Improving window manipulation and content interaction on high resolution, wall-sized displays

Chris Rooney
Middlesex University
London
NW4 3NH
England
+44 0208 411 4284
c.rooney@mdx.ac.uk

Roy Ruddle
University of Leeds
Leeds
LS2 9JT
England
+44 0113 343 1711
royr@comp.leeds.ac.uk

October 25, 2011

Abstract

Interaction with high resolution wall-sized (Powerwall) displays can be a tedious and difficult task due to large display areas and small target sizes. To overcome this, we developed techniques that reduce the precision required to manipulate windows and select data. The manipulation layer speeds up the common tasks of moving and resizing application windows by overlaying them with large, transparent target areas. The Power-Lens magnifies target sizes by automatically appearing once the cursor reaches the region of

Rooney, C. & Ruddle, R. (2011) Improving window manipulation and content interaction on high resolution, wall-sized displays. International Journal of Human-Computer Interaction, iFirst

interest. Two experiments evaluated these techniques against conventional desktop-style interfaces. Experiment 1 showed the window manipulation layer to speed up the tasks of moving and resizing a window by 24% and 27% respectively. Experiment 2 showed the Power-Lens to speed up the selection of 5x5 pixel targets by 18%. Together, our new techniques help to make interaction more fluid on Powerwall displays.

Keywords: Powerwall, interaction, precision.

1 Introduction

High resolution, wall-sized displays (or Powerwalls) are becoming increasingly popular due to the vast amount of information they can display for visualization applications (Jedrysik, Moore, Stedman, & Sweed, 2000; Buxton, Fitzmaurice, Balakrishnan, & Kurtenbach, 2000; Goodyer, Hodrien, Wood, Kohl, & Brodlie, 2009; Treanor, Owers, Hodrien, Quirke, & Ruddle, 2009). Powerwalls are generally constructed using arrays of TFT monitors and are designed to be viewed at arm's length. Examples include the 54 million pixel Powerwall in our laboratory (Figure 1), which was constructed using commodity hardware, and the 245 million pixel Hyperwall-2 at NASA Ames Research Center (NASA, 2008).

Most Powerwall applications are designed so that a given item of data occupies the same number of pixels on the wall as on a standard desktop display, so the quantity of information that is visible scales with the number of pixels on the wall. This additional real estate increases both the number of application windows that can be viewed at anyone time, and the physical distance between them. If the interface remains unchanged then, as Fitts' law (Fitts, 1954) predicts, the increased display area substantially slows down interaction.

This paper describes two techniques that address this problem of interaction speed by reducing the level of precision required for users to interact. The precision for window manipulation (moving and resizing windows) was lowered by overlaying windows with a novel *manipulation layer* that offers large, transparent target areas. Interaction with window content was improved by the introduction of the *Power-Lens*, which is unique in its ability to automatically appear, and remain stationary, when the user wishes to select a target, thus increasing the target size without increasing the sensitivity of the cursor.

Both solutions were evaluated against a conventional interface using Fitts' law-type tasks (Soukoreff & MacKenzie, 2004; Murata, 1999) to check whether the benefits that our new interaction techniques theoretically provide also occur in practice. This is important because interaction with high resolution, wall-size (Powerwall) displays is still an emerging field and: (a) the time required for interaction depends on both Fitts law factors (distance traveled & target size) and additional factors (e.g., interaction with transparent targets & the appearance of the Power-Lens mid-travel), (b) we wish to quantify the magnitude of the benefit provided by our new techniques, and (c) the predicted benefits of new interaction techniques do not always occur (Gutwin, 2002; Bederson, 2000). The following sections describe the background to, and implementation of, the manipulation layer and Power-Lens. Then experiments that evaluated the techniques are reported.

2 A Low Precision Approach for Window Manipulation

The increased display area means that users tend to have more windows open simultaneously on a Powerwall than a desktop system (Hutchings, Smith, Meyers, Czerwinski, & Robertson, 2004; Bi & Balakrishnan, 2009) but, with conventional interfaces on a Powerwall, window management tasks such as moving and resizing are slow and cumbersome (Robertson et al., 2005). The root cause is that the move and resize widgets are small in a standard system (see Figure 2(a)). Also, keyboard-driven alternatives (e.g., holding down the ALT key while pressing the left mouse button on Linux) are cumbersome and, therefore, rarely used.

Our new manipulation layer speeds up Powerwall interaction by overlaying a transparent layer on top of the window, to separate content interaction from window manipulation. When the cursor is inside the window, users can choose to interact with either the window content or the manipulation layer. A wide variety of interface devices may be used but, contrary to popular belief, the mouse is very effective with Powerwalls (Ball, North, & Bowman, 2007).

The manipulation layer is divided into nine regions (see Figure 2(b)). Holding a button while the cursor is in the center region allows the window to be moved, and when the cursor is in an outer region, the window can be resized. Since the manipulation layer is completely transparent and does not alter the appearance of the window, visual feedback is required to indicate which region the cursor resides in. Rather than modify the shape of the cursor, which is known to degrade performance (Phillips, Meehan, & Triggs, 2003), a small arrow appears inside the cursor to indicate which region the cursor resides in. The direction of the arrow lets the user know in which direction they can move the window. No arrow is present when the cursor is inside the central move area.

The button that is used with the manipulation layer depends on which interface device is adopted. Some devices have a multitude of buttons, whereas devices such as mice are limited. We used the middle button of a three button mouse, because that is the button that traditional interfaces use least. However, in a fully-functional system, this would require operations such as paste to be performed in another way (e.g., the keyboard shortcut <ctrl>v).

3 Interacting with Data

Techniques for aiding target acquisition can be generalized into two groups, those that are target-aware and those that target-independent. Target-aware techniques use the location of targets to help the user select them. While they have been known to improve performance (Murata, 1998; Lane, Peres, Sndor, & Napier, 2005), existing applications must be modified to accommodate them. Target-independent techniques either work at the device level (e.g., altering the mouse-cursor gain) or at the window-manager level.

At the window-management level, the adoption of a lens is a popular solution for helping users interact with fine data in applications. While a number of designs have been proposed, the basic principal is to show a magnified view of a small amount of data within the context of the main graphical window, which shows the data as a whole. Lens designs differ according to parameters such as the projection, position, when it appears, and transparency (see Table 1). The effect of these parameters on a lens' usability is reviewed below, based on reports of previous research and prototypes we implemented. This is followed by a description of a new design of lens (the Power-Lens), which was evaluated in Experiment 2 of this paper.

3.1 Parameters of Lens Design

3.1.1 Projection

The projection can be either linear or fisheye (also termed hyperbolic). Fisheye views (Furnas, 1986) magnify a region of interest by compressing the surrounding context. This allows the point of focus to be magnified without obstructing any of the data, but causes the data as a whole to become distorted. The distortion changes the rate at which data approaches the cursor, making selection of a given item of data more difficult. Speed-coupled flattening, a technique that reduced the level of distortion when the cursor traveled at high speed, improved target selection through the lens (Gutwin, 2002). Results, however, also showed that participants performed target selection quicker without any lens, indicating that Fisheye lenses actually hinder interaction. A similar problem was found with the Fisheye Menu (Bederson, 2000), which was slower than a hierarchical approach for selecting particular items from linear menus.

The negative aspects of fisheye lenses were confirmed by a prototype we implemented and tested with 5 x 5 pixel data items. Once an item of data was within the lens, the non-linear projection made it difficult for users to predict how far they needed to move the interface device to select the item, so interaction was slower than without a lens.

3.1.2 Position

Linear (i.e., non-distorted) lenses have the advantage of magnifying the region of interest without distorting any of the data. As a consequence, some of the underlying data is obscured, but which data that is depends on where the lens is positioned.

Implementations such as the Magic Lens (Bier, Stone, Pier, Buxton, & DeRose, 1993), DragMag (Ware & Lewis, 1995) and Idelix lens (Carpendale, Ligh, & Pattison, 2004), permanently display a *stationary* lens that the user can manually reposition. These lenses present a problem for precise drag and drop tasks that occur over large distances, because the lens needs to be manually positioned at both the pick and release points of the task. By contrast, the Pointing Lens (Ramos, Cockburn, Balakrishnan, & Beaudouin-Lafon, 2007) could be (de)activated by the user and displayed at the last cursor position, which is more appropriate for the large distances that are involved in Powerwall interaction. However, our prototyping showed that the actions needed to manually (de)activate a lens made target selection slower than selection without a lens.

Cursor-tracking lenses (Bederson, 2000; Forlines, 2005; Pietriga & Appert, 2008) are positioned relative to the cursor. These lenses magnify both the region of interest and the sensitivity of the cursor (e.g., a 10x magnification would make the cursor 10x more sensitive), so the lens makes a target easier to see but not easier to select. By contrast, the stationary lenses described above magnify the view but leave the cursor sensitivity unaltered, which reduces the precision needed to select targets.

Appert developed three types of High-Precision Lens (Appert, Chapuis, & Pietriga, 2010). The first two, *Key* and *Speed* track the cursor, while the third, *Ring*, is stationary as long as the cursor remains inside the lens. User experiments found the Ring Lens to perform the best, supporting the fact stationary lenses perform better than cursor-tracked lenses, partly because the cursor sensitivity was unaltered.

Cursor-tracking may either display a lens *on top* of the cursor, or *offset* from it. The latter simultaneously shows a magnified view of the data in the lens and a normal view of the data, but requires the user to alternate their focus between two viewpoints. Research has shown that there is no significant difference in performance between on top and offset lenses (Darling, Newbern, Kalghatgi, Burgman, & Recktenwald, 2004), and prototypes we implemented showed that both on top and offset lenses often made target selection slower than using no lens at all.

3.1.3 When a Lens Appears

Some lenses only appear when *manually* requested by the user, but most are *permanently* displayed present during interaction. The Pointing Lens (Ramos et al., 2007), a lens for touch screen interaction, could either appear when the stylus was pressed hard against the screen (pressure-activated), or could permanently trail behind the cursor. The pressure-activated lens out-performed the trailing lens for target selection. Verbal feedback also supported this, with some participants saying that the trailing lens sometimes occluded the target.

3.1.4 Transparency

Some lenses obscure any underlying data, whereas others are partially transparent. Studies have shown that 70% transparency allowed users to interact with both the lens and the underlying data simultaneously (Cox, Chugh, Gutwin, & Greenberg, 1998). However, transparency of as little as 10% can cause the user to divide their attention, adversely affecting performance (Harrison, Ishii, Vicente, & Buxton, 1995). Toolglass widgets (Bier et al., 1993) alter the way in which users interact with the underlying data, depending on the widget they use. Since the widgets are fully transparent, there is no divide in users' visual attention.

3.1.5 Magnification

Magnification is dictated by the size of the lens and size of the region of interest. There are an infinite number of combinations of size, but there appears to be no definitive answer as to which combination to use. The Pointing Lens (Ramos et al., 2007) actually lets the user alter the magnification (from 2x to 10x), but previous research showed that there was no significant difference for magnification levels of up to 4x (Gutwin & Skopik, 2003).

3.2 Power-Lens for Data Interaction

The results of previous research and our prototyping (see above) highlighted the following issues for lens design. Fisheye lenses slow down data selection because the distortion makes it difficult for users to predict the distance they need to move an interface device. Lenses need to be fixed if precision is to be reduced (cursor sensitivity is not magnified), but also near the target region on a display. Manual activation slows down interaction because of the time cost of pressing a button. Semi-transparency is beneficial because it helps to show detail of a dataset within its overall context.

Our solution, the Power-Lens is a linear, semi-transparent lens. What makes it unique is that it is neither manually activated nor permanent. Instead, the lens appears automatically as the cursor approaches a target. This novel approach is based on the observation that users move a pointer fast when traveling towards a target, then slow down to make the final selection (Worden, Walker, Bahrat, & Hudson, 1997; Appert et al., 2010; Keuning, Galen, & Houtsma, 2005). When the speed of the cursor falls below a certain threshold the Power-Lens automatically appears, centered at the cursor's threshold position and leaving the user free to interact with the content of the lens (see Figure 3). The lens deactivates automatically as soon as the cursor passes through one of the lens's edges. Each time the lens is activated it is displayed in a fixed position (the cursor's position at activation), which means that the lens combines the reduced precision and increased interaction speed of fixed position lenses with the mobility of cursor tracked lenses.

The Power-Lens is designed to be activated just before the cursor reaches the data, so that the user's final, slowing movements of the cursor mean that it comes to rest on top of the magnified data in the lens. Achieving this involved careful choice of several Power-Lens parameters, which was achieved with empirical Fitts' law (Fitts, 1954) calculations and some pilot testing. The final Power-lens design used a 1600 x 1200 window (the area of one panel on the Powerwall) and 6.7x magnification (the lower the magnification the more likely the desired data is shown within the lens, but the smaller that data is for selection). The Power-Lens was activated 400ms after the cursor speed dropped below a threshold speed of 1200 pixels/second. This combination of parameters suited most users (the cursor typically came to a stop on top of the magnified data in the lens). However, in case the lens appeared prematurely, users could click the right mouse button to toggle the lens on and off, and hold down the right mouse button to pan the position of the lens (in a fully-functional system, this functionality could be allocated to that button automatically when a lens was displayed).

4 Overview of Experiments

The following sections describe experiments that evaluated the manipulation layer (Experiment 1) and Power-Lens (Experiment 2) for Powerwall interaction, respectively. Different participants were used for each experiment.

Both the manipulation layer and Power-Lens were implemented at the window manager level and, therefore, could be used with a number of interface devices.

We have tested the techniques using pinch gloves (http://www.mechdyne.com/) with a magnetic hand tracker (http://www.ascension-tech.com/). For the experiments, the hardware device used was a standard desktop wireless mouse, since we were evaluating the software techniques, rather than non-conventional hardware devices. The mouse is one of the most precise positioning devices available, and it is known to work well with Powerwall displays (Ball et al., 2007).

5 Experiment 1: Window Manipulation

A within-participants design was used to compare the manipulation layer interface with a desktop style interface for window move and resize tasks on the Powerwall. The orders in which participants used the interfaces and performed the tasks with each interface were counterbalanced.

5.1 Method

5.1.1 Participants

The study was performed using eight participants. All were male and their age ranged from 19 to 34 years (M = 24.8), with between 8 and 25 years of computer use (M = 15.1). All participants were right handed. Each was paid £6 (roughly US$9).

5.1.2 Materials

The experiment was run on a Powerwall made up of 28 monitors, tiled 7x4, however, only 16 monitors (arranged 4x4) were used, so that the aspect ratio was the same as a standard 4:3 desktop monitor (see Figure 4). The experiment's software was custom written in C++ using the OpenGL and VRJuggler libraries.

Participants stood during the experiment, and interacted with a mouse rested on a podium (Ball et al., 2007) that was positioned in the center of the 16 active monitors (see Figure 4), and adjusted to suit the user's height.

For the desktop style interface, windows appeared the same as they would on a standard Linux desktop. To move the window the cursor had to be positioned at the title bar (20 pixels wide), and to resize the window the cursor had to be positioned at its border (5 pixels wide). For the manipulation layer, the interface was as described in the previous section. Both interfaces employed the standard cursor gain increase: 1x for fine movements, 2x for coarse movements. This allowed users to traverse the display without having to reposition (clutch) the mouse.

5.1.3 Procedure

Each participant performed one task (e.g., move) and then the other (e.g., resize) with one interface (e.g., desktop style), and then performed the tasks in the same order with the other interface (e.g., manipulation layer). For each combination of task and interface, a participant performed five blocks of 16 trials. At the start of each combination, the participant was given a short amount of time to become familiar with both the interface and task. Participants were given a 30 second rest between blocks, and a two minute rest between conditions. Overall, each participant took approximately one hour to complete the experiment.

To begin a trial for the move task, a participant started with the cursor in the center of the 16 monitor display. After a delay of two seconds, the trial started. A window (1600 x 1200 pixels in size) appeared, as well as a destination that was 2400 pixels away from the window, indicating where the window should be moved to (see Figure 4(a)). The destination had a tolerance of 120 pixels (i.e. it was 240 pixels higher and wider than the window).

At the start of each trial the center of the window was placed 1200 pixels from the center of the display. The same 16 starting positions, evenly distributed around the center, were used for every block of trials, but the order in which they were used was randomized and seeded differently for each participant.

To complete a trial, a participant had to select and then move the window until it was totally contained within the destination outline. When the task was complete, the trial ended and the window and outline disappeared, ready to start the next trial.

The resize task trials were performed in the same way as the move task, but rather than move the window, one corner was dragged to increase the window size. The destination outline was drawn as an extension of the window to show which corner needed to be extended, and by how much (see Figure 4(b)). In the resize trials, the corner of the window to be manipulated was placed 1200 pixels from the center, making the distance the same for each trial.

Each trial could be split into four stages:

1. *Find Target* - From the window appearing until the participant moved the mouse.
2. *Select Target* - From moving the mouse until the window was selected.
3. *Find Destination* - From selecting the window until the participant moved the mouse again.
4. *Reach Destination* - From moving the mouse until the window was released at the destination.

Stages 2 and 4 were expected to account for most of the time participants took to perform a trial, and for both of these a Fitts' law index of difficulty (ID) was calculated using the Shannon formulation (Soukoreff & MacKenzie, 2004) (see Table 2). The ID is based on the distance to, and width of a target. The higher the ID, the more difficult it is to select a target (and the more time required to select it).

Based on the index of difficulties, we hypothesized that the manipulation layer interface would be faster overall for both the move and resize tasks. Also, that most of the difference would occur in the select target stage, because that was where the ID for the manipulation layer interface was lower. While it is clear that the target sizes are much larger for the manipulation layer, it is unclear whether the transparent target areas, and learning how to use it, will be detrimental to performance.

5.2 Results

Any trial where the participant failed at the first attempt to select the appropriate part of the window, or place the window within the outline area, was classed as an error, and excluded from the analyzes reported below. The maximum error rate recorded by one participant was 15.0%, with an average across all participants of 6.3%. Error rates were examined for each interface, the desktop style interface had an average error rate of 10.6% and the manipulation layer interface had an average error rate of 2.0%.

Participants' overall performance was analyzed using a mixed factorial analysis of variance (ANOVA) that treated the interface, task and block number as repeated measures, and the order in which the interface were used and tasks performed as between participants factors. This showed that participants performed the tasks significantly faster as the blocks progressed ($F(4, 16) = 5.74$, p=.005), although in percentage terms the performance improvement from Block 1 to 5 was small (less than 7% for each combination of interface and task). Participants performed the tasks significantly faster with the manipulation layer interface than the desktop interface ($F(1, 4) = 78.14$, p=.001), and performed the move task significantly faster than the resize task ($F(1, 4) = 41.84$, p=.003).

Our interest centers on participants' *trained* performance, once they were familiar with the interfaces and interacting on a Powerwall. For this, participants' mean performance in Blocks 4 and 5 was analyzed using mixed factorial ANOVAs, treating the interface and task as repeated measures. The order in which the interfaces were used and tasks performed were treated as between participant factors. Separate ANOVAs were used to analyze the times for the select target and reach destination stages of the task (see above). In neither ANOVA did the interface order or task order significantly affect performance, and there were no significant interactions.

Participants selected the window significantly faster with the manipulation layer interface than the desktop interface ($F(1, 4) = 160.95$, $p<.001$), and performed the move task significantly quicker than the resize task ($F(1, 4) = 37.77$, $p=.004$) (see Figure 5). For movement to the destination, there was not a significance difference between the interfaces ($F(1, 4) = 0.92$, $p=.39$) or tasks ($F(1, 4) = 2.45$, $p=.19$) (see Figure 5).

5.3 Discussion

Overall, participants performed window move and resize tasks significantly faster on the Powerwall with the manipulation layer than with a conventional desktop style interface, in agreement with our hypothesis. In percentage terms, moving a window took place 24% faster, and resizing took place 27% faster. The results also show that users reached a trained level of performance after only 48 trials, equating to approximately 9 minutes of use.

The results from the trained data show that, as Fitts' Law predicts, the difference between the manipulation layer and desktop interfaces occurred in the time taken to select a window rather than resize or move it to a new destination. This also matches our hypothesis.

6 Experiment 2: Power Lens

A within-participants design was used to compare the Power-Lens interface against an unaided desktop style interface for a data manipulation task on the Powerwall. Users performed a drag-and-drop task (similar to Chung (Chung, 2009)) with target sizes of 5x5 and 20x20 pixels representing the items of data. This corresponded with the narrowest width of the target regions in the move and resize tasks for desktop style interaction in Experiment 1. Similar experiments have used target sizes of 1 to 8 pixels (Ramos et al., 2007) and 4 to 32 pixels (Forlines, 2005). The order in which participants used the interfaces and the two target sizes was counterbalanced.

6.1 Method

6.1.1 Participants

The study was performed using eight participants. Six were male and two were female, their age ranged from 18 to 39 years (M = 26.3), with between 8 and 25 years of computer use (M = 14.9). All participants were right handed and none were affected by color blindness. Each was paid £7 (roughly US$10) since the experiment took slightly longer to complete than Experiment 1.

6.1.2 Materials

The experiment was run with exactly the same hardware configuration as Experiment 1. For the desktop style interface, participants had to select the target unaided. For the Power-Lens interface, participants had to use the lens described above.

6.1.3 Procedure

Each participant interacted with one target size (e.g., 20 pixels) first and then the other (e.g., 5 pixels) with one interface (e.g., unaided), and then performed the tasks in the same order with the other interface (e.g., Power-Lens). For each combination of task and interface, a participant performed five blocks of 16 trials. At the start of each combination, the participant was given a short amount of time to become familiar with both the interface and target size. Participants were given a 30 second rest between blocks, and a two minute rest between conditions. Overall, each participant took approximately one hour and twenty minutes to complete the experiment.

Each trial involved the same general procedure as the move task in Experiment 1. First, a participant moved to the center of the display, on top of a 4000x4000 pixel window containing a static image of a street map taken from a near by city (see Figure 6), increasing the realism of the task for a visualization application. After a two second delay, a yellow target (5x5 or 20x20 pixels) was drawn 1200 pixels away from the center of the display. As in Experiment 1, the same 16 starting positions, evenly distributed around the circle, were used for every block of trials. Again, the order in which they were used was randomized and seeded differently for each participant. To complete the task, the participant had to select the target (causing it to turn green in color) and drag it to a destination box. The destination had a tolerance of five pixels for both target sizes, measuring 15x15 pixels for the 5 pixel target, and 30x30 pixels for the 20 pixel target.

As in Experiment 1, each trial could be split into four stages. Again, the select target and reach destination stages were expected to account for most of the time participants took to perform a trial. Table 3 shows the Index of difficulty for both sections for all four conditions.

Our hypotheses were that the Power-Lens will significantly improve interaction, and will have a greater effect when used with the smaller (5 pixel) target.

6.2 Results

To remain consistent with the first experiment, any trial where the participant missed the target, or failed to release it correctly at the destination, was considered an error, and excluded from the analysis reported below. The maximum error rate recorded by any one participant was 6.6%, with an average across all participants of 3.8%. Error rates were examined for each interface, the unaided interface had an average error rate of 4.4% and the Power-Lens interface had an average error rate of 3.2%.

Participants' overall performance was analyzed using a mixed factorial ANOVA, which used the same factors as in Experiment 1. There was no significant change in performance as the blocks progressed ($F(1, 4) = 2.49$, p=.09). However, participants performed the tasks significantly faster with the Power-Lens than with the desktop interface ($F(1, 4) = 22.66$, p=.01), and performed the task faster with the 20 pixel target than the 5 pixel target ($F(1, 4) = 16.02$, p=.02).

As in Experiment 1, interest centered on participants' trained performance, for which the data from Blocks 4 and 5 were analyzed using the same types of mixed factorial ANOVAs as Experiment 1. Participants selected the target significantly faster with the Power-Lens than with the desktop interface ($F(1, 4) = 10.14$, p=.03), and performed significantly quicker with the 5 pixel target than the 20 pixel target ($F(1, 4) = 31.79$, p=.01) (see Figure 7). There was a significant interaction between interface and task ($F(1, 4) = 9.73$, p=.04), because the Power-Lens was faster than the unaided approach for the 5 pixel target, but there was little difference between the two interfaces for the 20 pixel target. For the time to reach the destination, there was no significance between the interfaces ($F(1, 4) = 0.83$, p=.41) or tasks ($F(1, 4) = 0.12$, p=.92) (see Figure 7).

6.3 Discussion

Overall, participants performed the task faster with the Power-Lens than with the desktop interface, matching our hypothesis. The results, however, show that the Power-Lens did not improve performance for the 20 pixel target. This indicates that the Power-Lens is only beneficial for high precision interaction.

The lack of significant improvement between the Power-Lens and unaided approach for movement to the destination suggests that the Power-Lens does not improve performance for 10 pixel targets. This also matches our hypothesis, and further concludes that the Power-Lens is suited to targets of 5 pixels or smaller, where the time difference between the two interfaces was 18%.

Verbal feedback, common amongst most participants, was that when using the desktop interface, interacting with the 5 pixel target caused users to strain their eyes and feel the effects of fatigue. The Power-Lens made it much easier for participants to see the target, and this benefit would increase if a Powerwall was used for hours of real work rather than a sequence of short experimental trials.

7 Conclusion

The aim of this research was to speed up interaction with both window data and the windows themselves on a Powerwall display. This was achieved by developing the manipulation layer for window manipulation, and the Power-Lens for precise target selection.

The manipulation layer increased target sizes by up to a factor of 80, reducing the time taken to select the target regions for moving and resizing a window. Experimental results found both tasks to be faster when using the manipulation layer, which would benefit any multi-window application running on a Powerwall.

The Power-Lens automatically magnified targets located at the region of interest. Results showed that the lens speeded up selection for small (5 pixel) targets, making it particularly beneficial to Powerwall applications that display very large amounts of data, which is what they are designed for.

Both the manipulation layer and the Power-Lens were implemented at the window manager level, which means they work automatically with any application, without it having to be modified. Finally, our recent field studies show that, when Powerwalls are used for visualization applications, users adjust what is shown on the display every few minutes and over the course of an hour hundreds of different selection actions are sometimes made (e.g., to move windows, adjust parameters and select data). Our new interaction techniques typically make each selection 1 second faster, not only saving a substantial amount of time, but also making interaction more fluid as a whole.

8 Acknowledgments

The authors would like to thank their colleagues who gave up valuable time to help evaluate the preliminary designs.

References

Appert, C., Chapuis, O., & Pietriga, E. (2010). High-precision magnification lenses. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 273-282). New York, NY: ACM.

Ball, R., North, C., & Bowman, D. (2007). Move to improve: promoting physical navigation to increase user performance with large displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 191-200). New York, NY: ACM.

Bederson, B. (2000). Fisheye menus. In *Proceedings of the Symposium on User Interface Software and Technology* (p. 217-225). New York, NY: ACM.

Bi, X., & Balakrishnan, R. (2009). Comparing usage of a large high-resolution display to single or dual desktop displays for daily work. In *Proceedings of* *the SIGCHI Conference on Human Factors in Computing Systems* (p. 1005-1014). New York, NY: ACM.

Bier, E., Stone, M., Pier, K., Buxton, W., & DeRose, T. (1993). Toolglass and magic lenses: the see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques* (p. 73-80). New York, NY: ACM.

Buxton, W., Fitzmaurice, G., Balakrishnan, R., & Kurtenbach, G. (2000). Large displays in automotive design. *IEEE Computer Graphics and Applications*, *20*, 68-75.

Carpendale, S., Ligh, J., & Pattison, E. (2004). Achieving higher magnification in context. In *Proceedings of the Symposium on User Interface Software and Technology* (p. 71-80). New York, NY: ACM.

Chung, L. (2009). Gaze-assisted pointing for wall-sized displays. In *Proceedings of Interact* (p. 9-12). Berlin, Heidelberg: Springer-Verlag.

Cox, D., Chugh, J., Gutwin, C., & Greenberg, S. (1998). The usability of transparent overview layers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 301-302). New York, NY: ACM.

Darling, E., Newbern, C., Kalghatgi, N., Burgman, A., & Recktenwald, K. (2004). An experimental investigation of magnification lens offset and its impact on imagery analysis. In *Proceedings of the IEEE Symposium on Information Visualization* (p. 215.5). Washington, DC: IEEE Computer Society.

Fitts, P. (1954). The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, *47*, 381-391.

Forlines, C. (2005). Zoom-and-pick: facilitating visual zooming and precision pointing with interactive handheld projectors. In *Proceedings of the Symposium on User Interface Software and Technology* (p. 73-82). New York, NY: ACM.

Furnas, G. (1986). Generalized fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 16-23). New York, NY: ACM.

Goodyer, C., Hodrien, J., Wood, J., Kohl, P., & Brodlie, K. (2009). Using high resolution displays for high resolution cardiac data. *Philosophical Transactions of the Royal Society*, *367*, 2667-2677.

Gutwin, C. (2002). Improving focus targeting in interactive fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 267-274). New York, NY: ACM.

Gutwin, C., & Skopik, A. (2003). Fisheye views are good for large steering tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 201-208). New York, NY: ACM.

Harrison, B., Ishii, H., Vicente, K., & Buxton, W. (1995). Transparent layered user interfaces: an evaluation of a display design to enhance focused and divided attention. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 317-324). New York, NY: ACM.

Hutchings, D., Smith, G., Meyers, B., Czerwinski, M., & Robertson, G. (2004). Display space usage and window management operation comparisons between single monitor and multiple monitor users. In *Proceedings of Advanced Visual Interfaces* (p. 32-39). New York, NY: ACM.

Jedrysik, P., Moore, J., Stedman, T., & Sweed, R. (2000). Interactive displays for command and control. In *Proceedings of the IEEE Aerospace Conference* (Vol. 2, p. 341-351). Los Alamitos, CA: IEEE Computer Society.

Keuning, H., Galen, G. van, & Houtsma, A. J. (2005). The role of size of an augmented force field in computer-aided target acquisition tasks. *International Journal of Human-Computer Interaction*, *18*(2), 219 - 232.

Lane, D., Peres, S. C., Sndor, A., & Napier, H. A. (2005). A process for anticipating and executing icon selection in graphical user interfaces. *International Journal of Human-Computer Interaction*, *19*(2), 241-252.

Murata, A. (1998). Improvement of pointing time by predicting targets in pointing with a pc mouse. *International Journal of Human-Computer Interaction*, *10*(1), 23.

Murata, A. (1999). Extending effective target width in fitts' law to a two-dimensional pointing task. *International Journal of Human-Computer Interaction*, *11*(2), 137.

NASA. (2008). *Hyperwall-2: press release.* `http://www.nasa.gov/centers/ames/news/releases/2008/08_57AR.html` Last visited in January 2010.

Phillips, J. G., Meehan, J. W., & Triggs, T. J. (2003). Effects of cursor orientation and required precision on positioning movements on computer screens. *International Journal of Human-Computer Interaction*, *15*(3), 379-389.

Pietriga, E., & Appert, C. (2008). Sigma lenses: focus-context transitions combining space, time and translucence. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 1343-1352). New York, NY: ACM.

Ramos, G., Cockburn, A., Balakrishnan, R., & Beaudouin-Lafon, M. (2007). Pointing lenses: facilitating stylus input through visual-and motor-space magnification. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 757-766). New York, NY: ACM.

Robertson, G., Czerwinski, M., Baudisch, P., Meyers, B., Robbins, D., Smith, G., et al. (2005). The large-display user experience. *IEEE Computer Graphics and Applications*, *25*, 44-51.

Soukoreff, R., & MacKenzie, S. (2004). Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. *International Journal of Human-Computer Studies*, *61*, 751-789.

Treanor, D., Owers, J., Hodrien, J., Quirke, P., & Ruddle, R. (2009). Virtual reality powerwall versus conventional microscope for viewing pathology slides: an experimental comparison. *Histopathology*, *5*, 294-300.

Ware, C., & Lewis, L. (1995). The dragmag image magnifier. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (p. 407-408). New York, NY: ACM.

Worden, A., Walker, N., Bahrat, K., & Hudson, S. (1997). Making computers easier for older adults to use: area cursors and sticky icons. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*

(p. 266-271). New York, NY: ACM.

Table 1: Classification of lens implementations. Cursor = Cursor-tracked; Manual = Manually Activated; N/A indicates information that is not available.

Lens	Citation	Projection	Position	When lens appears	Semi-transparent
Furnas	(Furnas, 1986)	Fisheye	N/A	N/A	N/A
Speed Coupled Flattening	(Gutwin, 2002)	Fisheye	Cursor	Permanent	No
Steering Lens	(Gutwin & Skopik, 2003)	Fisheye	Cursor	Permanent	No
Idelix Lens	(Carpendale et al., 2004)	Fisheye	Stationary	Permanent	No
Ring	(Appert et al., 2010)	Fisheye	Stationary	Permanent	No
Fisheye Menu	(Bederson, 2000)	Fisheye	Cursor	Permanent	No
Zoom-and-Pick	(Forlines, 2005)	Fisheye	Cursor	Permanent	No
Key	(Appert et al., 2010)	Fisheye	Cursor	Permanent	No
Speed	(Appert et al., 2010)	Fisheye	Cursor	Permanent	No
Trailing Lens	(Ramos et al., 2007)	Linear	Cursor	Permanent	Yes
Offset Lenses	(Darling et al., 2004)	Linear	Cursor	Permanent	No
Blending	(Pietriga & Appert, 2008)	Linear	Cursor	Permanent	Variable
Speed-Coupled Blending	(Pietriga & Appert, 2008)	Linear	Cursor	Permanent	Variable
Pressure Activated	(Ramos et al., 2007)	Linear	Stationary	Manual	Yes
DragMag	(Ware & Lewis, 1995)	Linear	Stationary	Permanent	No
Magic Lens	(Bier et al., 1993)	Linear	Stationary	Permanent	Yes
Transparent Lens	(Cox et al., 1998)	Linear	Stationary	Permanent	Yes
Power-Lens		Linear	Stationary	Automatic	Yes

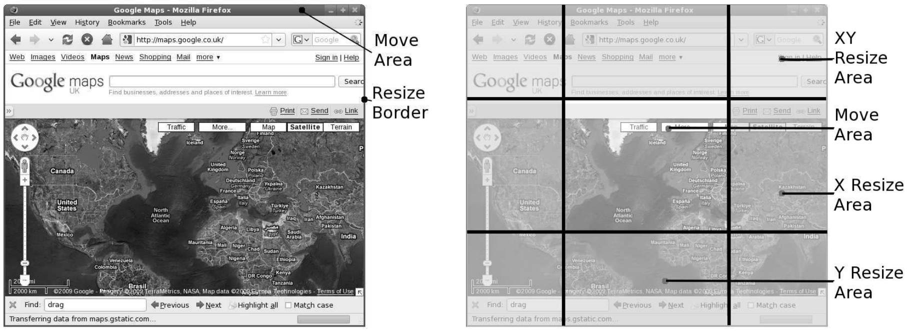
Stage/task	Interface	
	Desktop	Manipulation layer
2. Select (Move task)	W:20 ID:5.93	W:400 ID:2.0
2. Select (Resize task)	W:5 ID:7.91	W:400 ID:2.0
4. Reach Destination (Both tasks)	W:240 ID:3.46	

Table 2: The Index of Difficulty (ID) for each condition of Experiment 1, calculated from the width (W) of the target and the distance moved (1200 pixels for selection vs. 2400 pixels for movement to the destination).

Stage/task	Interface	
	Desktop	Power-Lens
2. Select (20 pixels)	W:20 ID:5.93	W:134 ID:3.32
2. Select (5 pixels)	W:5 ID:7.91	W:33.5 ID:5.2
4. Reach Destination (Both tasks)	W:10 ID:7.91	W:67 ID:5.2

Table 3: The Index of Difficulty (ID) for each condition of Experiment 2, calculated from the width (W) of the target and the distance moved (1200 pixels for selection vs. 2400 pixels for movement to the destination).

FIGURE 1: A 54 million pixel Powerwall made of 28 x 20-inch TFT monitors. The wall measures 3.02m by 1.32m.



(a) A desktop window

(b) The manipulation layer

FIGURE 2: (a) Move and resizing on a standard window. The move area and resize border are only 20 and 5 pixels thick, respectively. (b) The manipulation layer. The resize areas extend 400 pixels from the edge of the window. The move area fills the space in the middle, which is 800 x 400 pixels for window that occupies one whole monitor (the smallest sized window used in the Powerwall applications we have built). The layer is fully transparent to the user, and highlighting is only used for illustrative purposes.

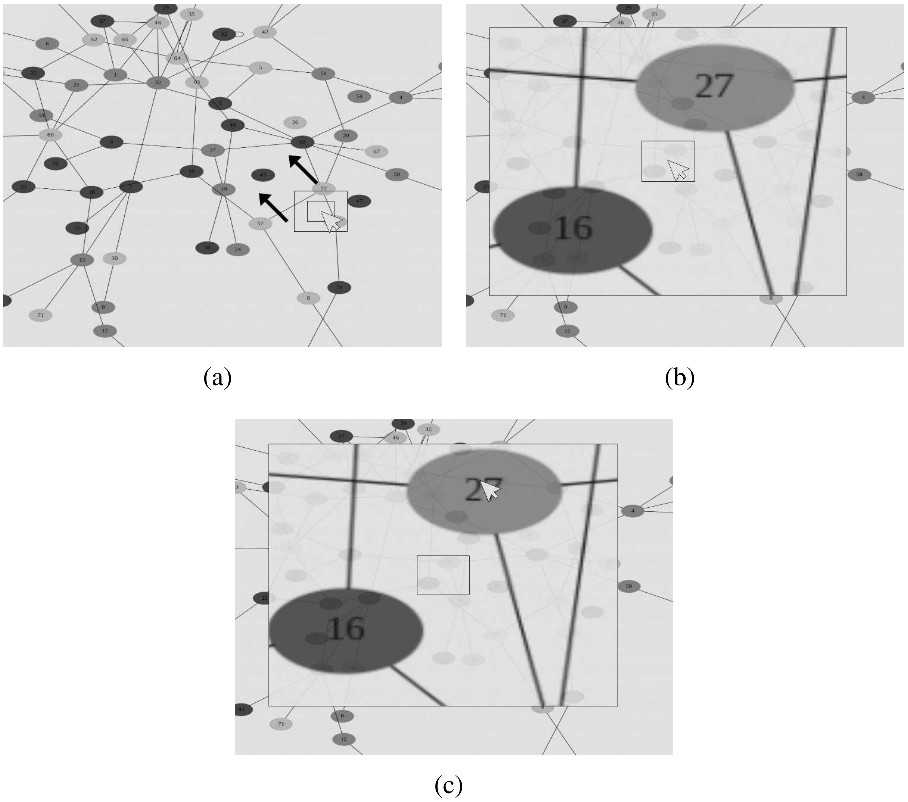
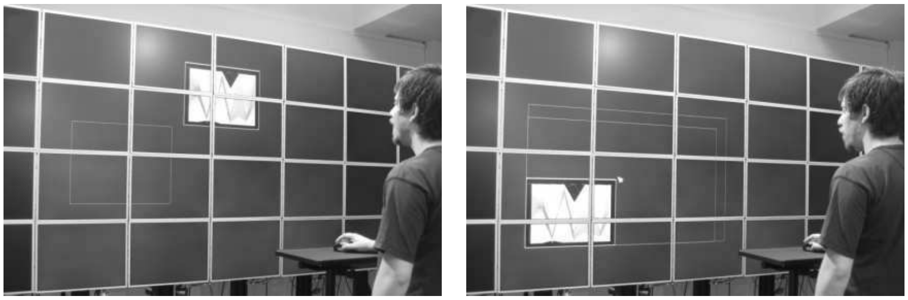


FIGURE 3: The Power Lens used on a cluttered graph. In this example, the user wants to select Node 27. (a) The user moves the cursor quickly toward node 27, so the lens is inactive. (b) The user slows the cursor as they approach node 27, so the lens automatically appears. (c) The lens appears on top of the magnified Node 27. If necessary the user may adjust the cursor position by moving it freely within the lens.



(a) The move task (b) The resize task

FIGURE 4: The starting positioning of the window and destination outline for (a) the move task and (b) the resize task.

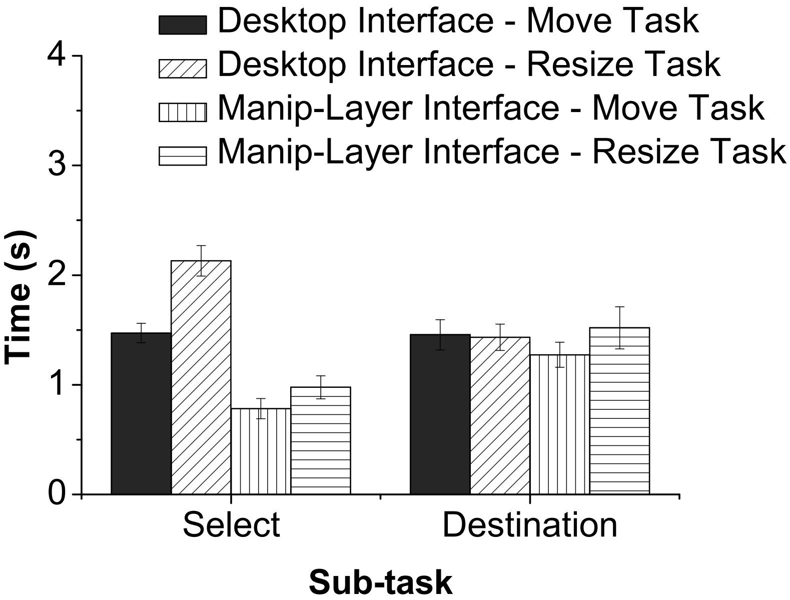


FIGURE 5: Experiment 1 - The mean time taken in blocks 4 & 5 to select the window and move/resize it to the destination. Error bars show the standard error of the mean.

Rooney, C. & Ruddle, R. (2011) Improving window manipulation and content interaction on high resolution, wall-sized displays. International Journal of Human-Computer Interaction, iFirst

FIGURE 6: A participant performing the data manipulation task on the Powerwall. The participant has reached the target and the lens has been activated, presenting a magnified target to select.

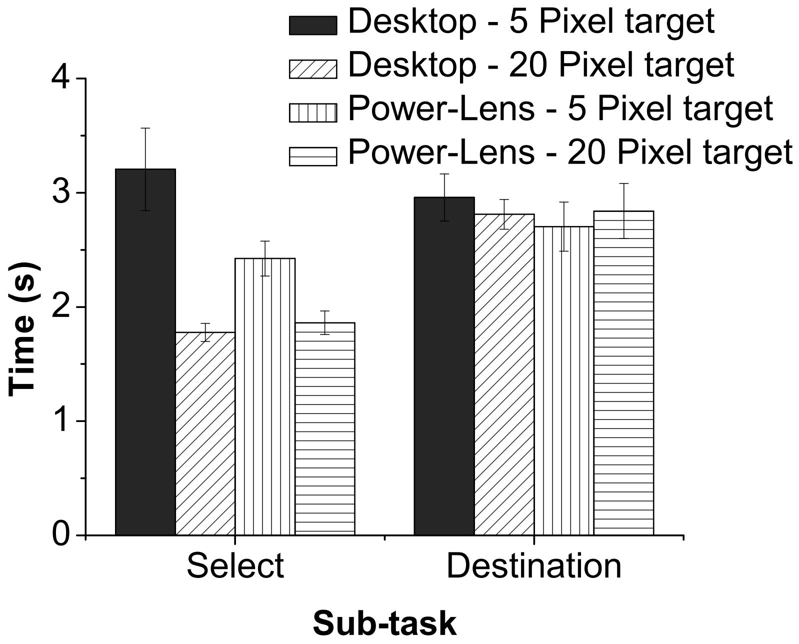


FIGURE 7: Experiment 2 - The mean time taken in blocks 4 & 5 to select the target and move it to the destination. Error bars show the standard error of the mean.