The Vacuum: Facilitating the Manipulation of Distant Objects

Anastasia Bezerianos, Ravin Balakrishnan
Department of Computer Science
University of Toronto
anab | ravin @dgp.toronto.edu
www.dgp.toronto.edu

ABSTRACT
We present the design and evaluation of the *vacuum*, a new interaction technique that enables quick access to items on areas of a large display that are difficult for a user to reach without significant physical movement. The vacuum is a circular widget with a user controllable arc of influence that is centered at the widget's point of invocation and spans out to the edges of the display. Far away objects residing inside this influence arc are brought closer to the widget's centre in the form of proxies that can be manipulated in lieu of the original. We conducted two experiments which compare the vacuum to direct picking and an existing technique called drag-and-pick [2]. Results show that the vacuum outperforms existing techniques when selecting multiple targets in a sequence, performs similarly to existing techniques when selecting single targets located moderately far away, and slightly worse with single targets located very far away in the presence of distracter targets along the path.

ACM Classification: H.5.2 [User Interfaces]: Input devices and interaction techniques, Interaction styles
Keywords: Large displays, distance reaching

INTRODUCTION
Interaction with large displays has long been of interest to the research community, with much of the early research focusing on single whiteboard-sized displays [12, 14]. More recently, the rapidly decreasing cost of projectors has spurred research in the construction of much larger wall-sized displays by tiling multiple projectors to form a single virtual image [1, 5, 7, 15]. These multi-projector high resolution large displays are interesting from an interaction perspective in that they enable users to view high quality imagery even when they are up-close to the display. In contrast, single projector systems at that scale are not suitable for up-close interaction as the image appears too pixelated. If we are to use these displays in the interactive manner for which they are well suited, we need to address the interaction challenges that arise due to their ability to display vast quantities of data over a large spatial canvas.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CHI 2005, April 2–7, 2005, Portland, Oregon, USA.
Copyright 2005 ACM 1-58113-998-5/05/0004...$5.00.

Unlike interaction on a desktop or even a small whiteboard-sized display where almost all displayed items are within arm's reach of the user, data on wall-sized displays often reside farther away, or in an unreachable location. As a result, if existing user interfaces are mapped onto displays of this scale for up-close interaction, they would at the very least require the user to walk around the display to accomplish even simple tasks, or they may be unusable altogether when, for example, the user can't reach the top of the display to operate an application's menu bar [11]. Admittedly, one could always operate such a display from afar, using a mouse and a keyboard, but we believe that such an approach does not fully leverage the potential benefits that can accrue with up-close direct interaction.

In this paper we design and evaluate a new technique, called the *vacuum,* for selecting and manipulating objects in remote locations on a large-scale high resolution display. Prior to designing our technique, we first identified some desirable design principles:

Transient use: The technique should have a low, ideally zero, invocation and dismissal cost. For example, regular pointing and clicking on a target has little overhead since there is no tool to invoke or dismiss. In contrast, a modal selection mechanism would not be as transient.

Minimize physical movement: The technique should keep to a minimum the amount of traveling the user has to do in physical space to achieve any task.

Predictable and consistent. The technique should have a visual layout and behavior that is persistent over multiple invocations in similar situations.

Transparent: The technique should integrate seamlessly with the real tasks that users want to do, rather than present the user with a whole new way of doing things.

Flexible: It should be possible to modify the technique's parameters online with minimal overhead, and complex functionality should be supported when needed, but should not get in the way of basic functionality.

In the following sections, we review the related literature, identify the desirable properties and shortcomings of existing techniques, describe the design of the vacuum, and present two experiments that evaluate the vacuum's performance in single and multi target acquisition tasks.

RELATED WORK

Tivoli [14], an early application for whiteboard-sized displays, focused on content structuring for meeting tasks, but it did identify potential problems arising from directly applying existing interfaces to larger displays. Flatland [12] is an application for whiteboard displays that concentrates on content management. Guimbretière et al.[7], introduce techniques for content creation and placement tailored to large displays. Streitz et al. [17] describe techniques for connecting multiple displays together and moving objects between them. Hinckley et al. [9] describe pen-based techniques for connecting multiple displays together in various topologies and subsequently manipulating content on them as though they were a single shared workspace.

Researchers have also developed techniques that allow users to define and access alternative views of work areas. WinCuts [19] supports acquiring and interacting with alternative views of arbitrary regions of existing windows. Swaminathan and Sato's [18] dollhouse metaphor uses a small scale model of the display and its contents to specify pointer movement on a large display. Frisbees [10] are interactive portals for accessing remote screen locations. The above are powerful methods for manipulating remote content, with many adjustable parameters for tailoring the interaction at each invocation. They are particularly useful when users intend to focus on remote content for a period of time. However, the invocation and parameter adjustment overhead make these tools less suitable for lightweight, transient tasks, like simply selecting a few remote objects and then returning to work on nearby content.

There has been significant research in reaching across distances in large interaction surfaces in a lightweight and transient manner. Pick-and-drop [16] and take-and-put [6] provide ways of moving content from one location of a screen to another or to a different screen entirely. However, these are largely suited to scenarios where all areas of the interaction surface(s) can be reached freely. Shuffle [6] and flick [20] enable content to be sent quickly to a remote location, either covering a specified distance [6] or to the edge of the interactive surface [20]. The throwing technique [6] is similar but user configurable. These techniques are designed for approximate interaction, and it is unclear how they perform in precise positioning tasks. Hascoët's [8] throwing techniques, although precise, do not support multiple operations, as discussed later in this section.

Drag-and-pop and drag-and-pick [2]

The research most directly relevant to our present work, and from which we draw significant inspiration, is the drag-and-pop and drag-and-pick techniques developed by Baudisch et al. [2], and accordingly we discuss these in greater detail. Drag-and-pop/pick allows users to quickly drag icons to, or pick items from, objects located at far distances by bringing proxies of those items closer to the user. When the user pens-down and starts moving, potential targets within a ±30° (this size can be varied, but not interactively) sector centered about the pen's movement direction are identified and proxies of those targets are created close to the current cursor position. These proxies are used in lieu of the original targets to complete the drag or pick operation, significantly reducing the cursor's travel distance. To maintain visual persistence, rubber-band lines are drawn connecting actual target icons to their proxies. A user study [2] showed that drag-and-pop was up to three times faster than regular direct dragging across a large-scale display.

Unlike other techniques surveyed (e.g., [10, 18]), an extremely nice property of drag-and-pop/pick is that it is a very lightweight technique that seamlessly enhances regular dragging and picking, without the user having to learn a completely new interaction technique or set/adjust multiple parameters. Furthermore, it is a modeless and transient technique in that no tool has to be invoked or dismissed since the entire operation is performed in a single pen-down, move, pen-up gesture. While these properties make drag-and-pop/pick almost ideal for single target operations, there are some shortcomings with this approach:

- *Target identification:* The sector of influence is determined by the cursor's initial movement vector. If the user starts off in an inexact direction and the wrong targets are brought closer, the only option is to abort and restart the operation. Baudisch et al. [2] discuss adjustments to the algorithm to favor targets in certain directions, but fundamentally the technique is limited in not allowing users to alter the sector of influence during the interaction.

- *Number and size of proxies:* Proxies retain the size of the original targets, and to prevent clutter or overlapping proxies around the cursor, the number of proxies is capped at a fixed value (5-10 typically). While this approach is reasonable for the iconic targets studied in [2], it does not scale to situations of larger numbers of distracter proxies or larger sized targets (e.g., tool palettes), since the space around the cursor would quickly fill up.

- *Layout of proxies:* The final position of proxies around the cursor is dependent on the cursor's initial movement vector and on the number of proxies. The algorithm also removes much of the void space around the proxies in the final layout. As such, the proxy layout does not exactly resemble the original target layout, and it can be difficult for the user to predict where exactly the proxies will end up. In particular, if there are many more proxies than the 5-10 item limit imposed in the original design [2], due to space constraints it may be necessary to place some proxies on the other side of the cursor's movement direction. Cursor backtracking would thus be required to select those proxies.

- *Multiple operations:* All interaction is completed within a single pen-down, drag, pen-up operation. While this contributes significantly to the transient, lightweight, operation of the techniques, it is limiting in that operations that require several pen-up/down events for a single task cannot be supported. For example, it would be difficult to interact with a slider widget at a remote corner of the screen, or make multiple selections from a tool palette.

THE VACUUM

The vacuum is an interaction technique that enables quick access to items on areas of the display that are difficult to reach by bringing them closer to the user for viewing and manipulation. In its simplest form, the technique acts as a "vacuum cleaner", bringing toward it items that reside inside an arc of influence centered at its point of invocation and spanning the entire display. The design of the vacuum was driven by, and satisfies, the design principles outlined in the introduction. The end result is an interaction technique that retains the excellent properties of drag-and-pop/pick [2] including lightweight, transient use, while resolving the shortcomings of drag-and-pop/pick identified above.

Visual Design

In our current implementation, the vacuum is designed as a circular widget, with an inner bull's-eye centre and an arc of influence with a user controllable angle that extends from the centre along a user defined line to the limits of the display (Figure 1). Proxies of all objects within the arc's influence are brought close to the centre of the widget, within easy reach of the user, thus conforming to our *minimal physical movement* design principle. As with drag-and-pop/pick, we define a within-arm's-reach buffer zone around the widget such that only objects outside that zone are under the influence of the vacuum. This allows nearby objects to be manipulated without proxies, and reduces the total number of proxies that have to be brought to the user.

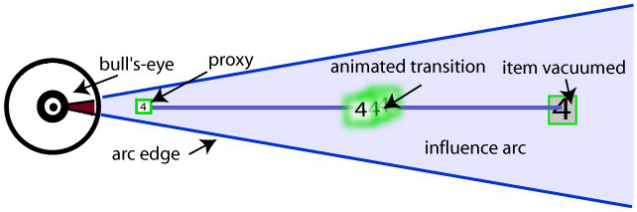

Figure 1. Vacuum.

When using drag-and-pop/pick in our early design explorations and pilot studies, we observed that users were not always aware of the sector of influence of the technique. Accordingly, in designing the vacuum, we ensured that appropriate visual feedback regarding the extents of the vacuum is always present in the form of the displayed arc edges and a faint semitransparent overlay indicating the vacuum's area of influence (Figure 1). Also, proxies are dynamically updated in a smooth animated fashion as the vacuum's extents are changed. This enables the user to easily comprehend the vacuum's actions, conforming to our *predictable and consistent* design principle.

Invocation and Adjustment

The vacuum is invoked after the user clicks and drags a distance beyond a fixed threshold (Figure 2a). The vacuum starts off with a 20° arc that extends along the movement axis of the cursor (Figure 2b). If the user continues dragging along the centreline of the arc away from the start position, the arc widens to a maximum of 120° (Figure 2c). Dragging back along the centre line towards the start position narrows the arc. If the user drags back through the start position and continues moving along another direction, the arc smoothly transitions to extending along the appropriate new direction (Figure 2d). Moving off the centreline towards either edge of the arc allows that edge to be dragged to any desired angle (Figure 2e,f). Thus, if the user initially started off in an incorrect direction, a simple adjustment allows for the desired targets to be captured. While the invocation of the vacuum is similar to drag-and-pop/pick, it differs in its operation in that the direction and extents of the arc of influence are fluidly adjustable by the user simply by dragging in the appropriate directions. This solves the *"target identification"* problem inherent in drag-and-pop/pick, and conforms to our *flexible* design principle.

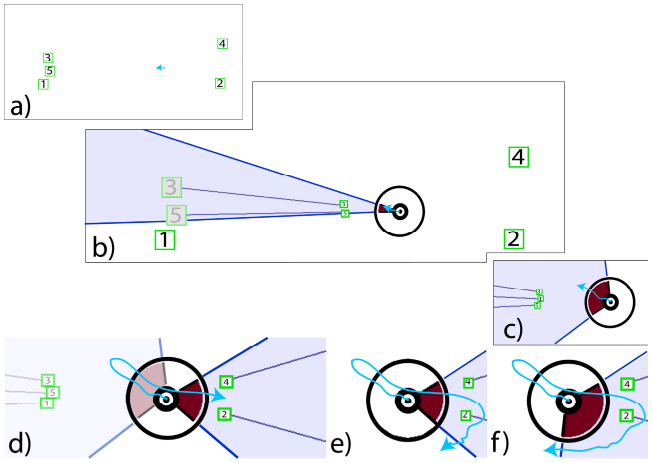

Figure 2. Vacuum invocation and adjustment. Blue arrow indicates cursor movement. (a) Cursor drag begins in centre of display. (b) When drag exceeds threshold, vacuum is invoked with 20° initial arc, bringing proxies of targets 3, 5 towards the centre. (c) Additional cursor movement increases arc angle. (d) Change in cursor direction changes direction of arc. (e, f) Cursor moving beyond the arc's edges expands the arc.

Interaction with Proxies

Users can interact with the proxies within the vacuum in lieu of, but in an identical manner to, the original objects. For example, if the user had started the drag by clicking on an object, releasing the drag on a proxy would be equivalent to dragging-and-dropping the original object on the target linked to the proxy, if such an action made sense in the given context. Functionally this is equivalent to drag-and-pop [2]. Alternatively, if the user had started the drag by clicking on empty space, releasing the drag on a proxy would be equivalent to selecting the target linked to the proxy. Functionally, this is equivalent to drag-and-pick [2]. As with drag-and-drop/pick, the vacuum conforms nicely to our *transient* and *transparent* design principles in that the invocation and operation cost of the vacuum is no different from a regular drag-and-drop or pick operation and would work across any application seamlessly.

Layout of Proxies

An important aspect of the design of the vacuum is in the representation and layout of the proxy items. The vacuum can collect many items from a large area of the screen, but must display copies of the items within the much smaller region near the tool's centre. For even a moderate number of items, it is impossible to display these copies at full size near the centre without significant overlap in layout. In an initial design, we explored a full-size proxy with stacking layout, where the stacked proxies would spread out when the cursor was in proximity. However, it was difficult for users to determine exactly where to position the cursor to select a particular proxy since it wasn't obvious how the stack of proxies would expand, thus violating our *predictable and consistent* design principle.

In the layout we decided upon, called the *scaling vacuum,* proxies of the vacuumed items are scaled down in size and displayed around the tool's centre, preserving the relative spatial relationships of items to one another. As items enter the vacuum's influence, we animate the movement of a copy toward the tool's centre, while a ghost image of the item remains at its original position. When an item leaves the vacuum's influence, the reverse animation takes place. This approach allows the user to maintain a sense of overall context of where everything is on screen. A semitransparent line connects the centers of the proxies to the centre of the original items. This line passes virtually through the center of the tool as well, so all three points are aligned. This property of the scaling vacuum allows for the copied items to be selected with cursor movements that are identical in direction, but smaller in magnitude, to the movement required for selecting the original item. Unlike drag-and-pop/pick, the relative position of the proxies to the user is identical to that of the original objects and if the user re-invokes the vacuum from the same position proxies will retain their position even though the user's direction might change. These properties are consistent with our *predictable and consistent* design principle, and alleviate the *"layout of proxies"* and *"number and size of proxies"* problems identified in drag-and-pop/pick. The disadvantage is that the proxies are significantly smaller than the original objects, and thus harder to perceive and interact with in detail without first selecting them.

Dismissal and Multiple Operations

In its simplest form, the vacuum is active only during a single pen-down, drag, pen-up gesture, with the tool being seamlessly dismissed at the pen-up event. However, as with drag-and-pop/pick, this precludes operations that require multiple pen up/down events such as interactive manipulation of a slider widget, or multiple selections from a tool palette without re-invoking the vacuum multiple times. An easy way to make the vacuum persistent would be to make it a modal tool with an explicit "dismiss" operator, which would detract from our *transient* design principle and reduce the vacuum's extremely fluid operation in the simple single operation situation.

Our solution was to leverage the "hover" state [3, 13] that is detectable in most pen-based systems. When the pen is lifted off the sensing surface, a hover zone (1-10 cm depending on the hardware) continues to track the position of the pen. As usual, a pen-down event occurs when the pen touches the surface, and a pen-up event when it leaves the surface, allowing pen-down, drag, pen-up operations to occur as expected. However, a pen-out-of-range event occurs when the pen leaves the hover zone, which we use to dismiss the vacuum. When the pen is off the surface but in the hover zone, the vacuum remains visible and active, but none of its parameters are adjustable. However, the user can now perform multiple operations with the proxies by multiple pen up/down and drag events within the persistent vacuum. For example, multiple proxies can be selected by simply tapping on them, or a complex interface widget can be operated by sequential tapping or pen-down and drag actions. The user can also adjust the vacuum's arc angle or direction by simple pen-down and drag actions (Figure 3).

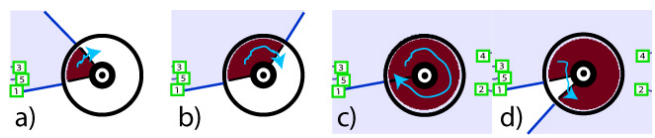

Figure 3. Arc adjustment. (a, b) Moving cursor beyond one edge moves it, expanding the arc. (c) When both edges meet, the arc is fully expanded. (d) Reversing cursor direction reopens the arc.

To prevent abrupt or accidental dismissal of the vacuum, we visually render the vacuum with increasing transparency as the pen's distance off the surface increases, as feedback to the user to indicate that the hover threshold is about to be reached. When the pen reaches the hover threshold, a simple "popping" animation indicates dismissal of the vacuum (Figure 4). This implicit method of dismissing the vacuum allows for multiple operations to be fluidly supported while simultaneously retaining the extremely valuable transient nature of the drag-and pop/pick technique. This conforms to our *transient* and *flexible* design principles, and also solves the "*multiple operations*" problem identified in drag-and-pop/pick. We note that other techniques recently reported in the literature have also used hover in interesting ways (e.g., tracking menus [4]), which is an indication that hover might eventually become a standard part of our interaction vocabulary.

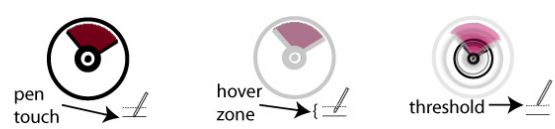

Figure 4. Hover and dismissal. (left) Vacuum is active and adjustable when pen is on surface. (middle) When pen is in hover zone, vacuum is active but not adjustable and fades out as pen reaches the edge of the hover zone. (right) When pen exceeds hover zone's threshold, vacuum is dismissed with a "popping" animation.

EXPERIMENT 1

Goals

This experiment compares the performance of the vacuum to drag-and-pick and direct (unaided) picking for a single remote selection on a large-scale display. We expect the dismissal cost of the vacuum to affect its performance. The effect of distance from the target and density of distracter objects is also investigated.

Apparatus

We used a 16' wide, 6' high, back projected large display, with imagery generated by 18 projectors (1024x768 resolution each) in a 6x3 tiling for an effective resolution of 6144x2304 pixels. The projectors are driven by a cluster of 18 workstations. Software was written in C++ with Chromium (chromium.sourceforge.net) providing graphics rendering over the cluster. A camera-based Vicon motion tracking system (www.vicon.com) tracked a pen's movement over the screen. Although the system could track the pen in 3D space, we used only x-y screen movements, a 5 inch hover zone, and a single button.

Participants

Two female and 4 male volunteers participated in the experiment. All but one participant was right handed. All participants had experience using a mouse, but none had previous experience with large display interaction.

Task

The task was discrete target selection in the presence of distracter targets, where users had to first click on a start target and then proceed to select the goal target. Participants had to successfully select the goal target before the trial would end, even if it required multiple clicks. This removed the possibility of participants trying to "race through the experiment by clicking anywhere". Targets were rendered as squares, with the start target in red, and the goal and distracter targets as numbered outlined squares. The goal target was numbered "1", and the distracters were assigned other random numbers. In pilot studies we used colors to identify the goal target but found that users could effectively ignore the distracters based on color cues only. We felt that this was not a realistic simulation of real interface use, and decided on the numbering mechanism which ensured that no target (apart from the start) stood out in a prominent way. All targets were visible at the start of the trial, so users could plan their actions before beginning, thus simulating the situation where users are familiar with the general interface layout.

Design

A repeated measures within-participant full factorial design was used. The independent variables were selection technique (*Technique*: direct, vacuum, drag-and-pick), distance between start and goal targets (*Distance*: Mid, Far), and density of distracter targets along the path from start to goal targets (*PathDistracters*: None, Few, Many).

In pilot studies, we found that varying the target size affected all three techniques to a similar extent in terms of selection time and errors. As such, in this formal study we used only one target size of 6 inches, which we feel is representative of icons on a large display of this size. Given that open windows and other applications will likely be larger, this size can be viewed as a lower-bound.

We used two distances between the start and goal targets: *Mid* = 69", roughly the display height, and *Far* = 103.5", roughly 3/2 the display height. For each of the distances, we positioned start and goal targets in 6 relative compass directions (E, NE, NW, W, SW, SE) about the centre of the screen, to counterbalance for any possible differences in movement direction. We did not use the N and S directions because they would have involved starting uncomfortably from the bottom or top of the screen and moving to the other edge for the *Mid* distance, and would have been impossible for the *Far* distance.

In real interfaces, the path towards a goal is rarely void of any interfering objects. As such, for any distance reaching technique to be successful, it must work well in the presence of intervening objects that can distract the user and the technique from the act of selecting the goal target. In pilot studies we explored the effect of two types of distracters: *area distracters* that were placed in the immediate vicinity of the goal target, and *path distracters* that were placed along the path from start to goal targets. We found that manipulating the density of area distracters had no significant effect on the three techniques, and we thus used a fixed density of area distracters in this formal experiment. For path distracters, we investigate three densities: *None*: no distracters, *Few*: 40% of a rectangular area between the start and goal targets is filled with distracters, and *Many*: 80% of this area is covered.

Participants were randomly assigned to 3 groups of 2. In each group participants used all 3 techniques, in an ordering balanced using a Latin square. For each technique participants completed 3 blocks of 36 trials. Each block consisted of trials for all combinations of *Distance* and *PathDistracters*. In summary the experiment consisted of:

6 participants x
3 techniques x
3 repetition blocks per technique x
2 distances x
6 directions x
3 path distracter densities
= 1944 trials

Prior to each technique participants were given a short warm-up session (15 trials) to familiarize themselves with the technique. Participants were allowed to take breaks between trials and blocks and were required to take breaks between techniques. The experiment lasted on average one hour for each participant. Participants filled out a brief questionnaire after the experiment to elicit their opinions.

Performance Measures

To evaluate the three techniques we examined time and error measures. *Total Time* is the time to fully conclude a trial, including error correction, and in the case of vacuum the cost of dismissing the vacuum by moving the pen beyond the hover threshold. *Selection Time* is the time to successfully select the target, including error correction, but not including the time to dismiss the vacuum. For the direct and drag-and-pick techniques *Selection Time* is equivalent to *Total Time*. *First Selection Time* is the time to the first selection, even if it was an error, and approximates the error free interaction of expert users. *Error Rate* is the percentage of trials where the target was not selected on the first click.

Results and Discussion

Trials were marked as outliers when *Total Time* was beyond 3 standard deviations from the mean for a given condition. Data from a total of 57 trials (3%) were identified as outliers and removed from further analysis.

Time Analysis (Figure 5)

Analysis of variance with post-hoc pairwise means comparisons showed no significant differences between the three techniques as measured by *Total Time*, except in the *Far* distance condition with *Few* or *Many* path distracters where the vacuum was significantly slower than drag-and-pick ($p < .01$). Overall mean times for drag-and-pick were fastest (2.49sec), followed by direct picking (2.586sec) and vacuum (3.037sec).

Interestingly, there was no significant difference between the techniques, even in the *Far* distance condition, when *Selection Time* was used as the dependent variable ($F_{2,10} = .155$, $p = .858$). In this case, drag-and-pick was slightly faster (2.49sec), followed by vacuum (2.532sec) and direct picking (2.586sec). Similarly, no significant effect for *Technique* on *First Selection Time* was present.

Comparing the overall means for *Total Time* with *Selection Time*, it is apparent that the vacuum has slight overhead (~0.5 sec) that can be attributed to the time required to dismiss the widget. Although the use of hover as an implicit dismissal mechanism was a design choice aimed at reducing this cost, it is clear that the cost is not completely eliminated. Interestingly users did not perceive this cost as a delay and some of them commented they found the action of lifting the pen after completing the task very natural. Moreover, we believe that this dismissal cost can be further reduced for experienced users by reducing the hover zone.

As expected from Fitts' law, *Distance* significantly affected *Total Time* ($F_{1,5} = 343.4$, $p < .0001$), but there was no significant *Technique* x *Distance* interaction ($p = .093$) indicating that all techniques were similarly affected by changes in distance. This result can be explained by the following analysis of the techniques:

The scaling property of the vacuum results in the size of its proxy targets being significantly smaller than the proxies in drag-and-pick, or the original targets in direct picking. The distance between the start position and the proxy target also shrinks, and since the scaling is uniform, the resulting Fitts' law index of difficulty (ID) of the proxies in the vacuum are similar to that of the targets in direct picking. In contrast, the proxies in drag-and-pick are unchanged in size but are moved closer to the start position, resulting in a smaller Fitts' ID compared to the vacuum or direct picking. Thus one might expect vacuum and direct picking to be similarly affected by changes in distance, whereas drag-and-pick should not be affected. The fact that drag-and-pick is affected by distance in our experiment is due to the presence of *PathDistracters.* Since *PathDistracters* were controlled based on density (40% or 80% of a rectangular area between start and target), and the area occupied by the distracters themselves could not be compressed by the drag-and-pick technique, the resulting area separating the target from the start position in the *Mid* distance is smaller than that in the *Far* distance. Thus, changes in distance affect drag-and-pick, but only when distracters are present.

PathDistracters had a significant effect ($F_{2,10} = 12.7$, $p < .01$) on *Total Time* with *Many PathDistracters* resulting in the slowest times followed by *Few* and *None*. A significant *Technique* x *PathDistracters* interaction ($F_{4,20} = 4.7$, $p < .01$) was present. Pairwise means comparison showed drag-and-pick significantly faster than vacuum for the *Far* distance with *Few* or *Many* distracters (all $p < .01$), with no other pairs being significantly different ($p > .05$).

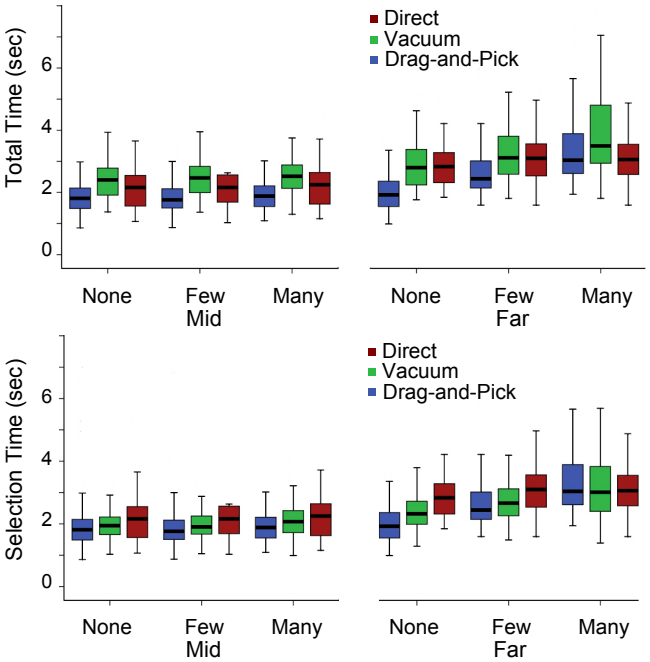

Figure 5. Boxplots (median,range) of *Total Time* (top) and *Selection Time* (bottom) broken down by *Technique*, *Distance* (Mid, Far), and *PathDistracters* (None, Few, Many).

As seen in Figure 5 direct picking is not affected by distracters since users know the location of the goal target and acquisition time is only affected by distance. In contrast, drag-and-pick and vacuum incur an extra cost of

identifying the proxy after a set of targets are selected, and performance degrades as the density of distracters increases. In the absence of distracters, however, drag-and-pick and vacuum are either equivalent or marginally faster than direct picking for *Far* distances.

It is interesting that the significant advantages of drag-and-*pop* over direct *dragging* observed by Baudisch et al [2] is not seen in our study comparing drag-and-*pick* to direct *picking.* We believe this is because direct dragging requires the user to maintain contact with the screen surface, which is difficult to do over long distances. In contrast, direct picking can be accomplished by tapping on the start target, lifting the pen, walking quickly over to and tapping the goal target. The pen does not have to maintain contact with the screen and so it is not all that much worse than using drag-and-pick or the vacuum. The user simply trades off walking for the complexity of the drag-and-pick or vacuum.

Error Analysis

Analysis of variance showed no significant effect for *Technique, Distance,* or *PathDistracters* on *Error Rate*. Overall, 2% of trials in direct picking were errors, 3% in the vacuum, and 5% for drag-and-pick. Interestingly, 67% of all errors in drag-and-pick were made by users invoking the technique with an angle that missed the target. Errors of this nature only accounted for 5% of all errors for the vacuum, whereas 52% of errors were due to accidental dismissal of the widget and 43% due to simply missing the target.

Subjective Comments from Participants

After the experiment users were asked to rank the techniques based on speed, accuracy, and ease of understanding. They were also asked to state if they had a preference for any of the techniques.

3 participants ranked vacuum and drag-and-pick as equally fast, 2 ranked the vacuum as fastest and 1 ranked drag-and-pick as fastest. The two who ranked vacuum higher mentioned that they could identify the target faster than with drag-and-pick: they would start moving towards the rough position of the target and quickly re-identify it amongst the proxies. On the contrary, they felt they spent more time deciding on the angle of approach when using drag-and-pick, time not captured for this task.

4 participants ranked direct picking as the least error prone, while the other 2 participants each ranked vacuum and drag-and-pick as the least error prone respectively.

All participants ranked direct picking as the easiest to understand. Interestingly, for the task at hand, 4 participants preferred the vacuum and 2 preferred drag-and-pick, even though vacuum was slower. They all preferred using a mediator instead of walking to the targets. This indicates that even though direct picking was not significantly different, users will use a remote reaching technique to avoid walking across a display of this size.

EXPERIMENT 2

Goals

The goals of this experiment were similar to experiment 1, except that we now use the three techniques in the context of selecting multiple targets in sequence. We examine how between target distance affects the different techniques, as well as the presence of distracter targets on the screen. Since the vacuum is designed for such a usage scenario we expect it to outperform the other two techniques.

Participants

3 female and 6 male volunteers participated in the experiment. All had previous experience using a mouse but none had previous experience with large display interaction.

Task

The task was consecutive selection of five targets, in the presence of distracters. Participants started a trial by first selecting a red start target, and proceeded to select each of the five goal targets in sequence. If the vacuum was accidentally dismissed before all targets were selected, the user had to re-invoke it from the starting position. For drag-and-pick, each invocation started from the red start target, in order to prevent users from simply reverting to direct picking once they reached a cluster of targets. We adjust for this additional overhead in our movement time analysis.

Targets were rendered in a similar manner to experiment 1. The five goal targets were numbered 1 to 5 and distracters were assigned other numbers. To prevent participants from "racing through the experiment by clicking anywhere", they had to successfully select each target in the correct sequence. All targets were visible at the start of the trial, so users could plan their actions before beginning, simulating situations where users are familiar with the interface layout.

As each target of the sequence was selected it changed color from white to green to help the participant keep track of where they were in the sequence. In direct pointing and drag-and-pick, the trial ended when the fifth target was successfully selected. For the vacuum, when the fifth target was selected, all targets turned red indicating it was time to dismiss the vacuum by pulling away from the hover zone.

Design

A repeated measures within-participant full factorial design was used. The independent variables were selection technique (*Technique*: direct, vacuum, drag-and-pick), between-targets distance (*Distance*: Close, Mid, Far), and number of distracter targets that were randomly scattered about the display (*Distracters*: None, Few, Many).

The three between-target distances used were: *Close* = 17", *Mid* = 34", and *Far* = 51". These represent 1/4, 1/2, and 2/3 of the height of our display respectively. In the *Close* and *Mid* distances we cluster the targets as close as possible (rather than stringing them along in a continuous sequence, for example) to simulate real situations where users group windows or icons in clusters. The result was that in the

Close distance, the targets were approximately within a quadrant of the display, in the *Mid* distance targets were in half of the display, and in the *Far* distance targets spanned the entire display. For each of the distances we used two orientations, one in which the first target was on the left and another on the right of the starting position.

Since our targets were located on different areas of the screen we decided to place the distracter targets randomly, rather than the more specific placements of area and path distracters in experiment 1. We used three densities: *None*, *Few*: 15% screen coverage, and *Many*: 20% coverage.

Participants were randomly assigned to 3 groups of 3. In each group participants used all 3 techniques, in an ordering balanced using a Latin square. For each technique participants completed 3 blocks of 18 trials. Each block consisted of trials for all combinations of *Distance* and *Distracters*. In summary the experiment consisted of:

9 participants **x**
3 techniques **x**
3 blocks per technique **x**
3 distances between targets **x**
3 distracter layout densities **x**
2 orientations
= 1458 trials

Participants performed 10 warm-up trials per technique and were asked to complete the trials as fast and accurately as possible, taking breaks between trials and blocks.

Performance Measures

To evaluate the three techniques we examined time and error measures. The first measure is the *Total Time* it took users to complete a task. This includes the vacuum dismissal cost. As discussed earlier, drag-and-pick is a single target selection technique. In order to simulate realistic use of drag-and-pick for multiple target selection, our experiment design required users to return to the red starting target between each selection of the five targets. In contrast, the vacuum and direct picking easily handled multiple target selection in sequence and there was no need to return to the start position each time (except when the vacuum was accidentally dismissed).

To avoid biasing against drag-and-pick, for all drag-and-pick data we computed an additional measure called *Adapted Total Time* where we removed the time between selecting a target and returning to the start position to start the process of selecting the next target. In a sense, this measures a best case scenario for drag-and-pick. Similarly, we also removed the time to return to the start position in the few cases of accidental dismissal of the vacuum. Since the final dismissal of the vacuum is crucial for this type of task, we take it into account everywhere in our analysis.

Finally, we examined the *Error Rate*, computed as the percentage of trials where targets were not selected correctly on the first attempt**.**

Results and Discussion

Trials were marked as outliers when *Total Time* was beyond 3 standard deviations from the mean for a given condition. Data from a total of 22 trials (1%) were identified as outliers and removed from further analysis.

Time Analysis (Figure 6)

Analysis of variance showed a significant main effect for technique on *Total Time* ($F_{2,16} = 100.7$, $p < .0001$). Vacuum was the fastest (8.117sec), followed by direct picking (8.525sec) and drag-and-pick (16.359sec). Pairwise means comparisons showed that drag-and-pick was significantly slower than vacuum and direct picking ($p < .001$).

Similar results were found for *Adapted Total Time,* with a significant main effect for technique ($F_{2,16} = 20.2$, $p < .0001$). Overall mean times were fastest for vacuum (7.645sec), followed by direct picking (8.525sec) and drag-and-pick (10.679sec). Despite the large reduction in time for drag-and-pick compared to the *Total Time* measure, drag-and-pick was still significantly slower than both vacuum and direct picking ($p < .01$).

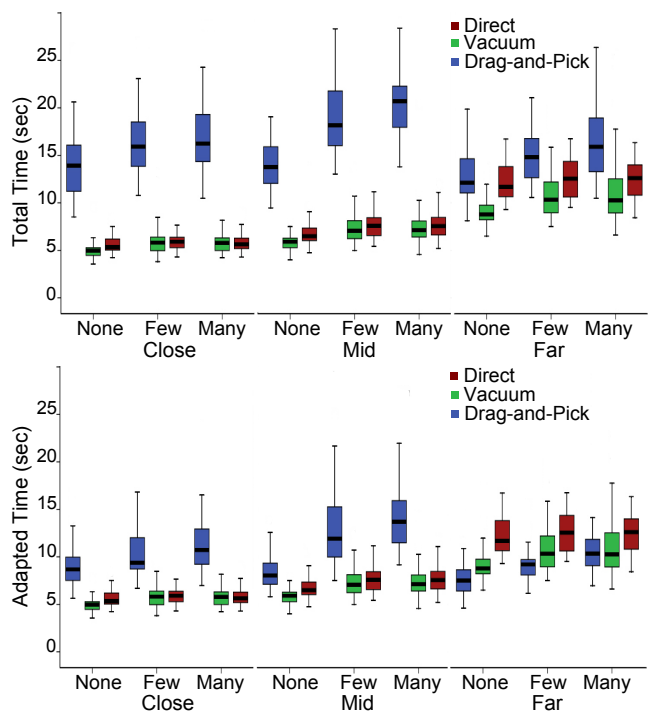

Figure 6. *Total Time* (top) and *Adapted Total Time* (bottom) broken down by *Technique, Distance* (Close, Mid, Far), and *Distracters* (None, Few, Many).

As expected, between target *Distance* had a significant effect on *Total Time* ($F_{2,16} = 174.7$, $p < .0001$). A significant *Technique* x *Distance* interaction ($F_{4,32} = 77.9$, $p < .0001$) was also present. Post-hoc pairwise means comparisons showed that the vacuum was significantly faster than drag-and-pick ($p < .01$) for all distances. Drag-and-pick was significantly slower than direct pointing for *Close* and *Mid*

distances ($p < .0001$) and not significantly different from direct pointing for the *Far* distance ($p = 0.065$).

Between target *Distance* also had a significant effect on *Adapted Total Time* ($F_{2,16} = 177.4$, $p < .0001$). There was a significant *Technique* x *Distance* interaction ($F_{4,32} = 78.5$, $p < .0001$), with vacuum and drag-and-pick significantly faster than direct picking in *Far* distances (all $p < .022$), but drag-and-pick significantly slower than both vacuum and direct picking in *Close* and *Mid* distances (all $p < .0001$).

Interestingly, although we expected the performance of drag-and-pick to be similar across distances, it performs better in the *Far* distance than the *Mid* distance. This may be a result of using proximal targets as guides to plan the next movement. In the *Far* case, although two consecutive targets are far away, there are two very tight clusters of targets at each side of the user. For instance, by selecting target 2 the user sees that target 4 is also going to end up very close to that position. In the *Mid* distance as the user selects a target, the remaining desired target group might or might not end up in the resulting set of proxies, making it harder to plan ahead.

The *Distracter* density had a significant main effect on *Total Time* ($F_{2,16} = 164.2$, $p < .0001$). Also, a significant *Technique* x *Distracter* interaction was present ($F_{4,32} = 78.7$, $p < .0001$) with drag-and-pick significantly slower than vacuum and direct picking in all distracter cases (all $p < .0001$). This indicates that in the presence of distracters, users exploit the persistent layout of vacuum and direct picking to identifying multiple targets.

The *Distracter* density also had a significant effect on *Adapted Total Time* ($F_{2,16} = 155.63$, $p < .0001$). A significant *Technique* x *Distracter* interaction ($F_{4,32} = 40.9$, $p < .0001$) was also present, with vacuum being significantly faster than drag-and-pick in all distracter cases (all $p < .005$) and drag-and-pick significantly slower than direct picking for *Few* ($p < .05$) and *Many* ($p < .01$), due to the persistent layout benefits of direct picking being more prominent in the *Few* and *Many* cases.

As discussed earlier, drag-and-pick is a single target selection technique and our experiment design required users to return to the starting target between each selection of the five targets. To not bias against drag-and-pick we used the *Adapted Total Time* measure. This metric is favorable for drag-and-pick since by subtracting the return to start time, we also remove the time the users needed to plan their movement in order to include all targets in the arc of influence of drag-and-pick. Nevertheless, the only major difference occurred in the *Far* between targets distance.

Error Analysis

Analysis of variance showed a significant main effect for technique on *Error Rate* ($F_{2,16} = 36.5$, $p < .0001$). Mean error rates were 6.6% for direct picking, 16% for the vacuum and 39% for drag-and-pick. As in Experiment 1, the majority of drag-and-pick errors occurred when users missed the target in their initial choice of movement angle (32% of all errors) and when they selected the wrong target in the group of proxy targets (29% of all errors) which indicates that users found it hard to distinguish the proxy from other targets. In the vacuum, the majority of errors (87%) were those where targets were missed due to their small sizes, and 13% were accidental dismissal errors.

Subjective Comments from Participants

After the experiment users were asked to rank the techniques based on speed, accuracy, and ease of understanding. They were also asked if they had a preference for any of the techniques.

All 9 participants ranked the vacuum as the fastest technique. All ranked drag-and-pick as the most error prone. 5 participants ranked vacuum as the least error prone, 3 chose direct picking, and 1 ranked both vacuum and direct picking as equivalent in terms of accuracy.

All 9 participants ranked direct picking as the easiest to understand, with vacuum generally ranking second over drag-and-pick. 4 participants mentioned that although the idea behind drag-and-pick was easy to understand, predicting the final position of the proxies was not. They added that they had to spend a lot of time to re-identify the appropriate proxy and that sometimes they missed the target completely with the selection arc, or failed to identify it in the group of proxies. One participant made it more concrete by stating it was hard to judge due to distance where the arc was in effect. For the vacuum, they all mentioned that when targets were at a single direction the technique was extremely easy to use, but that they had some problems getting used to the angle adjustment for items scattered in opposite directions.

In terms of overall preference for this particular task, all participants ranked the vacuum highest, 5 ranked drag-and-pick second, 2 ranked direct picking second, and 2 ranked drag-and-pick and direct pointing equivalent.

CONCLUSIONS

We have presented the design and evaluation of the vacuum, a new remote reaching interaction technique. The vacuum offers several advantages over drag-and-pop/pick. Its visible influence arc clarifies the effects of the vacuum for users. Users can adjust the direction and angle of the influence arc dynamically during an operation, and the addition of a hover region above the surface allows multiple operations to be combined in a single invocation of the tool. Shrunken proxies allow more proxies to be represented and permit the proxies to maintain the relative spatial arrangement of the original items.

Our experimental results indicate that the vacuum performs similarly to direct picking and drag-and-pick in single target selection tasks, except when the targets are located very far away with distracters along the path. However, the vacuum scales nicely to multiple target selection, and performs significantly better in this scenario than existing techniques.

The main drawback of the vacuum relative to drag-and-pick as a visualization technique is the scale of the proxies. This will be especially prominent if the need to differentiate between similar icon sized targets arises. We believe the high resolution of our display alleviates some of the scale issues since proxies are relatively decipherable. However, lower resolution displays may make the proxies unreadable.

On the other hand the main drawback of drag-and-pick is its unpredictability. This can be resolved in two ways: by providing a better visualization of the sector of influence, and by positioning the items based more on the starting position of the user and less on the direction of movement.

It is important to note that while direct picking performed reasonably well in our experiments, in real use it may actually not be possible to directly point to all parts of a large display. Moreover, in extended use, user comments indicated that they prefer techniques that minimize physical travel, like the vacuum, even if it incurs a slight overhead.

Our work has also indicated how small design variations can impact an interaction technique. For example, the use of hover enabled us to design the vacuum to work well with multiple targets. Further design alternatives for both the vacuum and drag-and-pick could also be explored. For example, a reviewer of this paper pointed out that an interesting variation to the vacuum would be to use a hard boundary right before dismissal to clearly warn the user of what is about to happen, rather than our current design of smoothly fading the widget as the pen moves away.

More generally, our work demonstrates that tasks requiring infrequent remote access typically benefit from low dismissal cost techniques like drag-and-pick or a vacuum with a small hover range. Tasks based on sequential or frequent remote content access are better accommodated by more flexible techniques, like the vacuum presented here. Finally, while our work focused on distance reaching techniques for large displays, the vacuum could also be used on smaller devices like TabletPCs, where users may prefer working in a comfortable zone and not reaching to the display extremities.

ACKOWLEDGEMENTS

We thank Fanis Tsandilas, Dan Vogel, John Hancock, Gonzalo Ramos, our experiment participants, the anonymous CHI reviewers, and DGP lab members.

REFERENCES

1. Baar, J., Willwacher, T., Rao, S., and Raskar, R. (2003). Seamless multi-projector display on curved screens. *Eurographics Workshop on Virtual Environments (EGVE)*. p. 281-286.
2. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B., and Zierlinger, A. (2003). Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch- and pen-operated systems. *Interact Conference*. p. 57-64.
3. Buxton, W. (1990). Three-state model of graphical input. *Interact Conference*. p. 449-456.
4. Fitzmaurice, G., Khan, A., Pieke, R., Buxton, B., and Kurtenbach, G. (2003). Tracking menus. *ACM UIST Symposium*. p. 71-79.
5. Funkhouser, T. and Li, K. (2000). Onto the Wall: Large Displays. *Special issue of IEEE Computer Graphics and Applications.*, 20(4).
6. Geibler, J. (1998). Shuffle, throw or take it! working efficiently with an interactive wall. *Extended Abstracts of the ACM CHI Conference*. p. 265-266.
7. Guimbretière, F., Stone, M., and Winograd, T. (2001). Fluid interaction with high-resolution wall-size displays. *ACM UIST Symposium*. p. 21-30.
8. Hascoët, M. (2003). Throwing models for large displays. *British HCI Conference*. p. 73-77.
9. Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., and Smith, M. (2004). Stitching: Pen gestures that span multiple displays. *ACM AVI Conference on Advanced Visual Interfaces*. p. 23-31.
10. Khan, A., Fitzmaurice, G., Almeida, D., Burtnyk, N., and Kurtenbach, G. (2004). A remote control interface for large displays. *ACM UIST Symposium.* p. 127-136.
11. Landay, J. and Pier, K. (1992). Issues for location-independent interfaces, Technical Report ISTL92-4, Xerox PARC.
12. Mynatt, E., Igarashi, T., Edwards, W., and LaMarca, A. (1999). Flatland: New dimensions in office whiteboards. *ACM CHI Conference*. p. 346-353.
13. Newman, W. (1968). A system for interactive graphical programming. *AFIPS Spring Joint Computer Conference*. p. 47-54.
14. Pederson, E., McCall, K., Moran, T., and Halasz, F. (1993). Tivoli: An electronic whiteboard for informal workgroup meetings. *ACM CHI Conference*. p. 391-398.
15. Raskar, R., Baar, J., and Chai, J. (2002). A low-cost projector mosaic with fast registration. *Asian Conference on Computer Vision (ACCV)*.
16. Rekimoto, J. (1997). Pick and drop: A direct manipulation technique for multiple computer environments. *ACM UIST Symposium*. p. 31-39.
17. Streitz, N., Geibler, J., Holmer, T., Konomi, S., Müller-Tomfelde, C., Reischl, W., Rexroth, P., Seitz, P., and Steinmetz, R. (1999). i-LAND: an interactive landscape for creativity and innovation. *ACM CHI Conference*. p. 120-127.
18. Swaminathan, K. and Sato, S. (1997). Interaction design for large displays. *Interactions*, 4(1). p. 15-24.
19. Tan, D., Meyers, B., and Czerwinski, M. (2004). Wincuts: Manipulating arbitrary window regions for more effective use of screen space. *Extended Abstracts of the ACM CHI Conference*. p. 1525-1528.
20. Wu, M. and Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *ACM UIST Symposium*. p. 193-202.