle may also be specified directly by using the properties window.

3.1.6 Lenses

The final tool available in the current version of HabilisDraw is the lens, sort of a very basic version of a MagicLens [27]. Currently only one lens may be used. The lens may be displayed or hidden independently of the other tools using the "View" menu. Right clicking and dragging the frame of the lens positions it on the page. Right clicking and dragging the colored bottom right corner of the lens allows a user to adjust the size of the lens. Looking "through" the lens, a user sees a magnified display of the page underneath the lens. Opening the detail window using the "View" menu displays a second view of the page underneath the lens. By adjusting the size of the detail window, the view may be further scaled.

3.2 Tools and Objects in HabilisDraw

Like tools in the real world, tools in HabilisDraw are often more effective when they are used in combination. The simplest case is using an ink well to quickly assign a new drawing color to a pen tool—moving a pen over an ink well and left clicking copies the ink well's color into the pen. This effect is easily achieved in a conventional drawing application, but we note that the mechanism in HabilisDraw more closely reflects the interaction of plausible real-world tools.

More complex interaction is also possible with HabilisDraw. Tools such as the ruler and compass may be used to constrain the motion of a pen tool as it draws. A pen may be used in combination with a single ruler for drawing lines of a given length and angle, used with a compass for drawing circles and arcs of a given radius, used with several rulers to generate lines meeting to form angles, or used with both rulers and a compass to generate semi-circles. Using the tools in combination is much easier than trying to draw these objects free-hand, and, once the tools are in place, the user can use them as a jig to create multiple copies of the same drawn object.

Tools also interact with previously drawn objects. The lens tool alters the display of drawn objects by showing a magnified view of any objects within the lens boundaries. Like an alignment stick [20], the ruler tool may be used to push drawn objects around the page, aligning them along the ruler's edge.

4. EVALUATION

We have carried out a partial task analysis of HabilisDraw as well as an initial formative evaluation. A partial, informal account of both will be instructive.

4.1 Task Analysis

Consider the task of drawing a line at some desired angle. For a conventional system this involves the following steps, described at a relatively coarse level:

```
Move to line tool icon.
Click line tool icon.
Move to planned line start.
Mouse down.
Move to planned line end.
Mouse up.
```

A HabilisDraw sequence that involves the optional use of the ruler is similar but more than twice as long.

```
Move to one end of a ruler.
Mouse down.
Move to planned line start.
Mouse up.
Move to other end of ruler.
Mouse down.
Move to planned line end.
Mouse up.
Move to pen tool.
Left mouse click.
Move to planned line start.
Left mouse down.
Move to planned line end.
Mouse up.
```

This comparison speaks against the HabilisDraw sequence, raising the issue of efficiency mentioned earlier. If the task is to draw several lines at the same angle in different locations, however, the apparent gap closes between the two systems. In HabilisDraw, this iterative task involves moving the ruler, which retains its orientation, and then repeating the pen action. In a conventional interaction, this would entail either drawing new lines, with close attention to their being parallel, or copying the first line, pasting it, and then moving it to the desired location. A comparison between these two sequences depends on keystroke and mouse movement times, but HabilisDraw is no longer at a clear disadvantage.

The point of interest here is in the different conceptual approaches to the task. In the conventional system, it involves creating copies and modifying their properties, such as their location. In HabilisDraw, it involves creating a jig to constrain an action that is repeated to generate copies; here the focus is on the properties of the tool, rather than on the created objects.

Other distinctions are in the way objects types are related to one another. In HabilisDraw, for example, arcs are components of circles, and thus the two types of objects are generated by the same compass tool, following comparable movement patterns. In our

current design, though as yet not implemented, generation of ellipses requires an extra initial step of creating an additional focus point for the compass tool before the pen is activated. In conventional drawing packages, circles and ellipses are the same kind of object, with arcs treated separately. Depending on the type of drawing tasks common in a domain, one conceptualization may prove more useful than the other, but both approaches appear to be equally reasonable.

4.2 Formative Evaluation

While still in the early stages of development, we have begun conducting limited user studies to evaluate the HabilisDraw interface. Our eventual goal is to make systems like HabilisDraw more accessible to novice users, but we have not yet reached the point at which we see significant performance differences between HabilisDraw and other systems. Our formative evaluation has involved six subjects, four expert computer scientists and two novice computer users. While it is clear from these studies that we have a long way yet to go, the results are encouraging.

Novice users tended to have difficulty remembering the distinction between left and right mouse clicks and were often unsure of which should be used to achieve a desired effect. Both novice and expert users encountered difficulty when both mouse buttons needed to be used in combination (e.g. using an ink well to change an objects color). These results have led us to consider a simplified interface that would rely on only one mouse button.

Users also had difficulty selecting some of the smaller drawn objects and tool elements, suggesting that targets should probably be made bigger, the selection procedure more forgiving, and that more feedback mechanisms should be implemented. In addition, some features in the current version of HabilisDraw are invisible (e.g. the siphon function on the ink wells) requiring users to read the documentation before being able to fully exploit the tools. Since an intuitive interface is one of our primary goals for HabilisDraw, it is clear that we must find ways to make this hidden functionality visible.

Despite initial difficulties, after exploring the application for a few minutes, users (especially those with more computing experience) did become proficient in its use. We received favorable comments on most of the individual tools, e.g., "The compass was the tool that I enjoyed the most. I found it easy to use, especially when trying to determine how large I wanted my circle to be." and "Using the ruler to push the objects that I had created around produced some really interesting effects that I doubt I could achieve with other drawing tools." While the ruler tool was seldom used (in the tested version, drawing a straight line was more easily accomplished with the pen tool alone), the compass tool was quite popular with users, who found it very easy to predict the size of the circle or arc that would be created. Most users also liked the option of having multiple instances of a tool, each with different settings.

Novice users tended to dislike the two-button interface, while the biggest complaint from expert users was the lack of more advanced drawing functions (such as splines). Experts also desired a docking area, separate from the drawing surface, in which tools could be stored without cluttering the work space. Suggestions for improvement dealt mainly with parameterization of the tools we are developing, such as the sizes of various tool components (e.g., the visible handles on the ruler, the selectable legs the compass.) The system is still receiving simple adjustments along these lines. Users also met with some confusion concerning the different functions of the the right and left mouse buttons, but were eventually able to use all the tools effectively. The main result of our informal evaluation has been to suggest that more tools would increase the power of the system, and that some refinement is needed in the details of the tools' visual and behavioral properties.

5. DISCUSSION

Our goal has been to build a software system that more fully exploits properties of physical tool use, in particular those related to the persistence of tools and the ecological nature of their use. Our development of HabilisDraw has so far been an exploratory effort. We cannot hope to match the performance of expert users of the most modern, sophisticated illustration applications with the current version of the system. However, we believe from our initial experiences with users and our preliminary efforts in providing a foundation for HabilisDraw functionality in easily comprehensible physical behaviors, that this approach has some promise of improving interactive systems, in the illustration domain as well as others.

Our work has suggested two areas for future research, one arising from the treatment of tools as first class objects, the other from the ecological properties of tool use.

The tools we have built encapsulate information; they represent information, usually explicitly, about the objects they create. The potential benefit here is that HabilisDraw tools are designed to interact with one another via direct manipulation, and these interactions are tracked by changes to the information recorded in the tools. One of the difficulties faced by the designers of systems to support end-user programming in direct manipulation systems is the conceptual gap between displayed objects and symbols. In a direct manipulation system one generally acts through the interface, objects in the environment being treated as stand-ins for their real-world counterparts [9]. A programmable system works on symbolic and numerical representations of these objects, at a level of abstraction that may not be apparent to users who are not programmers. For applications such as spreadsheets and databases, bridging the conceptual gap is not difficult, but it is worth noticing that scripting and macro-writing are far more common in these symbolic and numerically oriented domains than in drawing applications. We speculate that the tool-based approach designed into HabilisDraw can help. HabilisDraw incorporates symbolic and numerical information in explicit tools; these tools mediate the actions of users to create and manipulate objects. We believe that tools provide a natural stepping stone between the concepts of direct manipulation and programming, making end-user programming a plausible goal for this work. Our development of a scripting language within HabilisDraw is a first step in this direction.

The structured properties of tools also suggest that HabilisDraw might provide an appropriate environment for programming by example, also called programming by demonstration [14]. In collaborative drawing environments such as KidPad [26], children can be observed giving one another guidance in finding and using tools; comparable guidance might be exploited by an enhanced version of HabilisDraw.[2] The intuition is that tools impose informative structure on an activity, and can be used to communicate intentions between collaborating tool users. On a physical drafting board, for example, a pushpin or compass or ruler left in a specific configuration can suggest an intended sequence of actions in a way that, in a software environment, the selection of a drawing mode or the setting of invisible constraints on drawn objects does not. Although we have not pursued this idea, we speculate that the placement of tools and the construction of tool combinations could guide an automated assistant in interpreting user actions.

On the topic of ecological tool use, it has turned out that much of the software development effort in HabilisDraw has been devoted

[2]Thanks to Michael Gegick for making us aware of this point.

to managing the interaction between different types of tools and objects. The result is a relatively sophisticated physical simulation system supporting the behavior and interaction of tools and drawn objects in the drawing environment, which may collide with one another or overlap for their effect. Because physical behaviors can be simulated without regard for any specific types of interaction, the system allows for opportunistic use and combination of tools. The small number of tools implemented in HabilisDraw has so far led to only a small number of examples of combinations, and no unambiguous examples of opportunistic user (except, possibly, the example of using the ruler as an alignment tool as well as a drawing aid), but we expect with a more mature implementation we will see more compelling examples.

6. ACKNOWLEDGMENTS

This effort was supported by the National Science Foundation under award 0083281. The information in this paper does not necessarily reflect the position or policies of the U.S. government, and no official endorsement should be inferred.

7. REFERENCES

[1] P. Agre and I. Horswill. Lifeworld analysis. *Journal of Artificial Intelligence Research*, 6:111–145, 1997.

[2] B. B. Beck. *Animal Tool Behavior: The Use and Manufacture of Tools*. Garland Press, New York, NY, 1980.

[3] B. B. Bederson, J. D. Hollan, A. Druin, J. Stewart, D. Rogers, and D. Proft. Local tools: an alternative to tool palettes. In *Proceedings of the 9th annual ACM symposium on User Interface Software and Technology*, pages 169–170. ACM Press, 1996.

[4] D. Chapman. Intermediate vision: Architecture, implementation, and use. *Cognitive Science*, 16(4):491–537, 1992.

[5] A. Cypher. Eager: Programming repetitive tasks by demonstration. In A. Cypher, editor, *Watch What I Do: Programming by Demonstration*. MIT Press, 1993.

[6] G. Fischer. Turning breakdowns into opportunities for creativity. *Knowledge-Based Systems*, 7(4):221–232, 1994.

[7] D. Gentner and J. Nielsen. The anti-mac interface. *Communications of the ACM*, 39(8):70–82, August 1996.

[8] S. Greenberg. *The computer user as toolsmith : the use, reuse, and organization of computer-based tools*. Cambridge University Press, Cambridge, UK, 1993.

[9] E. Hutchins. Metaphors for interface design. In M. M. Taylor, F. Neel, and D. G. Bouwhuis, editors, *The Structure of Multimodal Dialogue*, pages 11–28. North-Holland, Elsevier Science Publishers, Amsterdam, 1989.

[10] E. Hutchins. *Cognition in the Wild*. MIT Press, Cambridge, MA, 1995.

[11] E. J. Ingmanson. Tool-using behavior in wild *pan paniscus*: Social and ecological considerations. In A. E. Russon, K. A. Bard, and S. T. Parker, editors, *Reaching into Thought: The minds of the great apes*, chapter 9, pages 190–210. Cambridge University Press, Cambridge, UK, 1996.

[12] C. M. Keller and J. D. Keller. *Cognition and tool use: The blacksmith at work*. Cambridge University Press, Cambridge, UK, 1996.

[13] D. Kirsh. The intelligent use of space. *Artificial Intelligence*, 73(31–68), 1995.

[14] H. Lieberman, editor. *Your Wish is My Command: Giving Users the Power to Instruct their Software*. Morgan Kaufmann, 2001.

[15] M. McCullough. *Abstracting Craft: The Practiced Digital Hand*. MIT Press, Cambridge, MA, 1996.

[16] W. C. McGrew. The intelligent use of tools: Twenty propositions. In K. R. Gibson and T. Ingold, editors, *Tools, language, and cognition in human evolution*, pages 151–170. Cambridge University Press, Cambridge, UK, 1993.

[17] D. A. Norman. Cognitive artifacts. In J. M. Carroll, editor, *Designing interaction: psychology at the human-computer interface*, pages 17–38. Cambridge University Press, 1991.

[18] D. Povinelli, editor. *Folk Physics for Apes*. Oxford University Press, NY, 2000.

[19] B. Preston. Cognition and tool use. *Mind and Language*, 13(4):513–547, December 1998.

[20] R. Raisamo. An alternative way of drawing. In *Proceedings of CHI'99 (ACM Conference on Human Factors in Computing)*, pages 175–182, 1999.

[21] M. O. Riedl and R. St. Amant. Toward automated exploration of interactive systems. In *Proceedings of Intelligent User Interfaces*, pages 135–142, 2002.

[22] G. R. Semin. Cognition, language, and communication. In S. R. Fussell and R. J. Kreuz, editors, *Social and Cognitive Approaches to Interpersonal Communication*, pages 229–258. Lawrence Erlbaum, Mahwah, NJ, 1998.

[23] A. W. Smitsman. The development of tool use: Changing boundaries between organism and environment. In C. Dent-Read and P. Zukow-Goldring, editors, *Evolving explanations of development: Ecological approaches to organism-environment systems*. American Psychological Association, Washington DC, 1997.

[24] R. St. Amant. User interface affordances in a planning representation. *Human Computer Interaction*, 14(3):317–354, 1999.

[25] R. St. Amant and T. E. Horton. A tool-based interactive drawing environment (extended abstract). In *CHI '02 (ACM Conference on Human Factors in Computing), Extended Abstracts*, 2002. To appear.

[26] J. Stewart, E. M. Raybourn, B. Bederson, and A. Druin. When two hands are better than one: Enhancing collaboration using single display groupware. In *Proceedings of CHI '98 (ACM Conference on Human Factors in Computing), Extended Abstracts*, pages 287–288. ACM Press, 1998.

[27] M. C. Stone, K. Fishkin, and E. A. Bier. The movable filter as a user interface tool. In *Proceedings of CHI'94 (ACM Conference on Human Factors in Computing)*, pages 306–312. ACM Press, 1994.

[28] L. van Leeuwen, A. Smitsman, and C. van Leeuwen. Affordances, perceptual complexity, and the development of tool use. *Journal of Experimental Psychology: Human Perception and Performance*, 20(1):174–191, 1994.

[29] J. Vauclair. *Animal Cognition*. Harvard University Press, Cambridge, MA, 1996.

[30] E. Visalberghi and L. Limongelli. Acting and understanding: Tool use revisited through the minds of capuchin monkeys. In A. E. Russon, K. A. Bard, and S. T. Parker, editors, *Reaching into Thought: The minds of the great apes*, chapter 3, pages 57–79. Cambridge University Press, Cambridge, UK, 1996.