Applying Cartoon Animation Techniques to Graphical User Interfaces

BRUCE H. THOMAS
University of South Australia
and
PAUL CALDER
Flinders University of South Australia

If judiciously applied, animation techniques can enhance the look and feel of computer applications that present a graphical human interface. Such techniques can smooth the rough edges and abrupt transitions common in many current graphical interfaces, and strengthen the illusion of direct manipulation that many interfaces strive to present. To date, few applications include such animation techniques. One possible reason is that animated interfaces are difficult to implement: they are difficult to design, place great burdens on programmers, and demand high-performance from underlying graphics systems.

This article describes how direct manipulation human computer interfaces can be augmented with techniques borrowed from cartoon animators. In particular, we wish to improve the visual feedback of a direct manipulation interface by smoothing the changes of an interface, giving manipulated objects a feeling of substance and providing cues that anticipate the result of a manipulation. Our approach is to add support for animation techniques such as object distortion and keyframe interpolation, and to provide prepackaged animation effects such as animated widgets for common user interface interactions.

To determine if these tools and techniques are practical and effective, we built a prototype direct manipulation drawing editor with an animated interface and used the prototype editor to carry out a set of human factors experiments. The experiments show that the techniques are practical even on standard workstation hardware, and that the effects can indeed enhance direct manipulation interfaces.

Categories and Subject Descriptors: I.3.6 [**Computer Graphics**]: *interaction techniques*

General Terms: Design, Human Factors

Additional Key Words and Phrases: Animation, direct manipulation, graphical user interfaces, graphics, toolkits

Authors' addresses: B. H. Thomas, School of Computer and Information Science, University of South Australia, Mawson Lakes, SA, Australia 5095; P. Calder, Department of Computer Science, The Flinders University of South Australia, GPO Box 2100, Adelaide, Australia 5001.

1. INTRODUCTION

This article explores how techniques borrowed from cartoons [Thomas and Johnston 1984] and computer animation can enhance the experience of interacting with a computer. However, we are not concerned with portraying animated data, as would be the case when operating animation editing tools (such as an algorithm animation editor [Stasko 1991]) or using animation to supplement the presentation of otherwise static information (such as an animated help system [Sukaviriya 1988; Sukaviriya and Foley 1990]). Rather, our goal is to apply animation to the interface itself—to enhance or augment the effectiveness of human interaction with applications that present a graphical interface.

Like other recent work [Chang and Ungar 1993; Chatty 1992; Hudson and Stasko 1993], we set out to extend current direct manipulation style user interfaces with animation. In her book, *Computers as Theatre*, Laurel [1991] argues that both the computer and the human are active agents working together to achieve some common goal. It is the goal of the designer of an application's interface to facilitate these two active agents in their effort to collaborate as closely as possible. Laurel suggests the use of dramatic techniques to encourage users to suspend their disbelief. Animation can be used to enhance some of these dramatic effects; for example, users are more willing to suspend their disbelief if the graphics of an interface move in a smooth and realistic fashion. As viewers, we often suspend our disbelief while watching a scene of a movie or computer screen. For example, while watching Superman fly in a movie, we suspend our belief system in order to enjoy the movie. If it is too difficult to suspend disbelief, the movie is no longer enjoyable. Effective animation smoothes the motion of objects on a computer screen and simulates more closely the human agent's expectation of the physical world, leading to a readier acceptance of what is shown on the screen.

This article presents a set of animation effects targeted at two main areas of the user interface: widget components and graphical object manipulation. To make animation effects for user interfaces useful and pleasing to the user, we have developed effects based on traditional cartoon animation principles [Thomas 1998].

We implemented the animation effects as an extension to an existing user interface toolkit, InterViews [Thomas and Calder 1995a]. Specifically, we extended InterViews by adding support for staging animations over predetermined time intervals and modified the drawing model to add a new kind of nonaffine transformation: a warp transformation. Since all drawing in InterViews is subject to coordinate transformation, this one addition was sufficient to add warping capabilities to all InterViews objects.

2. SMOOTHING CHANGES IN THE INTERFACE

Baecker and Small [1990] suggest that animation can smooth transitions from one process to another in a user interface by focusing the user's attention on changes to the working environment, such as the location and identity of newly created entities and new areas of interest. Further, animations can improve the legibility of the interface by maintaining object consistency during changes.

Transitions may cause dramatic change to large portions of a computer's display. Consider the operation of a pop-up menu. At one instance, the menu is represented by just a button; the next instance a complete menu window is visible, obscuring the underlying information. The user is momentarily started; a small but significant disruption occurs before the new appearance of the screen is digested and work can proceed. With experience, users learn to ignore the disruption, but some of its effect remains. The overall impression is of an abrupt, hard-edged interaction that is jarring and tiring.

Animation can help smooth these rough edges by changing the screen's appearance from the old state to the new more gradually [Gonzalez 1996]. Researchers [Donskoy and Kaptelinin 1997; Robertson et al. 1989] have shown that a smooth change, even it if occurs rapidly, shifts the interpretation of the change to the user's perception system, allowing the cognitive system to stay focused on the task at hand. The overall effect is that the interface seems smoother and more pleasant to use.

Robertson et al. [Card et al. 1991; 1996; Mackinlay et al. 1991; Robertson et al. 1991] used this principle to good effect in the area of 3D visualization. The effectiveness of their work relies heavily on good interactive animation. One of the 3D information visualization tools they developed was the Cone Tree [Robertson et al. 1991], which uses smoothly rotating cones to display large hierarchies in a limited space. The perceptual phenomenon of object constancy enables users to track structural relationships without cognitive effort. When a rotation animation is complete, no time is needed to reassimilate the new configuration of the information space. The authors make two claims [Robertson et al. 1991]: "it seems clear that interactive animation can effectively shift cognitive processing load to the perceptual system. And it seems plausible (but not yet proven) that 3D can be used to maximise effective use of screen space."

Traditional graphical user interface toolkits provide a set of standard widgets for the application programmer, such as push buttons, pull-down menus, status buttons, and scrollbars. Each of these widgets provides graphical information reflecting the current state of the widgets; for example, a menu button is highlighted when it is activated. The widgets are carefully designed with a consistent and aesthetically pleasing static appearance, but little thought has so far been given to their dynamic appearance. Animation can help smooth abrupt changes in appearance by gradually changing the screen's appearance from the old state to the new. These forms of smoothing animation have been investigated by a number of researchers. Whizz [Chatty 1992; Chatty and Beaudouin-Lafon 1992] provides tools for building animated interactive applications. The Self animated widgets [Chang and Ungar 1993] provide menus that grow, dialog boxes that dissolve, arrows that grow and shrink smoothly, and objects that bounce when "hit". In the "buttonfly" 3D graphical menu system for Silicon Graphics workstations, activating a submenu button brings up a new menu by smoothly flipping the button over, exposing a new set of buttons.

We applied a variety of animation techniques to smooth the transitions in interface components that change appearance in predictable ways; we implemented the animations by modifying the components in a standard interface toolkit [Thomas and Calder 1995b]. Because the user requires a small but finite

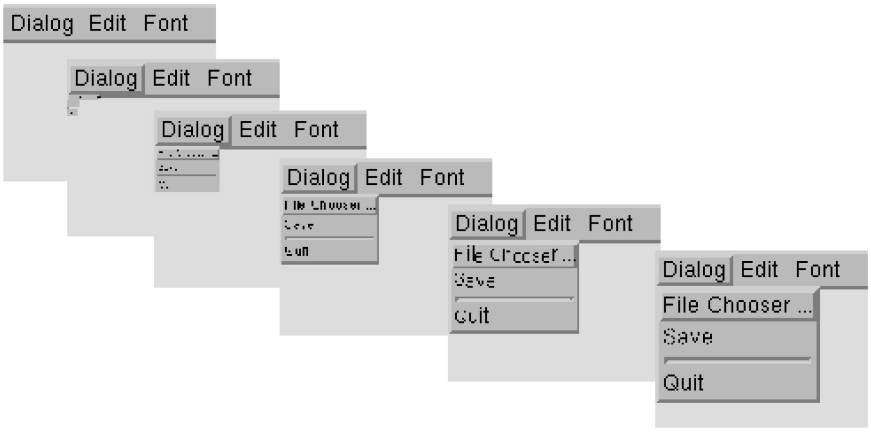

Fig. 1. An animated menu.

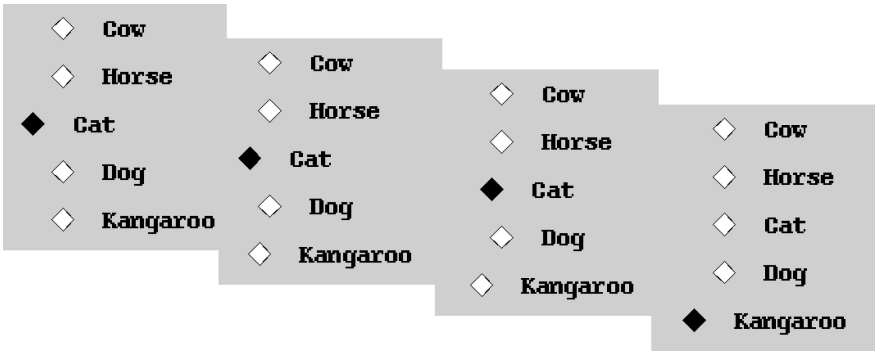

Fig. 2. Animated radio buttons.

amount of time to perceive and understand even an instantaneous change, animating the transitions makes them seem smoother without increasing the overall duration of the interaction [Card et al. 1991].

We modified pulldown menus so that the menu expands from zero to full size over a short but noticeable interval of time. (A similar menu animation is now incorporated into the Microsoft Windows interfaces.) Figure 1 shows that, as the menu expands, its contents scale smoothly to keep the window full. Following the animator's principle of "slow-in/slow-out," the menu grows slowly at first, quickly in the middle, and then slowly again as it approaches full size.

We also experimented with a variety of geometric transformations to create animation effects for various kinds of buttons. For example, we created push buttons that seem to recede when pushed, palette buttons that twist when selected, and checkboxes and radio buttons that slide to indicate the selected item. Figure 2 shows the effect for a bank of radio buttons.

Some changes cannot be represented by geometric transformations. In such cases we created individual frames for the animation, similar to the techniques used in old paper flipbook animations. For example, we used this technique to

create buttons whose label changed one letter at a time and checkboxes where the check mark appeared smoothly, as if stroked by a pen.

Finally, we created a palette menu where the menu choices are arranged in panels as if on the sides of a prism. If the desired choice is not in the visible panel, the prism can be rotated on its axis, bringing another panel of choices into view. The animation, which is based on the ideas that underlie the Cone Tree [Robertson et al. 1991], helps a user keep track of choices that have been temporarily rotated out of view.

3. GIVING MANIPULATED OBJECTS SUBSTANCE

Applications that present a direct manipulation interaction usually provide visual feedback to suggest the effect of the interaction. However, simple feedback techniques based on drawing object outlines tend to give the impression that a direct manipulation operation is happening to a surrogate object, rather than the real thing. The drawing of a full-featured object during the operation gives the object solidity, but can still fail to convey a sense of substance; somehow it seems *too easy* to manipulate the object. To be convincing, solid-seeming objects must do more than look solid; they must also *feel* solid. Cartoon animators often use techniques that mimic physical effects, such as inertia and friction, to reinforce the illusion of substance [Laybourne 1979]. Cartoon objects start moving slowly and come to a gradual stop, they move in curves rather than along straight lines, and they stretch and squash to suggest interaction with their environment.

The Self programming environment [Chang and Ungar 1993] incorporated cartoon-style animation techniques into its user interface. Chang and Ungar list three cartoon animation principles that apply to interface animation: solidity, exaggeration, and reinforcement. These principles are characterized as follows:

(1) Characters and objects should seem *solid.*
(2) *Exaggerating* the behavior of user interface objects makes the user interface more engaging.
(3) The interface should *reinforce* the illusion of reality.

Self's interface uses solid drawing, squash and stretch, motion blur, dissolves, anticipation, follow through, slow-in/slow-out, and arcs to strengthen the impression that programmers of Self programs are manipulating solid objects.

Artkit [Hudson and Stasko 1993] offers tools to the user interface application programmer to incorporate a similar set of animation techniques into their applications. Artkit includes all the animation effects from Self, with the exception of the dissolve technique, and includes pacing functions to control timing aspects of the animation effects. These functions allow the programmer to map the speed of a graphical object along a curved path to a nonlinear time function.

For interface objects that move spontaneously or along predefined paths, the application of animation techniques is straightforward. But in the context of direct manipulation, where users are in control of object movements, animated interface designers are faced with a dilemma: the need for immediate and accurate response to user actions is at odds with the need for objects to have

realistic-seeming behavior. If a user tries to rapidly drag an object across the screen, for example, how should the interface represent the inertia of the object resisting its movement? Or if a user tries to move an object that is fixed in position, how should the interface suggest the restraint?

The approach we take is to consider how a cartoon animator would depict such behavior. For example, to suggest movement caused by someone dragging an object, an animator could show the object distorted in the direction of the pull, as if the object is reluctant to change. To suggest an attempt to move a fixed object, the animator could show the object leaning in the direction of the pull, while its base stays fixed in place. With techniques like these an application can create the illusion of a greater sense of substance for its objects, while allowing users to feel that they are in control.

In the spirit of Chang and Ungar's principles for using cartoon animation in user interfaces [Chang and Ungar 1993], we have defined four new principles for animating direct manipulation operations [Thomas and Calder 1994]:

(1) **The principle of attachment** states that the objects being manipulated should at all times remain attached to the pointer, which maintains the impression that the user is always in control of the action.
(2) **The principle of reluctance** states that objects should, in general, seem reluctant to change, which reinforces the illusion of substance by suggesting that changing an object requires effort on the part of the user.
(3) **The principle of smoothness** states that objects must change in a continuous fashion, which reduces cognitive load by removing large and unexpected changes in visual information presented to the user.
(4) **The principle of anticipation** states that the result of a user's action must be obvious at all times, which reduces cognitive load by supplying additional visual information and minimizing the use of short-term memory.

This section looks at animation effects for direct manipulation operations such as *translating*, *scaling*, *rotating*, and *shape editing*. All of these operations change the visual characteristics of the object under the user's control. Adding animation to the visual feedback allows the application to display more information about the interaction. In addition to the current state of the objects, the animation can also suggest the direction and rate of change caused by the user's actions. This extra information gives the user a better understanding of the results of an action on an object. The electronic appendix provides movie animations demonstrating the translating, scaling, and rotating effects.

3.1 Translation

Figure 3 shows a direct manipulation move operation on a square. These static images cannot fully convey the dynamic feel of the interaction; several successive frames from the interaction were superimposed to suggest the effect. The shading of frames is intended to imply the passage of time, with the most recent frame being the darkest. The diagram on the left side of the figure shows the move operation without the animation effect; the square merely moves to follow the mouse. The diagram on the right side of the figure shows the move

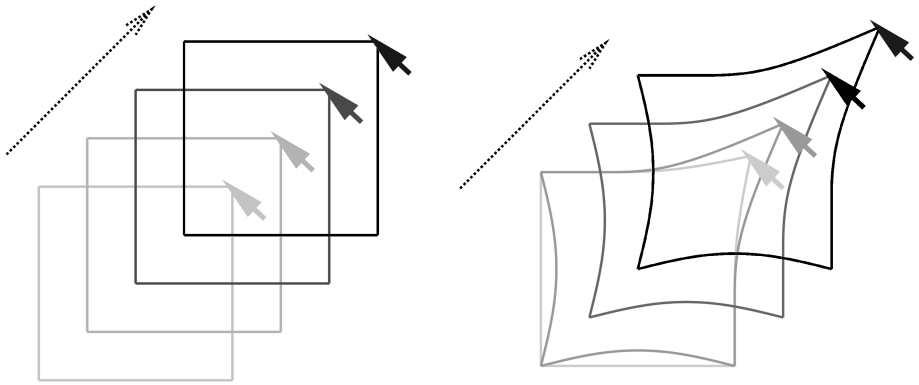

Fig. 3. Animating a move operation.

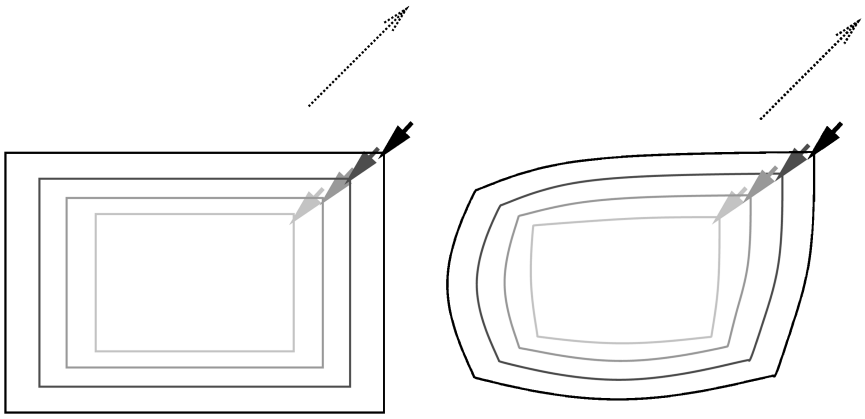

Fig. 4. Animating a scaling operation.

operation with animation; the corner of the square stays attached to the mouse point (the principle of attachment), while the bulk of the object lags slightly behind (the principle of reluctance.) This animation gives the effect of manipulating a heavy "rubbery" object that distorts as it is pushed and pulled. Although the effect does not correspond exactly to a physical model, the algorithm gives the impression that the shape is made of elastic material, with weights attached to the vertices, causing them to lag behind the movement.

3.2 Scaling and Rotating

The animation effect for translation is also effective when used in conjunction with other common direct manipulation operations. For example, Figure 4 illustrates an operation to make an object larger by scaling. The diagram on the left side of the figure is an unanimated version of the operation and the diagram on the right is an animated version. As in the translation animation effect example, the part of the object that is "grasped" is controlled by the mouse, while the bulk of the object lags behind.

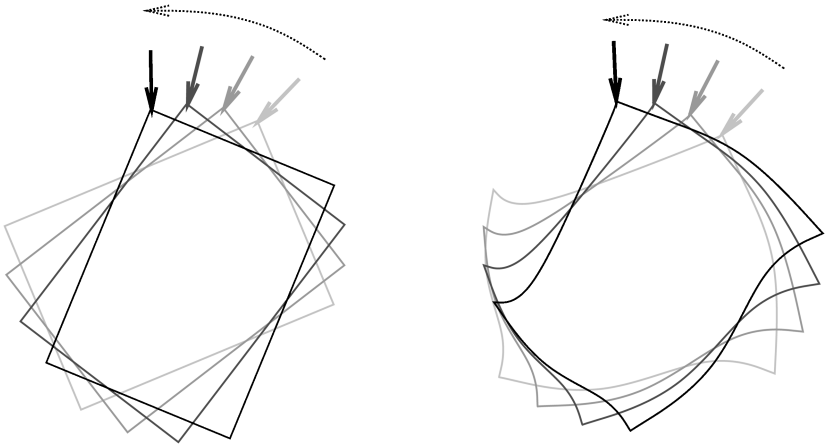

Fig. 5. Animating a rotate operation.

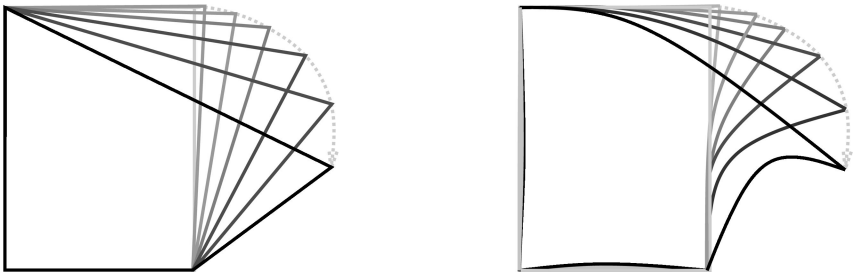

Fig. 6. Animating a vertex editing operation.

Figure 5 shows an operation to rotate an object anticlockwise about its center. An unanimated version of the operation is shown on the left diagram of the figure and an animated one on the right. Once again the animation effect is to have the object lag behind, so as to give the impression that the object has inertia.

3.3 Shape Editing

Operations that change an object's shape such as moving a vertex of a polygon can also benefit from animation effects. Moving a vertex entails changing the two adjacent sides of the polygon. Animation of this operation is achieved by distorting the sides that are changing. Figure 6 demonstrates the effect of moving the upper-right vertex of a square. The left diagram shows an unanimated version in which the vertex is moved from the original position to the final position (the black polygon has the vertex in the final position.) The right diagram shows the animated version. For all these animations, the magnitude of the effect depends on how fast the action is performed. Rapid movement causes more distortion, as it must do if the effect is to suggest a "real" object.

Moving a single vertex affects the object only locally; but as Figure 6 shows, the bulk of the object distorts slightly to indicate the "stress" it is suffering.

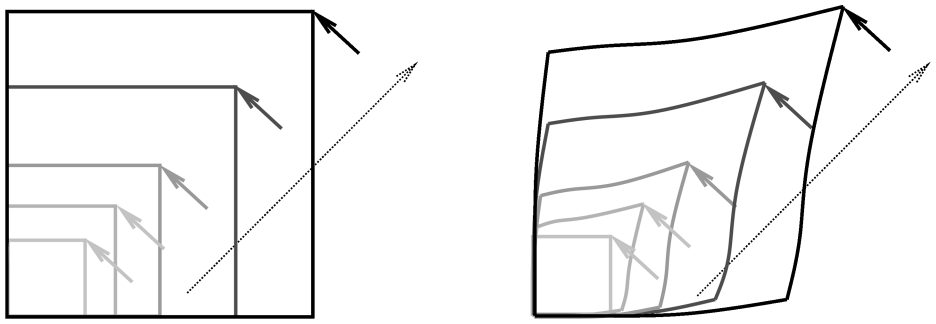

Fig. 7. Animating a stretching operation.

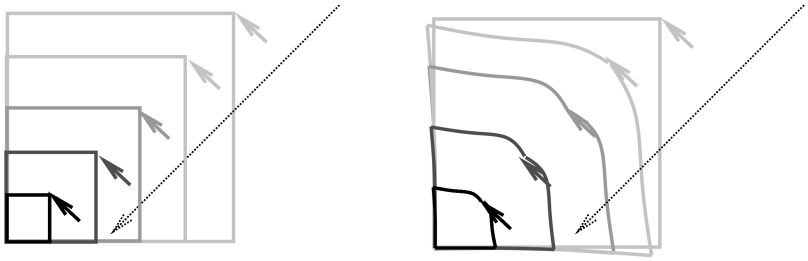

Fig. 8. Animating a shrinking operation.

Other shape-editing operations have a greater nonlocal effect on the object, and hence offer more opportunities for animation. For example, the operation to resize a rectangle by stretching it changes its width and height while still keeping it rectangular. This operation affects three vertices and all sides of the object. The regular nature of rectangles, however, allows for an extension of the basic animation described in the previous sections.

Figures 7 and 8 show the animation of a rectangle resize operation. The left diagram in both figures shows a nonanimated version of the rectangle growing and shrinking. The right sides of the figures show animated versions. Note how the vertices adjacent to the grasped vertex also participate in the action. When the rectangle is growing in size, they are drawn in slightly; when it is shrinking, they are pushed out. These animations, which correspond to the "stretch and squash" principle of cartoon animation, suggest the intuitive idea of the conservation of mass; if you press on an elastic object at one place, it will bulge out somewhere else. The overall effect strongly conveys the idea that the object's natural shape, a rectangle, is not being changed by the editing operation.

4. ANTICIPATING THE RESULT OF A MANIPULATION

The concept of anticipation is fundamental to the animation world as it is for static drawings. Imagine how a cartoon artist might depict a young boy fighting to raise a mouthful of food where the plate of food has a mind of its own and refuses to move. The boy might pull the fork to his mouth, but the drawing would lead the viewer to believe that the fork would snap suddenly back to the plate

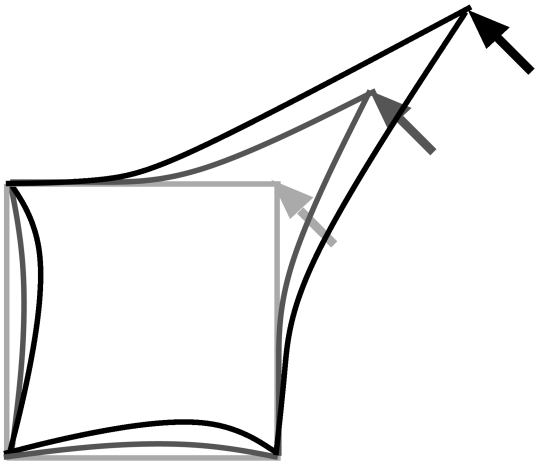

Fig. 9. Animating an attempt to move a pinned object.

if released. To help the viewer visualize the forces within the scene, the boy's arm would be bent to indicate his struggle, and the food would look stretched, indicating its tension. The effects combine to provide the viewer with a sense of anticipation as to the outcome of letting go of the fork. Similar effects can provide additional information and compelling visual cues to enhance graphical user interfaces. This section presents the animation effects to support visualization of the constraints inherent in the pinning, snap and drag, and indirect manipulation operations.

4.1 Pinning

Consider how an interface might portray an attempt to move an object that is fixed in place—that is, where the object is pinned. One response to this attempt might simply be to prevent the object from following the mouse. However, this lack of visible feedback might be misinterpreted as the result of a failure to "grasp" the object correctly. A user might make several attempts at the operation before realising the true cause of the lack of response. Another strategy might be to allow the object to follow the mouse, but then to snap it back to its original place when released. This approach avoids the problem with lack of feedback, but can lead to surprises when a carefully placed object suddenly jumps back to a previous position.

Figure 9 shows an animation effect that avoids both problems. As the user attempts to drag the pinned object, the grasped point stays attached to the mouse but the bulk of the object stays fixed. The effect is as though the user is pulling on a corner of an object that is anchored in place. The feedback provides extra information; it makes it clear that the user is attempting to move the object, but that the attempt will not succeed. When the grasped point is released, the object will spring back to its original shape. This pinning effect supports the principle of anticipation.

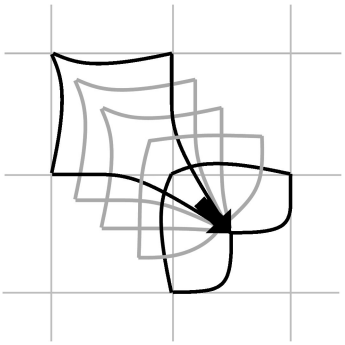

Fig. 10. Animating a move under the influence of "gravity."

4.2 Snap and Drag

Snapping is a technique commonly used as an aid for accurate positioning in direct manipulation systems. Key features on the object being manipulated are snapped to nearby "hot spots" such as other objects or regularly spaced grids. Graphical feedback in systems that use snapping is difficult to implement without animation due to the tension between the position constraint implied by the snapping action and the need to accurately track user input. One possibility is to prevent the object from moving at all until it is dragged sufficiently far from a grid point, then to have it suddenly snap to the next point. However, with this scheme, users may once again be uncertain whether the object has been "grasped." Another possibility is to allow the object to move freely, but to make it snap to the nearest grid point when let go. However, this scheme, may lead to surprises when an object unexpectedly jumps when released.

The use of the pinning animation effect enables us to avoid both of these pitfalls. For example, Figure 10 shows the effect of moving an object under the influence of a gravity field that causes the object to snap from the upper-left grid position to the lower-right position. As the user pulls the object away from a grid point, the grasped corner stretches while the remaining vertices of the object stay fixed. When pulled beyond a certain distance, the gravity suddenly "lets go" and the object snaps to the next grid point. The feedback makes it clear what the current state of the interaction is, what the user must do to achieve the desired result, and what will happen if the object is released at any time.

4.3 Indirect Manipulation

The visual changes caused by direct manipulation are easy for users to understand because the user focuses directly on the affected object and tracks the change as it occurs. But for changes caused by indirect manipulation, where the user's attention is focused elsewhere and the change is instantaneous, the effect can be confusing. Sudden visual changes can be disruptive because the user must subconsciously digest the new state of the screen; it takes a brief but significant moment before the new state is assimilated and the user can proceed with another task. For example, Figure 11 shows an operation where objects have been rearranged so that they abut left-to-right. By inspection, it

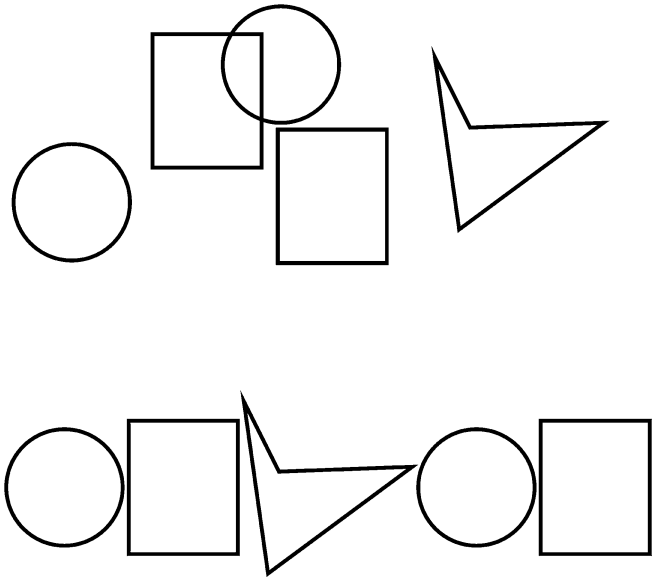

Fig. 11. Before and after an alignment operation.

is not obvious which objects in the initial configuration (top diagram) correspond to the objects in the final configuration (bottom diagram.) If the screen instantaneously changes from one configuration to the other, the user may well be confused. Moreover, if the initial and final state of the screen are similar, it may not be obvious just what has changed or how the change came about. This problem is exhibited in an operation that rotates a square through a quarter turn or flips it about an axis of symmetry.

The approach we have taken to smooth a change caused by indirect manipulation is to consider what the operation would look like if it was carried out by direct manipulation [Thomas and Calder 1996], albeit less quickly and precisely. For example, if an indirect manipulation causes an object to move to align with another object, then the effect portrays the change as if the object was dragged by one of its corners. As with direct manipulation, the object is distorted to suggest that it is slightly rubbery and is somewhat reluctant to change.

Figure 12 shows a situation where the square is moving to the left so that it is aligned with the circle. In use, the distortion of the manipulated object gives the impression that the object is being towed along by its corner but is reluctant to change. Without the distortion, the impression is as though the object is moving spontaneously. The net result is that the distortion helps convey the impression that the action, although indirect, is still under the control of the user.

When the user performs a complex operation, the final configuration of the objects is often not quite what was intended; the user might make several attempts before achieving the desired effect. One way to deal with such operations is to adopt a trial-and-error approach, with a heavy reliance on an "undo" facility; if the final result is incorrect, it can be undone and a new attempt made. At best, such a working style is cumbersome. Instead, we use animation to show

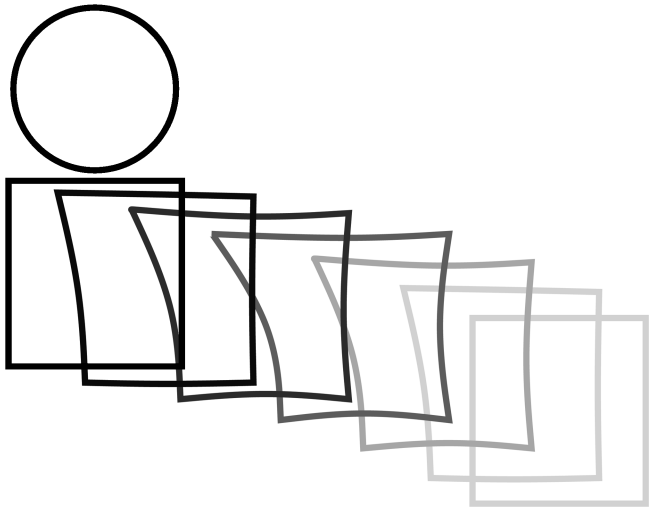

Fig. 12. Animating an alignment operation.

the effect of an operation while it is being considered; the animation suggests both what would happen if the operation is committed and what would happen if it is cancelled. By showing both states, this effect reduces cognitive load and minimizes the use of short-term memory.

The visual feedback to display the above approach uses an extension of the pinning technique previously used to show the effect of constraints on objects. In order to suggest that an operation has not yet been completed, the object is shown in the position it will adopt if the change is committed, but with a *telltale* corner stretched back to its original position, as if it were attached by an elastic string. The effect suggests both possible outcomes to the trial operation:

(1) If the change is committed, the telltale will retract into the object, showing that the new position has been accepted.

(2) If the change is cancelled, the object will be released and the telltale will drag it back to its original position.

Figure 13 illustrates the idea in the context of an alignment operation, where the square is moving to the left to align with the circle. The figure shows the situation at the point where the animation for the trial operation has been completed, but before the operation has been committed. The position of the square clearly shows what the final alignment would look like, but its top-right corner is stretched back to its original position to show that the object has not yet released its hold on that place. If the operation is committed, the stretched telltale retracts, leaving the square correctly aligned. But if the operation is cancelled, the square springs back to its original position.

The potential benefits of these techniques is twofold: the "liveness" of the objects helps strengthen the sense of direct manipulation that the interface conveys, thus making the interface more satisfying; and animating the effect of the user's actions on objects tells more about the process of the interaction

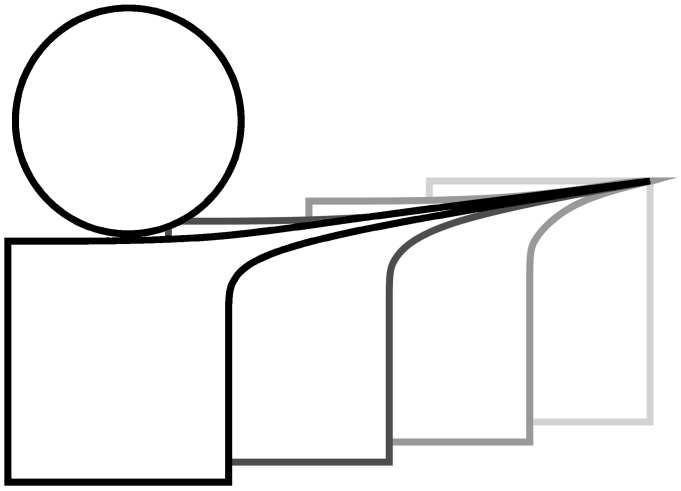

Fig. 13. Previewing an alignment operation.

and its outcome, improving the user's understanding of the effect of operations and leading to fewer surprises and mistakes. Furthermore, a recent study has shown that telltales increased users' recall of the previous positions of objects after an alignment operation [Thomas and Demczuk 2000].

5. EXPERIMENTS

To demonstrate our ideas, we built a drawing editor, named *adraw*, that uses cartoon-style techniques to animate the objects the editor manipulates. Adraw can be used to create and edit drawings made up of graphical objects like text, images, splines, rectangles, and polygons. The design of the editor is based on the drawing editor idraw [Vlissides 1990; Vlissides and Linton 1990], which is part of the InterViews toolkit [Linton et al. 1989]. Figure 14 shows the editor's interface.

This section describes two experiments conducted with the editor to test the benefits of using animation in the user interface [Thomas et al. 1998]. The first experiment was designed to test the effectiveness of the animations in providing feedback about the process of direct manipulation movement operations. Participants were asked to undertake a task with different forms of animated visual feedback. The measure of effectiveness was the time required to successfully complete the operation. The second experiment was designed to measure users' preferences for a range of animation effects. Participants were encouraged to adjust the amount of animation to their liking while performing a range of editing tasks; the different settings they chose were then recorded.

5.1 Experiment 1: Feedback Animation for Snap and Drag

The first experiment tested the idea that animation can provide information about the process of an interaction. This is particularly evident when an operation's effect on an object is subject to the presence of constraints. Section 4.2 describes how animation can provide feedback that suggests both

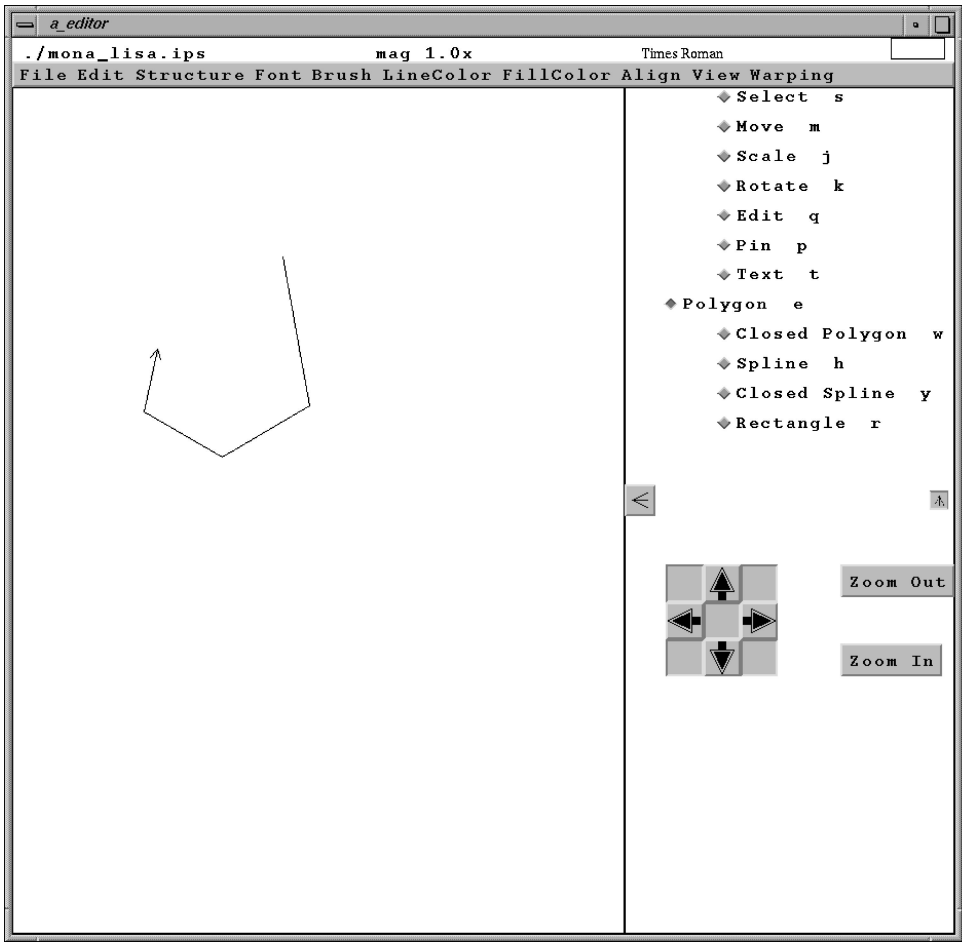

Fig. 14. The adraw main window.

the immediate affect of the user's action and the final outcome of the operation when the constraint is taken into account.

In adraw, the snap and drag constraint is provided for objects moving under the influence of a "gravity" field, which constrains the object's position to lie on a regular grid. To test the effectiveness of animations that suggest constraints, the editor was modified to use one of four feedback techniques for movement operations, as illustrated in Figure 15. These four feedback techniques are as follows:

(1) **no feedback:** Objects being moved show no visible feedback at all; they merely follow the mouse if unconstrained, and snap suddenly to their allowable positions if under gravity.

(2) **handles only:** Objects being moved show "handles" (small squares at the corners of the object) to indicate that the object has been selected, but the position of the object is as above.

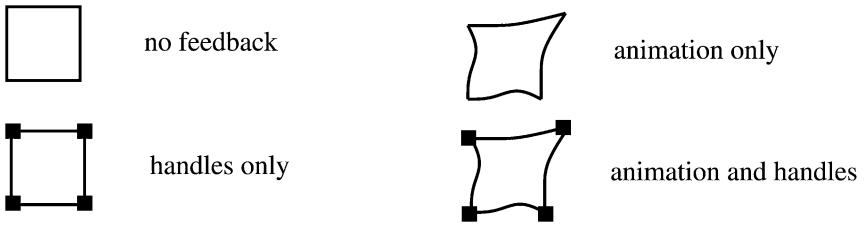

Fig. 15. The four visual feedback effects for movement.

(3) **animation only:** Objects are animated so that the corner being dragged follows the mouse while the remainder of the object lags behind; if an object is constrained by gravity, it stretches to follow the mouse, then snaps to its new position.

(4) **animation and handles:** Objects being moved show both handles and animation effects.

The experiment asked users to perform a simple rearrangement task under the influence of a gravity field. The experiment measured the users' performances by recording how long they took to complete the task and by noting instances of "misses" and "drops." A miss was defined as an attempt to move an object that failed to pick the intended target. A drop was defined as an incorrect release of an unmoved object, presumably because the user mistook the object's lack of observable response for a failure to pick it. The supervisor also asked users to complete an exit survey on their attitude to the various forms of feedback.

The experiment tested the following hypotheses:

(1) Cartoon-style feedback animation decreases task completion time and reduces miss and drop rates when compared with feedback that does not use such animation.

(2) Users prefer cartoon-style feedback animation over nonanimated feedback.

5.1.1 *Experimental Design.* The experimental task was to rearrange the objects from the configuration shown in Figure 16 on the left to that shown on the right.

Participants were tested individually, working on a Silicon Graphics Indy workstation in a quiet isolated office. The experiment supervisor read from a script to explain the requirements of the experiment and gave the following instructions:

(1) given the graphical objects in the starting formation, move the objects to the finishing formation, as indicated on a paper diagram;

(2) perform the task as accurately and quickly as possible;

(3) the object labelled "1" will remain in the same location;

(4) objects can be "picked up" only by their outlines; and

(5) the grid is turned on; so the objects can only be placed at the grid points.

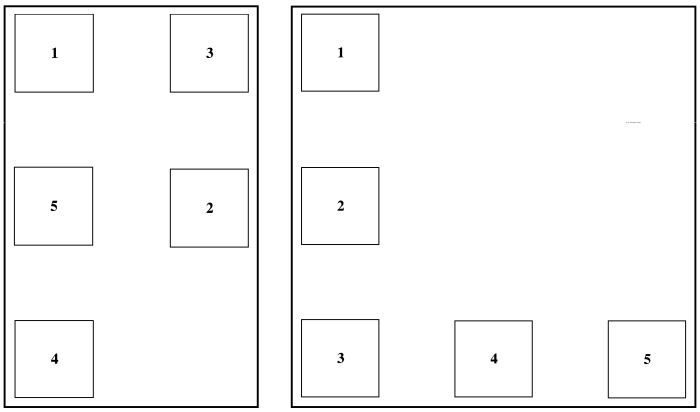

Fig. 16. Example of the starting and finishing configurations for the rearrangement trials.

The participants were asked to read an information sheet and a set of instructions, then asked if they had any questions. Each participant was then asked to sign a consent form.

To familiarize themselves with the editor and the feedback effects, each participant performed four trials of the editing task, one for each feedback effect. Then each participant performed five sets of trials, each consisting of ten repetitions of the task for each of the four feedback effects. Thus, each participant performed 50 repetitions of the task for each effect, or a total of 200 repetitions. To minimize ordering artifacts, the feedback effects were presented in a random order within each trial set. Participants were given a one minute rest break between each trial set. The experiment took about one hour for each participant to complete.

The editor was instrumented to record, at each press and release of the mouse button, the coordinates of the mouse and the time. In addition, the editor recorded at each mouse press the object that was picked (if any) by a mouse press, and at each mouse release the new grid point to which an object had been moved (if any). This information allowed for analysis of both interparticipant and intraparticipant effects.

The exit survey asked participants to indicate their attitude towards the various feedback effects on a 7-point scale, with 1 being "strongly liked" and 7 "strongly disliked."

The experiment was completed by 17 participants (mean age 27 years, age range 19–42 years). Participants were self-selected (on a first-come basis) from a pool of computer-literate students drawn from the undergraduate and postgraduate programs of the School of Computer and Information Science at the University of South Australia.

5.1.2 *Results and Discussion.* Initial analysis of the experimental data focused on task completion times. Analysis was carried out to compare the

completion times for the four feedback effects and to compare the times between the five trial sets.

An ANOVA analysis (Tukey HSD test, SYSTAT for Windows version 5) showed no significant difference in execution time between the four feedback conditions. However, a significant difference was found between the time for the first trial and the second trial ($p = .005$), and between the first trial and trials 3, 4, and 5 ($p < .001$). This difference was most likely due to a learning effect: that is, the participants took 1 set of 40 repetitions to stabilize their performance. The participants showed no significant difference in execution time between trials 2, 3, 4, and 5.

For the attitude survey, the participants showed a significant affinity ($p < .001$) for the three feedback conditions ($mean = 2.97$), compared to the no feedback condition ($mean = 5.5$). The participants showed a greater affinity for the combined feedback of handles plus warp ($mean = 2.4$) than for either the handles alone ($mean = 3.1$) or warp alone ($mean = 3.4$), although the difference was not significant.

The lack of difference between the completion times for the four feedback effects is surprising, but may reflect the simplicity of the task. Perhaps participants quickly learned to ignore the feedback (or lack of it) through performing the same task many times.

5.2 Experiment 2: Level of Animation Preference

To gauge users' affinity for the animations, a slider that allowed users to smoothly vary the amount of animation was added to adraw. At its minimum animation setting (level 0), the editor behaved like a conventional (unanimated) editor; at its maximum animation setting (level 20), manipulated objects were extensively distorted. A setting of 1.0 or more produced animation effects that were clearly evident during manipulation operations. For example, the diagram in Figure 3 shows a move operation with an animation setting of 1.0. Subjectively, the effect was as if the "weight" and "stiffness" of the objects varied: at low animation settings objects seemed light and rigid; at high settings they seemed heavy and rubbery.

The experiment encouraged users to adjust the amount of animation in the editor while performing a range of editing tasks. The changes users made to the animating setting were recorded as the trial proceeded. To determine user-interface satisfaction, the supervisor also asked users to complete a questionnaire about their attitude towards the editor and the animation.

The experiment tested the following hypotheses:

(1) When given a choice, users choose an animation setting that corresponds to a clearly evident level of animation.

(2) Users prefer animated interfaces over nonanimated interfaces.

5.2.1 *Experimental Design.* The experimental task comprised the construction of three drawings of increasing complexity. Figures 17, 18, and 19 show the completed drawings for tasks one, two, and three. The remainder of this section describes how the experiment proceeded.

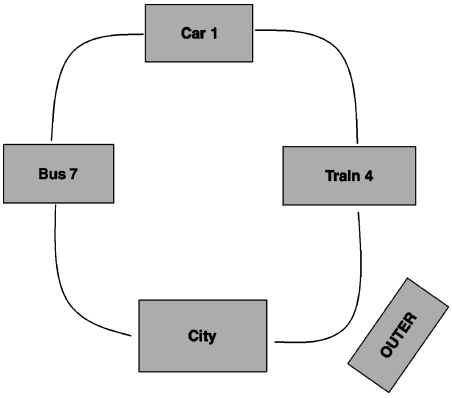

Fig. 17. The first drawing task.

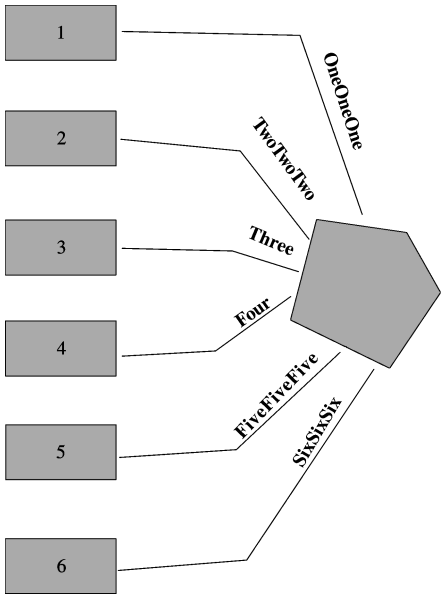

Fig. 18. The second drawing task.

As in experiment one, participants were tested individually, working on a Silicon Graphics Indy workstation in a quiet office. The experiment supervisor read from a script to explain the requirements of the experiment. Participants were asked to read an information sheet, then asked if they had any questions. The participants were then asked to sign a consent form.

Participants were told that they were to construct the three target drawings, which were presented to the users as paper handouts. They were also informed that they could adjust the amount of animation whenever they desired. At the start of each drawing task, the editor was set to zero animation, and the participants were asked to construct and manipulate a simple rectangle to experiment with the effect of different animation settings. When they were happy with the

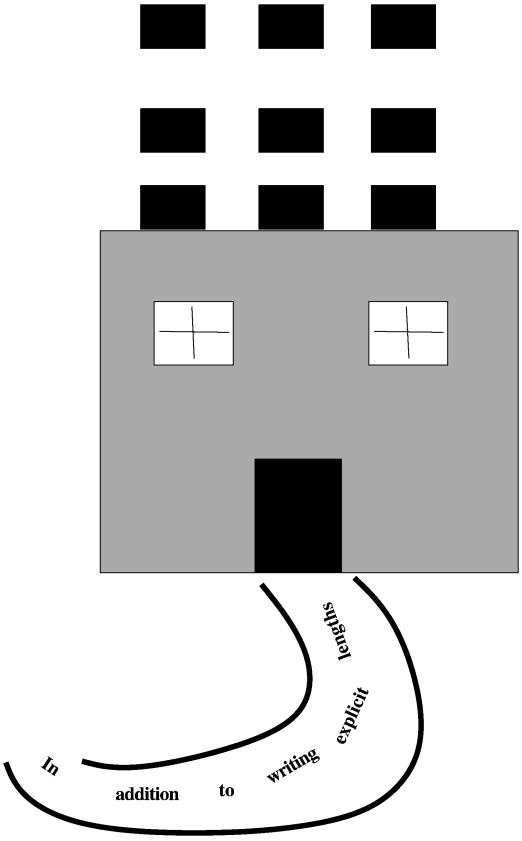

Fig. 19. The third drawing task.

effect, they were to delete the rectangle and proceed to construct the drawing. Participants were given a two-minute rest break between drawings. The experiment took about one hour for each participant to complete.

The editor was instrumented to record changes in the amount of animation, and the time at which the change occurred. This information mapped how participants adjusted the amount of animation over time, both at the beginning of a drawing task and during the execution of the task.

The questionnaire for user-interface satisfaction asked the participants about their impressions of the animation effects and the overall drawing editor. Participants recorded their responses on a 7-point scale, with 1 corresponding to negative reactions ("terrible," "frustrating," "dull," "difficult") and 7 corresponding to positive reactions ("wonderful," "satisfying," "stimulating," "easy"). Another set of questions asked participants to agree or disagree with statements about the animation.

Data was collected during the experiment from 13 different participants (mean age 27 years, age range 19–42 years). On average, participants had 12 years computing experience, including an average of 7 years experience, with mouse-based interfaces and a current average of 24-hours per week using such

Table I. Participant Animation Preferences for the Drawing Tasks

Participant	Drawing 1			Drawing 2			Drawing 3		
	max	*final*	*time*	*max*	*final*	*time*	*max*	*final*	*time*
1	4.5	2.1	20	14.9	2.7	14	3.2	3.2	1
2	19.0	4.3	73	4.0	4.0	67	5.0	5.0	69
3	10.7	4.8	61	19.0	4.9	42	4.8	4.8	2
4	10.5	6.0	42	5.4	5.4	2	5.2	5.2	1
5	19.0	0.9	46	4.3	1.9	33	2.0	1.8	23
6	5.0	2.3	35	2.1	2.1	57	2.2	2.2	3
7	19.0	2.5	33	4.8	3.2	16	19.0	3.0	13
8	11.5	7.5	72	2.9	2.9	1	2.9	2.9	7
9	9.3	0.0	54	8.0	0.0	59	6.3	0.0	44
10	19.0	3.5	84	3.0	0.8	8	1.2	1.2	1
11	11.0	6.3	69	7.2	2.8	26	6.4	5.4	1
12	11.4	2.7	37	4.7	4.7	1	2.4	2.4	1
13	20.0	1.5	97	2.0	1.3	89	4.0	0.9	39

interfaces. Participants were self-selected (on a first-come basis) from a pool of computer-literate students drawn from the undergraduate and postgraduate programs of the School of Computer and Information Science at the University of South Australia.

5.2.2 *Results and Discussion.* Initial analysis of the experimental results showed that all participants chose to experiment with the animation level at the beginning of the trial, most exercising the editor over the full range of animation settings. The usual pattern was for a participant to adjust the setting, then spend some time manipulating objects to get a feel for the animation at that level. Most participants tried several different settings before adopting a final setting and proceeding to the editing task. Some made further adjustments during the task or at the end; others left the animation setting alone after the initial experimentation.

Table I summarises the participants' preferences. For each drawing task, the table records the highest animation value the participant experimented with (max), the final value they settled on before beginning the task (final), and the amount of time (in seconds) that they spent adjusting the animation at the beginning of the task (time).

The data shows that, although each drawing task required the participant to reset the animation level (each task began with the level set to its minimum value), most chose to set the animation for each task to a similar level. Moreover, in the second and third tasks, the participants tended to spend less time experimenting with the animation. Indeed, three participants in task 2 and 7 in task 3 merely set the animation to about the same level as before, and proceeded straight to the drawing task, although others spent about the same amount of experimentation time for each task.

The data also shows that all but one participant (participant 9) chose to set the animation level to a value above zero. (More detailed analysis showed that participant 9 completed part of the drawing task with the animation set at around 1.0). The average final animation setting over all participants and drawing tasks was 3.0 ($minimum = 0.0$, $maximum = 7.5$, $sd = 2.1$).

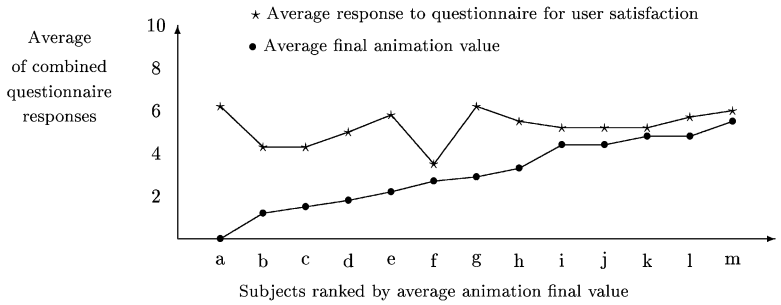

Fig. 20. Graph of total *final* values and six questionnaire values.

The attitude questionnaire showed that participants had a positive response to both the animation ($mean = 5.2, sd = 1.0$) and the editor as a whole ($mean = 4.9, sd = 1.0$). The more positive response to the animation compared with the editor (significant at $p = .001$) suggests that the animation itself was a contributing factor to the participants' rating of the editor. Of various suggested characterizations of the possible benefits of animation in drawing editors, participants rated it as "fun to use" ($mean = 6.3, sd = 0.9$) and "intuitive to use" ($mean = 5.5, sd = 0.9$). They were ambivalent about whether it improved productivity ($mean = 4.3, sd = 2.0$) and slightly positive that it encouraged them to work ($mean = 4.9, sd = 1.0$). The participants agreed that "the manipulation of objects seemed smoother" ($mean = 5.2, sd = 1.5$) and that "the manipulated objects seemed to have substance" ($mean = 5.3, sd = 1.1$).

Further investigation of data shows no strong correlation between the level of final animation setting and the responses to the questionnaire. The graph in Figure 20 shows the average of the participant's final animation values (marked with bullets) and their average responses to the questionnaire (marked with stars). The average final animation value is an indication of where the participants found an animation level that appealed to them; this ranged from 0 to 16.6, but all participants' average responses to the questionnaire were above 4.0 (where 3.5 is the mean of the scale). It seems animation appealed to all the participants, although the level and kind of animation differed between participants. In the main, participants who set the animation to a high level responded about as positively as those who set it to a low level. Even the participant who set the animation level to zero was generally positive about the animation effects.

5.2.3 *Users Comments.* The questionnaire also asked participants to make comments on the system and the animation effects. Some comments related to what the participants liked about the animation effects:

—it gives the objects a "feel";

—objects seem "softer";

—[animation] makes the work you are doing look fun. [It] makes things more interesting;

—it was easy to see what your action was doing when you move the object—it warps but still keeps a recognizable shape;

—good drawing tool. I enjoyed playing around with it;

—[animation] gives a sense of movement and speed;

—it seems to be really good for the text rotations;

—It can give the user more information ([as when] using gravity).

Other comments related to how other applications could be improved with animation:

—objects could be animated when they are pushed against other objects;

—animation could perhaps be used for fall-down menus or fold-out menus;

—animation could be used to show a "peeling off" action when objects are duplicated (currently, duplicates just appear);

—modifications to formatting of text [could be animated] to give a flowing effect; At the moment, the text changes too rapidly in WYSIWYG editors.

Several participants commented that "Shape distortion made placing objects slightly more difficult." A modification to the system could be to turn animation off when the object is moving slowly. A few participants made the verbal comment that they would have liked separate control of the animation for different operations. For example, the animation for the rotation could be set to a different level than the animation for translation.

6. CONCLUSION

This work makes three key contributions in the context of animating direct manipulation graphical interfaces:

(1) it describes a set of animation effects based on those used by cartoon animators that extend the visual feedback for direct manipulation interfaces;

(2) it shows the effects to be effective and enjoyable for users;

(3) it shows that convincing animation effects can be effectively incorporated into a significant application, a drawing editor.

Animations must be used to direct or focus the user's attention to key activities in the user interface of an application. However, inappropriate use of animation will merely distract the user by drawing attention to the animation itself rather than the task at hand. The effects of animation on the user's perception of the interface (like the effects of other aesthetic elements such as color) can be profound. On the one hand, this influence suggests that great improvements are possible; on the other hand, it warns that equally great disasters can happen. Just as there are good and bad uses of color, so there are good and bad uses of animation. Inappropriately applied, animation will seem childish and drive users away. But sensibly applied, it can make an interface more graceful and enjoyable to use.

The use of animation to highlight constraints in an application seems to be a fruitful area to investigate. One additional area we investigated was a

structured graphical editor for the mapping of logical parallel processes onto a physical parallel architecture [Trichina et al. 1997]. An application like this has many constraints, and we believe animation will provide better visual cues to the user. We believe the incorporation of our animation effects would improve the interface of any application that involves direct manipulation of objects subject to many constraints, such as a CAD system for architectural design.

REFERENCES

BAECKER, R. AND SMALL, I. 1990. Animation at the interface. In *The Art of Human-Computer Interface Design*. B. Laurel Ed., Addison-Wesley, Reading, MA.

CARD, S. K., ROBERTSON, G. G., AND MACKINLAY, J. D. 1991. The information visualizer, an information workspace. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, Information Visualization. ACM, New York, NY, 181–188.

CARD, S. K., ROBERTSON, G. G., AND YORK, W. 1996. The webbook and the web forager: An information workspace for the world-wide web. In *Proceedings of the ACM CHI'96 Conference on Human Factors in Computing Systems* (Vancouver, Canada, April). ACM, New York, NY, 111–117.

CHANG, B.-W. AND UNGAR, D. 1993. Animation: From cartoons to the user interface. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Animation/Visualization. ACM, New York, NY, 45–55.

CHATTY, S. 1992. Defining the dynamic behavior of animated interfaces. In *Engineering for Human-Computer Interaction. IFIP TC2/WG2.7 Working Conference*, Vol. A-18 (Ellivuori, Finland, 10–14 Aug.), 95–111.

CHATTY, S. AND BEAUDOUIN-LAFON, M. 1992. Integrating animation with interfaces. In *Proceedings of the ACM CHI'92 Conference on Human Factors in Computing Systems—Posters and Short Talks*. ACM, New York, NY, 70.

DONSKOY, M. AND KAPTELININ, V. 1997. Window navigation with and without animation: a comparison of scroll bars, zoom, and fisheye view. In *Proceedings of the ACM CHI'97 Conference on Human Factors in Computing Systems*, S. Pemberton Ed., Volume Extended Abstracts (March), ACM, New York, NY, 279–280.

GONZALEZ, C. 1996. Does animation in user interfaces improve decision making. In *Proceedings of the ACM CHI'96 Conference on Human Factors in Computing Systems* (Vancouver, Canada, April). ACM, New York, NY, 27–34.

HUDSON, S. E. AND STASKO, J. T. 1993. Animation support in a user interface toolkit: Flexible, robust and reusable abstractions. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Animation/Visualization. ACM, New York, NY, 57–67.

LAUREL, B. 1991. *Computers as Theatre*. Addison-Wesley, Reading, MA.

LAYBOURNE, K. 1979. *The Animation Book*. Crown, New York, NY.

LINTON, M. A., VLISSIDES, J. M., AND CALDER, P. R. 1989. Composing user interfaces with InterViews. *IEEE Computer*, 8–22.

MACKINLAY, J. D., ROBERTSON, G. G., AND CARD, S. K. 1991. The perspective wall: Detail and context smoothly integrated. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*. ACM, New York, NY, 173–179.

ROBERTSON, G. G., CARD, S. K., AND MACKINLAY, J. D. 1989. The cognitive coprocessor architecture for interactive user interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 3D/Gesture (Nov.). ACM, New York, NY, 10–18.

ROBERTSON, G. G., MACKINLAY, J. D., AND CARD, S. K. 1991. Cone trees: Animated 3D visualizations of hierarchical information. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, Information Visualization. ACM, New York, NY, 189–194.

STASKO, J. T. 1991. Using direct manipulation to build algorithm animations by demonstration. In *Proceedings of the ACM CHI'91 Conference on Human Factors in Computing Systems*, Programming by Demonstration. ACM, New York, NY, 307–314.

SUKAVIRIYA, P. 1988. Dynamic construction of animated help from application context. In *Proceedings of the ACM SIGGRAPH Symposium on User Interface Software*. (Nov.). ACM, New York, NY, 190–202.

SUKAVIRIYA, P. AND FOLEY, J. D. 1990. Coupling a UI framework with automatic generation of context-sensitive animated help. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, Automatic Generation II (Nov.). ACM, New York, NY, 152–166.

THOMAS, B. H. 1998. Animating direct manipulation in human computer interfaces. Ph.D. dissertation, Dept. of Computer Science, The Flinders University of South Australia, Adelaide, South Australia.

THOMAS, B. H., CALDER, P., AND DEMCZUK, V. 1998. Experiments with animating direct manipulation in a drawing editor. In *ACSC'98—The 21st Australasian Computer Science Conference* (Perth, Australia, Feb.), 157–168.

THOMAS, B. H. AND CALDER, P. R. 1994. Using animation to enhance look and feel. Tech. Rep. CIS-94-014 (Sept.), School of Computer and Information Science, University of South Australia.

THOMAS, B. H. AND CALDER, P. R. 1995a. Animating direct manipulation interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, Nov.), ACM, New York, NY, 3–12.

THOMAS, B. H. AND CALDER, P. R. 1995b. Animating widgets in the InterViews toolkit. *Lecture Notes in Computer Science*, *Vol. 1015*. Springer Verlag, New York, NY, 169–175.

THOMAS, B. H. AND CALDER, P. R. 1996. Animating indirect manipulation in direct-manipulation editors. In *Proceedings of the Computer Human Interaction Special Interest Group of the Ergonomics Society of Australia, OZCHI'96* J. Grundy and M. Apperley Eds. (Hamilton, New Zealand, Nov.), Ergonomics Society of Australia, 184–188.

THOMAS, B. H. AND DEMCZUK, V. 2000. Evaluation of animation effects to improve indirect manipulation. In *Proceedings of the First Australasian User Interface Conference* (Canberra, Jan.), IEEE, New York, NY, 110–117.

THOMAS, F. AND JOHNSTON, O. 1984. *Disney Animation*: *The Illustion of Life*. Abbeville Press, New York, NY.

TRICHINA, E., THOMAS, B., AND OINONEN, J. 1997. Visualization of data dependency analysis in parallel program design. In *Proceedings of the 2nd International Conference on Multimedia Information Systems* (*ICMIS 97*, Motorola University, Schaumburg, IL, April).

VLISSIDES, J. M. 1990. Generalised graphical object editing. Ph.D. dissertation. Stanford University, Stanford, CA.

VLISSIDES, J. M. AND LINTON, M. A. 1990. Unidraw: A framework for building domain-specific graphical editors. *ACM Trans. Inf. Syst. 8*, 3 (July), 237–268.

Received September 2000; revised April 2001; accepted May 2001