Extending 2D Object Arrangement with Pressure-Sensitive Layering Cues

Philip L. Davidson, Jefferson Y. Han
Perceptive Pixel, Inc
111 8th Avenue, 16th Fl
New York, NY 10011
philipd@perceptivepixel.com

ABSTRACT

We demonstrate a pressure-sensitive depth sorting technique that extends standard two-dimensional (2D) manipulation techniques, particularly those used with multi-touch or multi-point controls. We combine this layering operation with a page-folding metaphor for more fluid interaction in applications requiring 2D sorting and layout.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.
General terms: Design, Human Factors, Algorithms

Keywords: Direct Manipulation, Multi-touch, Pressure, Sorting, Tilt

INTRODUCTION

Direct manipulation encourages free-form grouping and arrangement of objects, particularly when Rotate-Scale-Translate (RST) methods give immediate feedback for controlling four degrees of freedom at a user's fingertips. While many methods for RST have been explored, layering operations use simple rules such as 'always bring to the front', or are simply left out.

Fine control of object layering involves dedicated UI components. Traditional 2D editing programs supply either a drag-and-drop 'layer palette', displaying a list of the elements to be reordered, or a set of contextual operations (key command or menu) on a selected element to send it up, down, to the front, or to the back. These controls tend to respect global element order, rather than the relative order; thus, switching the order of two overlapping objects may involve a sequence of several commands. Often, the simplest way to reorder elements is often to send both objects "to the back" or "to the front" sequentially, potentially breaking the layering order for scene elements. Our approach uses pressure cues to directly control pairwise layering constraints in conjunction with two-dimensional transformation.

PREVIOUS WORK

Piling Interfaces

Mander et al. [8] use 'piles' to represent a folder as an intentionally untidy stack of icons. This was motivated by real-world document manipulation, where the uneven edges of a pile allow for a quick visual search of its content. Beaudouin-Lafon [2] introduces a number of novel window interactions, including 'loose' window collections, and demonstrate a folding gesture for a stack of documents (as in [8]), where a user 'pulls' the corner of a foreground element to reveal the contents of the element behind it. The 'Bumptop' environment described by Agarawala et al. [1] uses pen-based interaction to drag and toss document objects around into physically simulated piles. Ordering and layout actions are inherently possible, but are limited by side effects of the simulation environment. Instead, precise ordering gestures appear when objects are grouped into more formal groups or stacks. Terrenghi et al. [11] performed a user study of photograph sorting and puzzle manipulation on real tabletops in comparison to interaction patterns using a surface computing environment. While they did not strictly analyze layering operations, several interesting observations were made regarding the difference in physical and virtual interaction, even when both environments allowed for multi-point and bimanual control.

Layer Arrangement

Ramos et al. [10] present two novel techniques for 2D layering operations. The first provides a graphical representation of a cascaded stack of layers above the selected elements, and using a sequential drag and drop model. The second uses a 'splatter' effect to radially distribute overlapping. Concentric rings are then used as a proxy for rearranging object order. Dragicevic [5] describes several methods for 'leafing' through the exposed edges of a stack of windows in a drag-and-drop action, by sequentially folding back document windows, as in [2], as the cursor travels back and forth across the document edge. The work does not discuss pressure as an interaction toggle; instead, a speed-limit is used to distinguish the leafing behavior from ordinary cursor motion.

DESIGN CONSIDERATION FOR LAYERING TASKS

Pressure data provides useful sideband information with multi-point sensing, and tilt control uses position and pressure to generate a three degree-of-freedom control (normal direction and depth). Our tilt gesture uses a direct

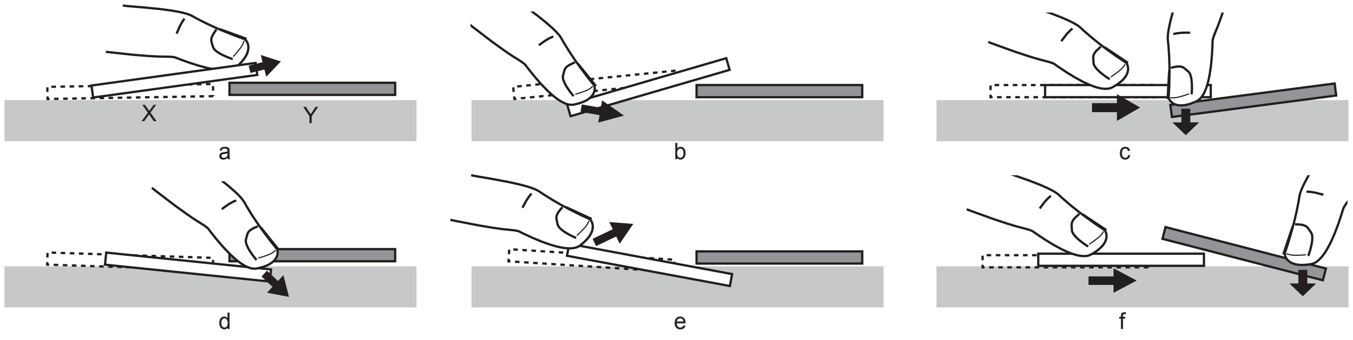

Figure 1. Common Layering Gestures

manipulation metaphor, but the tilt value is used as input to the layering algorithm, instead of imparting rotation.

Minimizing UI Detritus

One benefit of multi-point RST controls is that they require less decoration of an object – once a user learns the basics, no visual feedback is needed to indicate that an object is in "rotation mode" or "scaling mode" – they are inherent properties of the object. The absence of this modality encourages a simple transition between object manipulation and other control tasks, especially when operations may be shared between hands. Tilt conditions may be shown with shading or slight rotation, communicating orientation through secondary visual properties of the object, instead of tool marks, menus, or other adornments.

Pressure-Based Control

A number of studies of interaction with pressure-sensing devices demonstrate that our capacity for absolute estimation of pressure is quite poor, and is more appropriate as a rate control. It has been shown [4,9] that users are able to develop adequate precision with absolute pressure controls when presented with immediate visual feedback, either at the point of contact or on the control display. However, the effectiveness of visual feedback drops in the context of multi-touch environments, when the user must divide their attention between several pressure points, in addition to any other visual task. Gauging relative pressure between two or more points requires less effort, as long as the sensor is of adequate resolution, and consistent an continuous pressure measurement. A simple use of pressure information would be to estimate a singular depth value for each object. By representing pressure values as a tilt, we can localize the depth adjustment to a particular edge or corner, allowing more precise layer gestures in complex arrangements.

Relative vs. Absolute Layering

If precise placement of an object at a specific depth is not required, the use of absolute depth values is more a hindrance than a benefit. As an example, a alternate implementation allowed the user to slide objects in absolute space along the tilted plane. By tilting the element and sliding in the correct direction, objects could be shifted forward and back along the depth-axis. Casual experimentation revealed this to be impractical as a layering control, because an accurate estimate of both position and attitude were required to reach the target depth. The tilt of a edge or corners conveys the intended orientation of an element as it is moved relative to neighboring objects. For sorting operations in which the relative order elements is more critical than absolute depth, tilt alone is sufficient as a layering control.

INTERACTION MODEL

The elements of our system are displayed in a two-dimensional canvas where they can be freely repositioned using multi-touch RST techniques [3]. Atop this two-dimensional transformation, objects can be tilted into the plane by varying the pressure applied. The user can estimate the relative depth of adjacent elements from rendering cues and from physical pressure feedback, and thus infer which element will overlap the other. As elements are moved and rearranged into overlapping clusters, pairwise overlaps are used to maintain a consistent global ordering for all elements in the scene.

Figure 1 shows a side view of six canonical layering operations that can be performed on a pair of elements using the tilt layering semantics. For each of these cases, the grey element, Y, stays in a fixed position while the white element X is moved so that the two overlap. The top row shows instances where X is placed atop Y: in 1a the right edge of X is lifted and pulled on top of Y, in 1b the left edge of X is depressed to pivot the right edge up and pushed over Y, and in 1c, one finger lowers the left edge of Y and another slides X to the side. These may be respectively described as 'depress and pull,' 'wedge and push,' and 'raise and slide.' The second row of example demonstrates a similar gesture as above, except that the order X is now placed below Y. It should be noted that only 1c and 1f specifically show more than a single point of pressure contact.

Tilt calculation is performed in tandem with RST transformations, so each point constraint may influence one or both calculations, depending on the pressure applied. Because the tilt gesture and the orientation control may be performed from similar hand positions, the user can switch modes without the cognitive load of changing the pose of their hand or rearranging of their fingers on the object. This encourages the natural bimanual manipulation behaviors and staging patterns that occur in real-world arrangement tasks.

IMPLEMENTATION

Hardware

We use a multi-touch sensor described in [7], which provides an arbitrary number of position and pressure sensitive contact

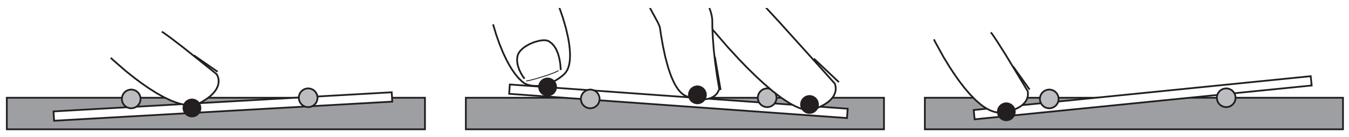

Figure 2. Single Point Tilt Example, Multi-point Tilt Example, 'Edge Pivoting' Example

points. Our implementation is written in C++ and OpenGL, and displays various desktop elements, such as standard polygons, free form shapes, photos, videos, and documents, running on a 2.4 GHz quad core PC with an nVidia GPU.

Tilt Calculation

We use the isometric tilt-plane formulation from [6], where pressure values indicate the depth of each contact point, and a best-fit plane is solved over all points touching an object. We select a depth mapping function such that light pressure corresponds to a positions above the surface, so that elements may be lifted above other objects at rest, while strong pressure maps deeper into the surface, with a slight deadband 'in plane'. As in [6], default constraints are placed to create a soft 'pivot ring' around the center of the object, so that a plane is well defined even with a single contact point. The ring of constraints are placed in a ring approximately 90% smaller than the inscribed circle of the convex hull. Pressure near the center pushes object the object down uniformly, pressure away from center tilts in that direction, and 'leverage' on the outer corners can raise the opposite edge of the object, as shown in Figure 2. This tilt calculation is easily combined with any position-based RST transformation, making control of each operation relatively separable.

Pairwise Overlap

The system maintains a directed acyclic graph (DAG) of overlapping elements, defined by pairwise 2D intersection tests. Any appropriate intersection routine may be used to detect overlap, provided that is symmetric and suitable for real-time performance. Each time intersection condition changes between objects A and B, the following rules are applied to determine their overlap order.

Transitive Overlap/Underlap Conditions: If A can be reached from B via a strict upward or downward traversal of the graph, then the overlap direction is already well-defined. This prevents cycles from appearing in the graph.

Tilt Comparison: If either A or B have tilt values, the relative depths of A and B are compared in their overlapping regions, using ID-buffer methods or geometric analysis. If A is consistently above or beneath B in all of these regions, then we update the edge between A and B. If multiple overlapping regions do not have a consistent ordering, the default order is selected. Audiovisual feedback may be provided to indicate that the ordering may not be what was intended.

Default Order: If neither of the two elements are tilted past the deadband threshold, or the overlap check does not return a consistent ordering, then we simply use existing layer values.

Global Ordering

Using connected component analysis on the overlap graph, the system groups elements in the scene into 'clusters'. An absolute ordering over all elements is valid as long as the relative ordering is consistent in each of these clusters. We calculate a total relative ordering in each cluster using depth-first-search traversal of the pairwise DAG, and re-order elements accordingly. Valid solutions are not unique, but they will be visually identical. Methods to better preserve global ordering constraints between elements will be explored in future work.

Rendering Cues

A combination of cues are used for indicating the relative tilt of an object to user. To start with, the slight out-of-plane rotation is applied correlating to the tilt value calculated by the system. This alters the visible outline of the object as pressure is applied. Hardware based depth attenuation (e.g., OpenGL fog) can be used to darken portions of the object as it is pushed into the surface, and if a lighting model is used, subtle shading effects can indicate direction. If an object is lifted above the plane, a drop shadow changes in offset and penumbral width as the object rises. These effects are shown in Figure 3.

Audio/Haptic Cues

Physical feedback is integral to real-world manipulation tasks. To add some measure of this to our system, we experimented with adding simple audio cues for arrangement, by playing short sound effects when overlap events occurred.

Occluded Content

Element to element layering operations are only useful when object edges are accessible. However, this is not generally the case as scenes become denser and more complex. Occluded elements can be revealed using traditional floating palette, or through spatially coherent gestures such as the 'Exposé' operation, or the 'tumble' and 'splatter' method described by [10]. These interface layers distort the context of the editing operation or occlude other relevant content. As demonstrated in [2], and extended by [5], page-folding and peeling provide a simple metaphor for searching through and revealing occluded content, while keeping with a paper

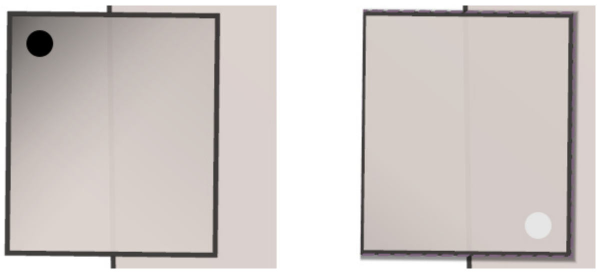

Figure 3: Tilt Shading, Drop Shadowing

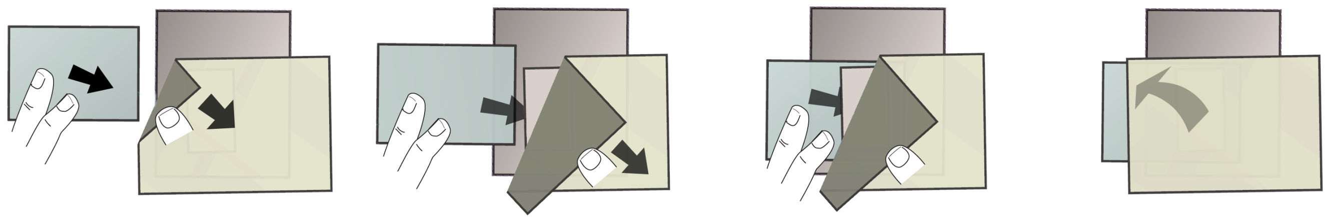
Figure 4: Use of a peel-back gesture to allow layering with occluded elements

metaphor. We implement a variant of this folding behavior using a pressure clutch for points that are dragged across the edge of an element. Figure 4 illustrates a short sequence of actions integrating the folding model with the layering operations. While peeling back the corner of the yellow rectangle with one finger, the blue rectangle can be moved right and inserted between the two elements that are layered behind it. The blue rectangle is dropped, the fold is released, and yellow element is restored.

Restoring 2D Orientation
The downside to providing layer control operations with 2D gestures is that it is ill-suited to cases where we wish to adjust the layer depth alone. In keeping with our goal of reducing extraneous UI elements to a minimum, we leave a proxy outline in the original location of the component. If an object is released in a similar position, it will snap to its former location.

FUTURE WORK
While the tilt gesture provides a well-defined planar models, more deformation methods could be applied, such as non-linear bending, or adding a permanently curled or folded corners. These would also provide for a variety of rendering cues to guide user interaction. The initial target for the layering operation work was restricted to strictly linear sequencing model. An interesting extension of the layering model would be to extend this to 2-1/2-dimensional representations, to allow for complex overlap effects such as self-overlap, or circular overlap patterns.

In current implementation, the layering model only considers active overlap relationships among elements in the scene. A straightforward extension of the overlap algorithm would be to extend this scheme by inserting prior overlap relationships as lower priority constraints, or allowing the user to 'freeze' layering relationships for groups of elements regardless of overlap state.

REFERENCES

1. Agarawala, A. and Balakrishnan, R. 2006. Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In *Proceedings of SIGCHI 2006,* pages 1283-1292. ACM Press, April 2006.

2 Beaudouin-Lafon, M. 2001. Novel interaction techniques for overlapping windows. In *Proceedings of UIST 2001*, pages 153-154. ACM Press, November 2001.

3 Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. 1993. Toolglass and magic lenses: the see-through interface. In *Proceeedings of SIGGRAPH 1993*, pages 73-80. ACM Press, 1993.

4 Buxton, W., Hill, R., and Rowley, P. 1985. Issues and techniques in touch-sensitive tablet input. In *Proceedings of SIGGRAPH 1985*, pp. 215-224. ACM Press, July 1985.

5 Dragicevic, P. 2004. Combining crossing-based and paper-based interaction paradigms for dragging and dropping between overlapping windows. In *Proceedings of UIST 2004*, pp. 193-196. ACM Press, October 2004.

6 Gingold, Y. I., Davidson, P. L., Han, J. Y., and Zorin, D. 2006. A direct texture placement and editing interface. In *Proceedings of UIST 2006*, pages 23-32. ACM Press, October 2006.

7 Han, J. Y. 2005. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of UIST 2005*, pages 115-118. ACM Press, October 2005.

8 Mander, R., Salomon, G., and Wong, Y. Y. 1992. A "pile" metaphor for supporting casual organization of information. In *Proceedings of SIGCHI 1992*, pages 627-634. ACM Press, May 1992.

9 Ramos, G., Boulos, M., and Balakrishnan, R. 2004. Pressure widgets. *Proceedings of SIGCHI 2004*, pages 487-494. ACM Press, April 2004.

10 Ramos, G., et al. 2006. Tumble! Splat! helping users access and manipulate occluded content in 2D drawings. In *Proceedings of AVI 2006*, pages 428-435. ACM Press, May 2006.

11 Terrenghi, L., Kirk, D., Sellen, A., and Izadi, S. 2007. Affordances for manipulation of physical versus digital media on interactive surfaces. In *Proceedings of SIGCH1 2007*, pages 1157-1166. ACM Press, April 2007.