Literature Survey on Interaction Techniques for Large Displays

Brian Badillo bbadillo@vt.edu
Doug A. Bowman bowman@vt.edu
William McConnel wmcconne@vt.edu
Tao Ni nitao@vt.edu
Mara Silva mara@vt.edu

1 Introduction

There has always been a desire for large screen systems. Large screens can easily be used for public media. They are also desirable because they can cover a larger visual angle for information. As screens become bigger, we also try to increase resolution to maximize the visual information bandwidth. With a larger capacity for visual information, these large screens become attractive for numerous applications like information visualization, visual analytics, and computer-supported collaborative applications.

These days, large screen displays come in many forms. A simple way to create a large display is to use a high definition projection screen. However, there is a resolution limit in commodity projectors. If the projection was made big enough, the resolution would deteriorate. Another way to create large displays is to tile smaller commodity displays to act as one. As the cost of commodity display systems comes down, the ability to tile these displays increases. However, with a visible bezel is an inherent problem with tiled displays. To combat this, tiled projection screens have been developed to minimize this bezel and make it virtually disappear. In any case, large screen displays are fast becoming an attractive solution for visual applications.

With increased popularity of large displays, come a slew of interaction issues. These include less effective cursor tracking, target acquisition, multiple selection, and target manipulation.

2 Large-display interaction challenges

Large displays, either in form of multimon or projector array [Li et al. 2000], affords users a dramatically increased screen real estate to operate. A longitudinal study [Ball and North 2005] has shown that users take advantage of large screen space by launching much more windows and keeping them viewable at once, rather than managing only a few applications at a time on single monitor desktop. Czerwinski et al also carried out a preliminary user study to provide evidence about increased productivity on very large displays for complex, multitasking office work [Czerwinski et al. 2003].

Meanwhile, it has been recognized that large displays bring HCI professionals a new set of interaction challenges, from fundamental selection and manipulation, to high-level window and task management. Conventional Graphical User Interface (GUI) metaphors such as Windows, Icons, Menus, and Pointing (WIMP) do not always scale well to large-format displays [Ni et al. 2006]. To better understand what factors have left traditional interaction techniques awkward to use on large displays, we identify several major usability issues as follows.

1. Tracking the cursor. Visually tracking the cursor on very large displays becomes more difficult [Ni et al. 2006; Robertson et al. 2005]. A cursor is a standard interface element for pointing, selecting and manipulating, and thus keeping track of where a cursor is located on a display is a crucial visual feedback for other interaction tasks to succeed. With increased display size, users tend to employ higher mouse acceleration to traverse a large display with a reasonable speed [Baudisch et al. 2003b]. The faster the cursor moves, however, the higher the chance is to lose visual track of it, since the cursor appears to jump from here to there. It has been reported that losing the cursor when interacting with large displays is an annoying effect that reduces users' task efficiency and satisfaction [Baudisch et al. 2003b]. In certain circumstances, it is even problematic to locate a stationary cursor on large displays [Khan et al. 2005].

2. Large screen target acquisition. As display size grows, accessing icons, windows, and other objects is a difficult and time-consuming task, since they often reside farther away on large displays, compared to a desktop monitor [Bezerianos and Balakrishnan 2005; Ni et al. 2006; Robertson et al. 2005]. With standard interaction techniques, it requires dragging a cursor across large distance to reach distal display content. For large displays with touch-screen or pen-based interaction capability (which are appealing in manner of up-close interaction), it is hard to access data out of our arm's reach. It is further complicated when a user has to access multiple objects scattering over a large display.

3. Task management. There is a lack of effective task management mechanisms for large displays. In fact, task management has been a challenging research theme in HCI community for many years, and even in modern operating systems such as Windows and Mac OS X, task management is still not well supported. As Robertson et al pointed out [Robertson et al. 2005], an effective task management should provide convenient means that allow users to group sets of windows, layout the groups and windows on the screen, and switch between groups and windows. Current window management systems (e.g. Apple Expos‘ and Windows Taskbar) are application- and window-oriented, rather than task-oriented. Users, however, are relatively easy to handle multitasking on a desktop monitor, since they normally do not open too many windows at once. In contrast, multitasking becomes more common for large display users [Czerwinski et al. 2003], and thus effective multi-window task management is critical to a productive and enjoyable user experience.

4. GUI widgets for large displays. Widgets found on current desktop GUI have already been standardized. Drop-down menus, radio buttons, tabs, and check boxes are just a few examples. While they work well in mouse- and keyboard-based user interface, problem occurs when it comes to large displays, especially when non-traditional input devices are used. For example, drop-down menus are probably not optimal for wand-like devices, since they do not afford as accurate operation as mice do.

5. Touch screen. Researchers have worked on equipping touch-sensitive capability on large displays. An obvious advantage is that users are able to work with large displays in manner of up close touch screen interaction. There are certain scenarios where interaction from a distance is ideal, such as sorting photos, reading papers, and navigating a high-resolution map [Vogel and Balakrishnan 2005]. With touch-sensitive large displays, however, several interaction challenges emerges. Up close interaction is not always desirable by large-display users, since in some circumstances, they have to step back and overview the screen content. With current user interface and input devices, transition between interaction from a distance and up close interaction is poorly supported. Also, target acquisition as aforementioned is particularly problematic on touch-sensitive large displays, as selecting an object residing beyond arm's reach with a stylus is unconceivable. In addition, touch-

sensitive large displays may introduce severe fatigue in extended period of use, since most operations require significant amount of hand, arm, and even body movement.

3 Tracking the cursor

The increased distances on large displays create some problems for users. In order to avoid clutching, usually users increase the mouse speed or acceleration. When moving the cursor over large distances across the display, the cursor's high speed makes it hard for users to visually keep track of its position.

Adding to this problem, sometimes the cursor visually jumps from one position to another. This is called *temporal aliasing* [Dachille and Kaufman 2000]. This happens because the visual representation of the cursor is updated with the monitor refresh rate. When the cursor is moving at high speed, the decreased refresh rates of monitors makes the apparent gaps in the path of the mouse get increasingly larger.

On these situations the user has to spend some time detecting and reacquiring the cursor position. Since it is hard for the user to visually track the cursor position, predict its trajectory and detect it as it approaches the desired target, these problems limit the speed at which the users can reliably operate the mouse.

Strategies to solve this problem have focused primarily on two goals: (1) enhance the detection of cursor position; (2) increase the predictability of the cursor's trajectory [Baudisch et al. 2003b].

Microsoft Windows mouse trail enhances the visibility of the cursor's path. It achieves this effect by leaving the cursor image on the screen for two or more frames, which produces the appearance of a trail following the cursor. The problem with this technique is that it gives the user the undesirable perception that the cursor doesn't stop when the user stops the movement, since the trailing cursor images continue to move. This creates a responsiveness problem.

The *high-density cursor*, presented by Baudisch et al [Baudisch et al. 2003b], increases the visual continuity between images of the cursor, as shown in figure 1. The denser cursor track helps users to effortlessly detect the cursor position and extrapolate its path. This technique also features an optional cursor scaling governed by its speed. Unlike Windows mouse trail, the high-density cursor avoids responsive problems by creating a new set of cursor images for each frame. A direct comparison of the two techniques is shown in figure 2.

Many of the application for large displays involve presentation of data and results to an audience. The presenter often desires to call the attention of the audience to a particular area of the screen. In order to use the cursor to do this, its position should be easily detected. On a wall-sized display, the size of the cursor is very small relative to the display size, so the presenter and the audience can easily lose track of the cursor position.

Khan et al presented the *Spotlight* technique to direct the audience's attention on large wall-sized displays [Khan et al. 2005]. The metaphor is based on how spotlights are used on theatric productions. When the Spotlight technique is in use, the display will be darkened, with exception of the region of the spotlight. The spotlight consists of cursor position, spotlight interior region, and spotlight exterior region. The cursor position is represented by the arrow present in the most GUIs. The spotlight interior region is a fully transparent circle around the cursor position. The display outside the interior region has its luminosity reduced by about 75%, which draws the attention of the audience to the illuminated area.

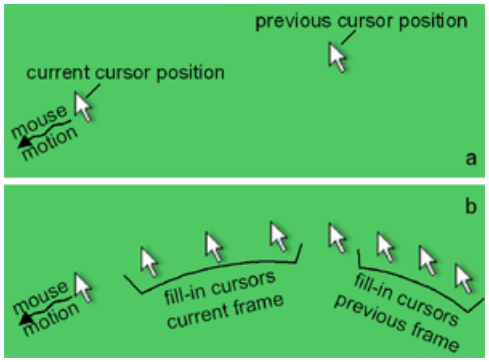

Figure 1: Advantage of *high-density cursor* over regular cursor. (a) The problem: at high mouse speeds, the mouse cursor seems to jump from one position to the next. (b) *High-density cursor* makes the mouse cursor appear more continuous by inserting additional cursor images between actual cursor positions. [Baudisch et al. 2003b].

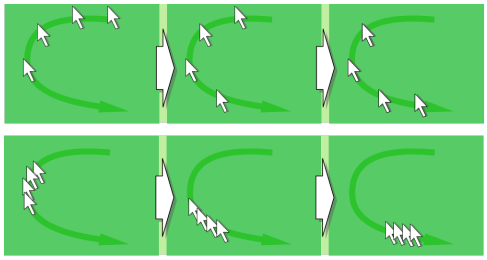

Figure 2: (top row) Three successive screenshots from a mouse movement enhanced with the Windows mouse trail. Trailing cursor images lag behind. (bottom row) The same traversal with high-density cursor; additional cursor images are inserted *between* the two most recent cursor positions, resulting in a dense and lag-free trail. [Baudisch et al. 2003b].

The spotlight also works as a tracking menu, which means that the cursor position can be moved freely while in the interior region the spotlight, being able to point to something inside the spotlight. When the cursor position moves beyond the edge of the region, the cursor point drags the spotlight to a new position. Another option of the technique is to present a 'beam of light' coming down from the center of the display, called *Searchlight*. This is especially useful when the spotlight is not in viewer's field of the view, because he can follow the beam to locate the spotlight. Figure 3 shows the technique in use.

The *Prince* technique [Zhai et al. 1994] is based on Fitts' law [Fitts 1954; MacKenzie 1992] and treats the object to be selected as a point, and the cursor as a region. The *Prince* technique and many other *area cursor* techniques that followed usually represent the cursor as a square or a circle. Even though the main objective of these techniques is to improve target acquisition, it has to be mentioned that because of the larger than usual cursor size, these techniques also help improve detection of cursor position. In fact, most of the techniques that address the problem of target acquisition will also end up improving tracking of cursor position and predictability of its trajectory. Many of these techniques will be discussed further on section 4.

4 Large screen target acquisition

Many researchers have seen fit to simply extend or modify certain aspects of WIMP-style interfaces to make them more usable for large-screen displays. Most accomplish this by using Fitts' Law

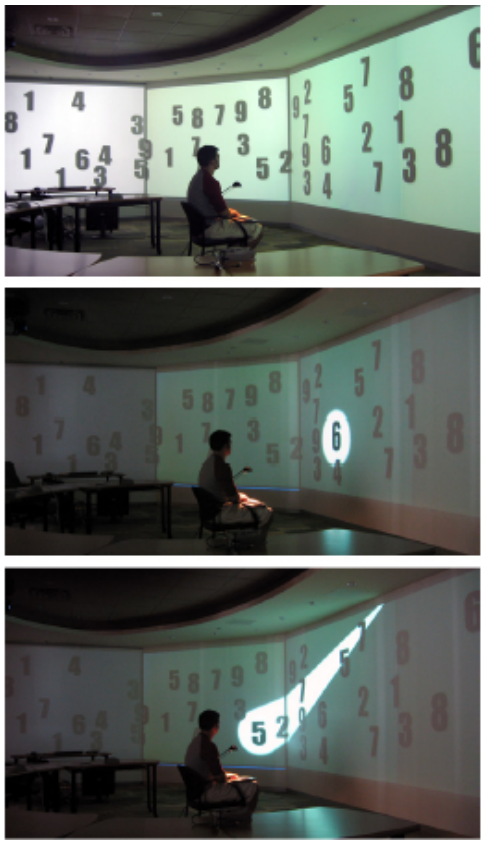

Figure 3: A number on the screen is being indicated by the cursor. (Top to Bottom) Regular cursor, spotlight, and searchlight. [Khan et al. 2005]

as a basis for hypotheses, while others seek creative extensions of currently employed 2D interface metaphors, and still others merely present novel interfaces. Typically, there are three problems in large screen interaction that these researchers address: fast target acquisition, multiple selection, and target manipulation.

Fitts' Law states that the time it takes to maneuver towards a target is a logarithmic function of the distance of the target divided by the width [Fitts 1954; MacKenzie 1992]. Researchers can vary the parameters of this model to minimize pointer movement time. One way to do this is to increase the effective width of the target. McGuffin and Balakrishnan explored this idea by evaluating the effects of expanding targets[McGuffin and Balakrishnan 2005]. Their work confirmed that dynamic enlargement of the targets continued to net results in accordance with Fitts' Law. It was also found that positive effects of target expansion could still be seen even if the target enlargement took place in the last ten percent of the entire cursor movement. Conversely, the effective target width can be increased simply by enlarging the cursor. For instance, Grossman and Balakrishnan developed the *Bubble Cursor* [Grossman and Balakrishnan 2005] to improve target acquisition by dynamically increasing the width of the cursor to always include only one target at a time. Still other researchers have tried to decrease the distance to the target to optimize the index of difficulty in Fitts' Law. Baudisch et al developed the *drag-and-pop/pick* techniques [Baudisch et al. 2003a] to bring copies of targets closer to the cursor dynamically. Guiard et al [Guiard et al. 2004] went about the same idea in a different way by using a *timid cursor* that did not ever pass over desktop space that wasn't occupied by a target.

Target acquisition is a thriving area of research in the large screen interaction community. Single target selection is the simplest case of target acquisition. Baudisch et al developed the drag-and-pop and drag-and-pick interaction techniques for fast target acquisition[Baudisch et al. 2003a]. They present the problem as being for more than just large screen displays but also for screens that support different modes of input. For example, this technique has the potential to be highly usable on desktop systems that span between displays that support pen-based interaction and displays that don't. In this case, with normal pointing methods, a user would have to switch between a mouse and a pen quite often or lose the advantages of using the pen and only use the mouse. The drag-and-pop technique takes an initial movement direction of the mouse while dragging a target and brings a copy (or proxy) of all selectable objects in that direction to within a certain proximity of the actual cursor. To facilitate association of the proxies with the actual objects they represent, a line of association (or rubber band) is drawn. The drag-and-pick technique is identical except that it allows you to pick the selected object instead of dragging a selected object. However, these techniques contribute to higher selection error rates. This technique also requires the system to create and redisplay proxies for items on the desktop making implementation more complex. But in the end, these techniques do provide significant benefits to target acquisition on large screen displays.

Bezerianos and Balakrishnan [Bezerianos and Balakrishnan 2005] argue that there are some major shortcomings in the drag and pop/pick technique. Target identification completely depends on the correctness of the initial vector drawn by the user. This makes it easy for the user to accidentally specify the wrong direction to create proxies. In addition the number and size of the proxies is fixed. This makes it hard for a user to get good performance in the sparse or dense spaces. Having a fixed size also clutters the layout of the proxies around the cursor. Multiple operations are not supported in one context of proxies. That is, multiple drag operations need to be performed in order to select multiple targets because the proxies are only created if an initial vector points in their general direction. In response, Bezerianos and Balakrishnan present and evaluate a technique for use on large screen displays that extends on the basic idea of drag and pop/pick technique called the *Vacuum* [Bezerianos and Balakrishnan 2005]. First, their research identifies guidelines in designing such a technique. Techniques should enable transient use, minimize physical movement, be predictable and consistent, be transparent, and be flexible. The Vacuum technique is very similar to the drag and drop/pick technique in that proxies are created. But the technique implements some extensions that address the shortcomings mentioned above. The Vacuum technique makes the arc of influence adjustable with simple pen drag strokes. An illustration of the technique is shown on figure 4. The Vacuum technique also makes use of pen hovering functionality, adding another degree-of-freedom for the user to more effectively control operations. By holding the pen over the surface, a user can perform multiple pen-down pen-up operations without having to dismiss the Vacuum. The Vacuum scales proxies and places them in a scaled layout so that the display of proxies makes sense and overlapping is minimal. Some drawbacks to the technique are the small proxies that it may produce. However, users liked the Vacuum technique over direct picking because, although it may have had more overhead time, it was more comfortable than physically reaching or moving. It is important to note that small changes on an existing technique such as the changes they made on the existing drag and drop/pick technique may net greater changes on usability.

However, with the above selection techniques, the layout of the targets is changed. Implementing such a system might involve a bit more than just superficial changes on the pointer used. An implementation would have to have access to the bitmaps that are used to represent the targets and would need to redisplay these at vari-

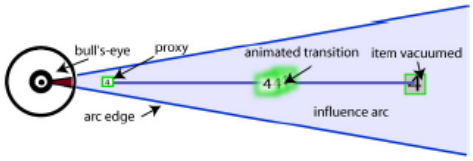

Figure 4: The Vacuum. [Bezerianos and Balakrishnan 2005].

ous points on the screen. In addition, the user's ability to use spatial memory to recall a target may be diminished because the placement of proxies around the cursor is not consistent with the placement of the original targets. To overcome these issues, a solution with the cursor rather than the layout of targets must be sought after.

Guiard et al [Guiard et al. 2004] present *object pointing*. The basic problem that object pointing addresses is the fact that cursors only point at one pixel at a time rather that one meaningful (or clickable) object at a time. Object pointing introduces a "void-phobic" cursor that does not travel in any space between selectable objects. Instead it travels continuously within object space until a boundary is met at which point it determines which object to jump to next by current trajectory. Although this does cut down the distance a pointing device must travel to select targets, it also makes the pointer difficult for the user to track. This technique has been proven more effective than normal pointing techniques as the index of difficulty increases or as the object density decreases. It also allows the layout of interfaces to remain unchanged. Object pointing has a smaller footprint for input devices than normal pointing techniques. Guiard et al argue that this is advantageous for the obvious reason that sometimes users have limited space for movement or that less physical movement is less tiresome. However, a smaller footprint also means less buffer space for error. This leads to more needed movement accuracy for this technique.

As an alternative, Grossman and Balakrishnan present the Bubble cursor [Grossman and Balakrishnan 2005], a continuously moving cursor that attempts to optimize Fitts' law over large distances for use on large screen displays. It is closely related to object pointing in that an object is always selectable by it. However, the footprint of the input device is not shrunken by this technique. As a result, accuracy with the device is not affected by this technique. The bubble cursor works by simply enlarging the selection area around the actual position of the cursor so that it is always big enough to encompass exactly one item, as illustrated in figure 5. It improves on previous area cursors by making it automatically resizing and circular. A major advantage of using cursors that span large areas is that they become easy to track. As with many target acquisition techniques, this one is also currently limited to single object selection. As with object pointing, an advantage of the Bubble cursor is that the interface layout remains unchanged. Gains from using the bubble cursor however deteriorate as target density increases just as in object pointing. As is the case with object pointing, when the density of targets increases, the distance the cursor has to travel to the desired target also increases and performance approaches that of a normal cursor on large screen displays.

Other more creative methods of target acquisition are found when researchers move away from traditional pointing devices, such as the mouse or pen, and use other kinds of input. Such input methods include laser pointing, eye tracking, and hand tracking. It is interesting to note that a major difference between a mouse and these other methods of interaction is that the other methods allow so-called random access to all areas of the screen while with a mouse and other relative positioning techniques one is forced to be constrained by the current position of the pointer on the screen. Vogel

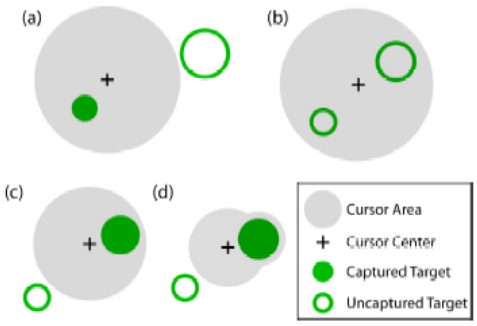

Figure 5: (a) Area cursors ease selection with larger hotspots than point cursors. (b) Isolating the intended target is difficult when the area cursor encompasses multiple possible targets. (c) The bubble cursor solves the problem in (b) by changing its size dynamically such that only the target closest to the cursor centre is selected. (d) The bubble cursor morphs to encompass a target when the basic circular cursor cannot completely do so without intersecting a neighboring target. Note that the same legend is used for Figures 1 to 5. [Grossman and Balakrishnan 2005].

and Balakrishnan define a set of desirable design characteristics for pointing devices on large displays [Vogel and Balakrishnan 2005]. First, they believe the device should be accurate. Touch screens and absolute device mappings on large screens may heavily detract from the level of accuracy users can obtain. Second, the device should be easy and fast to acquire as well as maneuver. A device that takes a long time to find, grip, or move will impose an undesirable overhead on any task that a user is trying to perform with it. Next, the device should be comfortable to use and should not contribute to high fatigue levels or be overly obtrusive. And lastly, the device should be highly usable up-close to the screen and far away to encourage a users physical movement.

Malik, et al [Malik et al. 2005] have created an interface for interacting with large displays that allows fast and accurate random access to the entirety of a large display system. Their method leverages asymmetric dependent tasks. They have divided the area of a touchpad into two separate regions that are mapped to different parts of the screen. One region (the non-dominant side) of the touch pad is mapped to the entire screen. Interactions with this part of the touchpad result in the placement of a context workspace on the large display. The other region (the dominant side) of the touchpad is then mapped to the context workspace. This mapping is shown on figure 6. This enables a user to coarsely and quickly randomly access any part of the screen with their non-dominant hand while still providing a fine positioning mechanism with the dominant hand. The important contribution here is that the asymmetric touchpad partition presented effectively augments the fast random access capabilities of a touch screen onto a large display area while not sacrificing precision and accuracy. Furthermore, the touchpad is a device that imposes little time to acquire and is not obtrusive. Another facet of the work by Malik et al is that it combines a touchpad with gestural input by using computer vision technologies. By using gestures the device becomes even less obtrusive to the user. Even still, the touchpad is designed for a user to be sitting at predefined distance from the screen which heavily constrains the users physical movement in front of the screen.

Cao and Balakrishnan put forth a design principle that makes using passive gestural input attractive[Cao and Balakrishnan 2003]. Postures and gestures make up an inferred set of actions that should be meaningful and easy to understand to the user. These meaningful gestures should allow for eyes-free operation leaving the user free to concentrate on other aspects of interaction [Cao and Balakrishnan 2003]. Using this principle as a premise, they define an

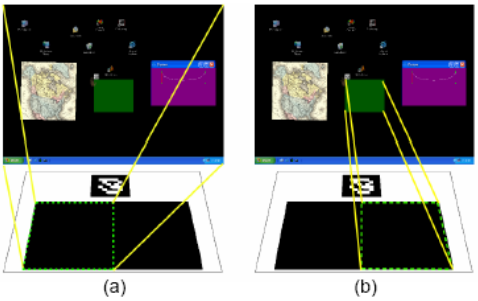

Figure 6: Touchpad mapping for asymmetric interactions for: (a) the left hand; (b) the right hand. [Malik et al. 2005].

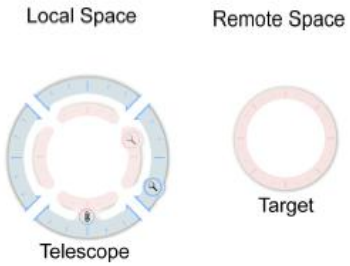

Figure 7: The Frisbee widget. [Khan et al. 2004].

interaction technique using a passive wand tracked in 3D. The *VisionWand* [Cao and Balakrishnan 2003] accomplishes the goal of being a low-cost alternative to other tracked devices in that it is a passive device, and that it uses computer vision and two commodity cameras for five degrees-of-freedom tracking. The VisionWand is a simple device that simplifies gesture recognition in comparison to freehand gestures. Because the wand has a shape that does not change, it is easier to process and recognize the movements of the device. To select an object with the VisionWand, a user simply needs to point at the object with the wand and make a tapping gesture. However, as with all gestural input, there is room for error in determining what is input and what is random movement.

Simple freehand gestures for pointing and clicking have been developed by Vogel and Balakrishnan [Vogel and Balakrishnan 2005]. These gestures are simple gestures that approach pointing and clicking in very different ways. For pointing, one obvious and natural way to indicate a point on the screen is through raycasting. However, much like laser pointing, the slightest bit of hand tremor can cause quite a bit of jitter at the point of selection. The authors recognize this and propose relative pointing and clutching. In this way, a user can move the cursor around in a precise and continuous way, while still being able to clutch (by using a closed hand posture) to continue cursor movement in a certain direction. This enables more accurate and less jittery cursor movements. Seeing that there are advantages and disadvantages to each of these pointing techniques, a hybrid between the two is also proposed. In the raycasting/relative pointing hybrid, users can point to the general area of context they wish to work in, and then move their hands through the air much like using a mouse to accomplish more precise cursor movement. This technique makes it closely related to the method developed by Malik et al except that it imposes a mode switch on the user. To address the issue of clicking, the authors also developed two methods. The first method imitates a tapping motion, like one would perform on a touchpad with their index finger. This method is intuitive as it is used on many other interfaces. However, when performing a tap in the air a user does not have a definite stop point for kinesthetic feedback. In addition, this kind of movement may make it hard for a user to keep their hand still. So the second method remedies this by using a trigger thumb click. By using a sideways thumb click, users can obtain kinesthetic feedback on how far to move the thumb before a click is perceived. Both clicking techniques are well-received and could be used in conjunction to be consistent with left and right mouse button clicks. On a side note, they also make clever use of visual and auditory feedback to facilitate usability.

Although there is a multitude of work being done in single target selection on large displays, many researchers neglect to address the problem of multiple object selection on large displays. For some techniques such as area cursors or the bubble cursor, parallel multiple selection can be and is addressed with some ease due to the wide area nature of the cursor. However, this still does not address the need for selecting multiple targets in different areas of the screen. With other techniques such as the drag-and-pick, object pointing, and VisionWand techniques, multiple selection is often left for future work as an unanswered problem. Since these techniques are designed for fast single target acquisition, a quick and easy fix for making multiple selections would be to add a mode for making multiple serial selections. Malik et al deal with multiple random selections simply by providing an alternate gesture for selection and keeping a last-in first-out queue of multiply selected objects [Malik et al. 2005]. This makes random multiple selections easy and provides a means for users to drop items of the collection somewhat independently from each other. But this method of multiple selection requires a display of icons that easily becomes cluttered as the number and size of grabbed items increases. In the end, the effectiveness of multiple target acquisition methods on large screens has not yet been studied in great detail.

A lot of work in large screen display interaction deals with fast target acquisition. But there is still the issue of what to do with targets after they have become selected. Not quite as much attention has been put into manipulation tasks on large screen displays. The drag-and-pop and Vacuum techniques somewhat address target manipulation by allowing users to drag some items on top of other items[Baudisch et al. 2003a][Bezerianos and Balakrishnan 2005]. However, the technique does not allow arbitrary placement of items. To address this, Azam Khan et al have created a remote control interface widget they call a *Frisbee* [Khan et al. 2004]. A Frisbee is composed of two parts: the telescope and the target, as shown in figure 7. The telescope is the part of the widget that a user interacts with. The telescope is merely a portal to another part of the screen that is represented by the target widget. The target widget is simply visual feedback for users to see what is currently being remotely operated on. The important aspect of Frisbees is that they enable users to manipulate to and from remote areas of the screen within a reachable area. The widget acts as a kind of portal or hyperlink between two areas of the screen. Major shortcomings of this technique are its lack of effort to bring together fast target acquisition with remote manipulation. That is, the method for moving the target or the telescope widgets is rather primitive and slow. So even though this widget enables quick manipulation over distances, the overhead time required to set up the widget is rather large.

5 Task management

This section will discuss the topic of task management as it concerns or can be applied to large screen display devices. Since large displays have the capability to display much more information than a typical computer system, it is important for users to effectively complete tasks. Standard desktop environments have long estab-

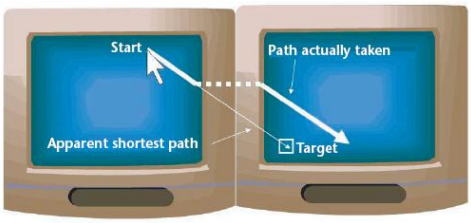

Figure 8: Mouse Ether. [Baudisch et al. 2004].

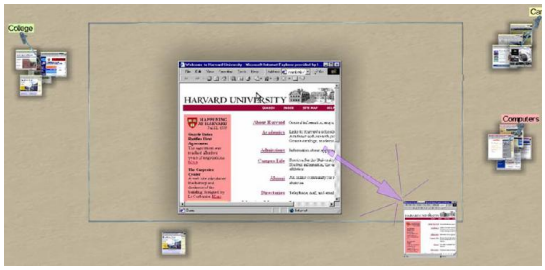

Figure 9: Scalable Fabric. [Robertson et al. 2004].

lished methods for typical tasks like moving, open, close, copy, and others. As discussed in the paper *The Large Display User Experience* [Robertson et al. 2005], porting typical desktop tasks to larger screens result in six general problems categories: loss of the cursor, bezel crossover, information distance, window management, task management, and configuration.

Given the Gigapixel's huge display size, information distance is a notable issue. Specifically, information distance refers to the amount of user effort required to select menus, move objects, etc. If a user waists to much time doing physical movement it will drastically affect task efficiency. Icons and menu bars are very useful on desktop systems because they have optimized for standard resolutions. But, if icons and menus are mapped to an extremely large resolution they can become less optimal. One way to overcome this is the *Drag-and-Pop* technique. This technique brings system functionality from the display's other sectors into a convenient working space around a selected object. In other words Drag-and-Pop is an attempt to bridge the gap between desktop software and large screen technology.

Bezel occlusion in large displays is probably the most notable setback of the hardware. Any large display that uses multiple monitors has this technological setback. Typical ways to counter bezel discontinuity include snapping and bumping. However, snapping and bumping may not work for a display like the Gigapixel given a single object may span many screens. Bezel issues result in two major options for graphical rendering. First, a system can display the direct mapping of the virtual work to the physical pixels. Of course, the result is a disorienting jump from screen to screen. The second option is to render the virtual work such that the bezels overlap the graphics. In this case, the obvious problem is the loss of visual data behind the bezels. A method called *Ether* [Baudisch et al. 2004] offers a compromise between the two options. This technique allows the hardware to display the entire graphical scene. However, the path of the cursor is mathematically altered to make the cursor appear to move from screen to screen without jumping, see figure 8. Fixing the cursor's behavior remains only part of the problem; behavior of the rest of the graphical workspace is still an open challenge to overcome.

Using multiple windows on large displays is another major challenge to deal with. For example, in a normal desktop display the maximize task is simple. However, for the Gigapixel if a window is maximized should the system increase the window size to cover the entire display or some subset of monitors? One method that helps this situation is called *WinCuts* [Tan et al. 2004]. This system allows users to define the display's regions as separate workspaces. WinCuts—of course—allows users to logically divide up related tasks or information, which can improve understanding and efficiency.

The *Scalable Fabric* [Robertson et al. 2004] technique is another method designed to deal with multiple windows. This technique allows a user to specify a "focus area" on the display by setting boundary markers, see figure 9. The marker may be moved to resize the focus area at anytime. While working inside the focus area, all windows and functionality behaves just as it would on a standard desktop. Furthermore, if windows are located or moved outside they are scaled down dynamically the closer they get to the edge. Periphery objects—if clicked—are automatically restored to their last known position in the focus area. Vice versa, if an object is minimized it goes to its last know peripheral position. Windows in the periphery can also be grouped together into clusters, allowing the user to logically group information. The main advantage of the Scalable Fabric is that it preserves the functionality that a user is used to with a desktop display while at the same time utilizing the rest of a large screen for secondary information.

Another technique called *Quickspaces* [Hutchings and Stasko 2002] focuses on window management but specifically deals with utilizing as much visual spaces as possible. The Quickspaces method is and expansion of the existing desktop metaphor. The expand feature is one task added to the desktop metaphor. Expansion allows a user to select a window causing it to automatically fill up the largest unoccupied area around that window. On a standard desktop metaphor, an expansion task would have to be manually done by the user by clicking and dragging edges, causing less speed and accuracy. Another task defined by Quickspaces is the shove technique. Shove is similar to expand given it is designed to increase the size of windows in the workspace. Shove expands a window and if it contacts another window it pushes it along until the user is satisfied or the side of the display is reached. The advantage of shove is that the size of one window may be greatly increased without subverting other windows. However, the task might move windows that the user does not want to be moved. Quickspaces also specifies related windows as "friend" or "non-friend". If a user performs a shove of expand, only friend windows will be kept non-subverted. The main advantage of Quickspaces is that it adds relatively simplistic task management techniques on top of well known desktop metaphors.

Up to this point, most task management techniques try to adapt desktop methods to work with large displays rather than designing new metaphors. However, until techniques are optimized for large screens due to years of use we must continue to design and try new approaches. After considering the techniques in this section we can extract some key design concerns. If designing new task management techniques for the Gigapixel the following questions should be asked: Is the workspace dividable, are multiple windows needed, can elements in the application be grouped, does the user have to use the entire screen all the time, and how much total information can displayed at any one time? Answering these questions define most of the needed functionality for any task management technique developed for the Gigapixel.

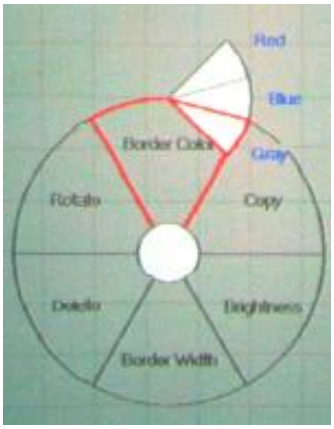

Figure 10: Widget used with VisionWand. [Cao and Balakrishnan 2003].

6 GUI widgets for large displays

Separate from direct object manipulation, widget interaction can enhance usability and provide system level control. Widgets include things like menus, virtual buttons, or anything that provides information or functionality. Furthermore, the primary advantage of using widgets is it allows a great deal of functionality into a concentrated area. Secondly, widgets can make task more intuitive by providing graphical feedback to a user. This section will discuss widgets as they apply specifically to large displays or large task management operations.

In the target acquisition section we discussed the *VisionWand* [Cao and Balakrishnan 2003] device. Most of the VisionWands primary functionality is gesture based. If too much functionality is controlled by gestures then the system becomes very complicated, which is why the designers use widgets for secondary functionality. The primary widget used by the VisionWand is a 2D pie menu, see figure 10. Secondary widgets including the tilt, dial panel, and compass are also used. In this case, the design of the widgets is directly related to the interaction hardware.

The *Vacuum* [Bezerianos and Balakrishnan 2005] technique utilizes another widget for its primary functionality. In this technique, users interact with distant objects by controlling a bulls eye object. The user controls two rays emitted from the circle that define the area of influence. The actual hardware for the Vacuum is done with a cursor, which illustrates that this widget was designed for standard desktop harware.

The previously discussed *Vision-Tracked Multi-Finger Gestural Input* paper [Malik et al. 2005] shows examples of how widgets can be used to provide visual feedback on a large screen display. In this system, the current gesture of each hand is displayed on the screen. If the current gesture activates a given functionality, a corresponding icon is displayed. For example, if the resize gesture is made, a standard resizing icon will appear between the users fingers, as shown on 11. This use of widget for visual feedback allows a user to be certain they have made correct hand gestures and will therefore increase system usability.

The *Frisbee* widget [Khan et al. 2004] shows an example that provides both functionality and visua feedback. In this case, the primary part of the widget looks similar to VisionWands widget and holds all the main functionality. This part of the widget resides in a local space area where the user does actual work. The second part of the widget is shown on remote space area and provides visual feedback of where the menu was in relation to the object before modification, see figure 7.

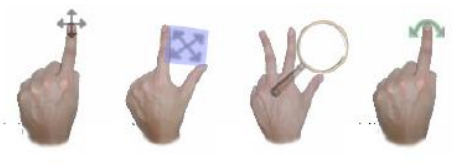

Figure 11: Widgets used for visual feedback. [Malik et al. 2005].

Widgets are important to large displays for several reasons. First, they can be designed specifically for any new interaction devices made for large screens. They can also be designed to accent new large screen interaction techniques with visual information. Most importantly, widget can alleviate some problems inherent to large displays. For example, widgets can be used to keep system control and other functionality nearby to where a user is working; this will help the problem of information distance and work better than standard desktop widgets like the Start menu that stay in one place. Lastly widget—if designed carefully—can greatly improve the usability and performance on large display applications, which will further advance the need for them and their use outside of research.

7 Touch screens

The ability of manipulating data by directly touching it, without the use of any other intermediary device, makes touch screens an appealing input device. This simplicity makes it especially good for novice users. The fast learning curve and lack of moveable parts (which could otherwise brake or be lost) makes touch screens an ideal interface to be place on public installations.

The drawbacks of this technology are high error rates, lack of precision, and arm fatigue [Potter et al. 1988]. Most devices allow for only one point to be tracked on the surface, which can limit interaction techniques. There are multi-touch sensitive devices, but sometimes the technology beneath them (such as vision-based sensing) them can make the input noisy [Benko et al. 2006]. Other sources of noise can come from device tracking errors, tremor in the user's hand, and noise due to the clicking motion [Benko et al. 2006], which cause cursor stabilization problems, and jeopardize precise selection. The problems with touch screen devices are more evident when interaction with this devices use GUI software developed for a normal mouse interface. It can be hard to select a small target considering the noisy input and the lower tracking resolution. Also, small targets can be easily occluded by fingertips, hands, and arms [Benko et al. 2006].

Some touch screen devices provide continuous pressure sensing information, meaning that they can report the degree of pressure applied to a point. Many other devices can only detect contact between the finger and the surface, thus are able to report only two states of interaction: touch or no touch. For interaction techniques, Buxton showed that is important to be able to differentiate at least three states: no touch, low pressure, and high pressure [Buxton et al. 1985]. Low pressure and high pressure states are used to differentiate between tracking and clicking. Different pressure states can be simulated by associating them with different finger areas [MacKenzie and Oniszczak 1998; Benko et al. 2006].

One solution presented for the occlusion problem is adding a cursor offset. An offset solves the occlusion problem, and also offers

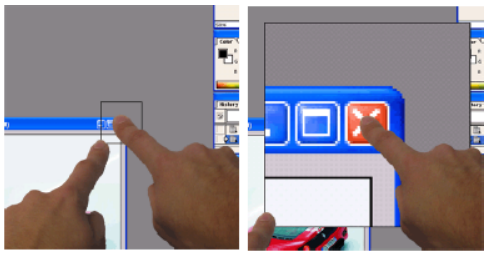

Figure 12: *Dual Finger Stretch* technique adaptively scales the user interface: a) The secondary finger specifies the square zooming area centered at the primary finger's location, b) Primary finger performs precise selection while, simultaneously, the secondary finger adjusts the level of magnification. [Benko et al. 2006].

a more comfortable way of pointing to hard to reach areas. An example is the *Take-Off* technique [Potter et al. 1988], which offers a fixed offset. The problem with a fixed offset is that the user has to be constantly compensating for the difference between the cursor and its finger, even when the target could be easily selected with direct pointing.

Benko et al presented a set of interaction techniques called *Dual Finger Selections* [Benko et al. 2006]. They address the precision problem by providing interaction using two fingers (called the *primary* and *secondary* fingers). The secondary finger acts as a helper to the primary finger. On their techniques they attend to the occlusion problem by providing a user-invoked and temporary offset. They developed two offset techniques: *Dual Finger Offset* and *Dual Finger Midpoint*. Dual Finger Offset always place the cursor above-and-to-the-right, or above-and-to-the-left of the primary finger, based on the secondary finger position. Dual Finger Midpoint places the cursor at the midpoint between the primary and the secondary fingers. More than just an offset, this technique allow for a more refined cursor speed control. Both offset techniques still have limitations. The fixed predetermined amount of offset used on Dual Finger Offset still makes it difficult to reach some areas of the screen, i.e., a lower corner. Dual Finger Midpoint also doesn't allow access to corners, since the cursor is always between the fingers. The *Dual Finger Stretch* technique (shown on figure 12 let users scale a portion of the screen with their secondary finger, while the primary finger does the selection. *Dual Finger X-Menu* provides a circular menu that is invoked each time the secondary finger touches the screen. Four of the six menu options are for speed control. They allow users to slow down the cursor speed and even freeze its position, which are great improvements toward cursor stabilization. One of the other options provides a magnification area in the middle of the menu, and the other removes the current offset (if any) and returns the cursor back to the current location of the primary finger. Another technique presented is *Dual Finger Slider*, which allows for easy cursor speed control. Figure 13 shows some Dual Finger Selection techniques in use.

When using touch screen on a wall-sized display, a difficulty is that not all surface of the display will be within arm's reach. On tiled displays that have bezels around each monitor, there is an additional problem: bezels can interfere with the flow of the interaction. For example, on a interaction where the user would have to slide its finger from one screen to another, the bezel would make the finger lose contact temporarily with the surface.

Simulating touch sensitive devices

Not all large displays are touch-sensitive devices. Through the use of computer vision tracking and image processing techniques, many systems have been proposed that could turn wall-sized display into a touch screen. Some examples include MetaDesk [Ullmer and Ishii

Figure 13: Precise dual finger selection techniques enable pixel-precise selections and manipulations on multi-touch screens. This image shows the use of *Dual Finger X-Menu* in selecting "slow 10X" mode. [Benko et al. 2006].

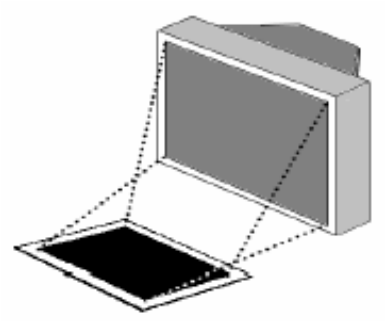

Figure 14: Touchpad to screen mapping. [Malik and Laszlo 2004].

1997], HoloWall [Matsushita and Rekimoto 1997], Designer's Outpost [Klemmer et al. 2001], and TouchLight [Wilson 2004].

On some of these systems, the user doesn't touch the display directly. Instead, the manipulation is done on another surface. The positive aspect is that even if the display have bezels, they will not interfere with the user's contact with the surface. A drawback is that even though the user is using its hands to reach and manipulate data, the feeling of direct manipulation is diminished.

Another advantage of these devices is that they are capable of more than tracking points on the surface. Most of them detect full *touch images*, enabling them to track body parts (hands, arms), allowing gestural input.

Many large displays applications are designed for multi-user interaction. While "real" touch screen cannot associate a point of contact with a specific user, some vision-based touch screen devices can more easily provide this feature [Malik et al. 2005].

Malik and Laszlo presented the *Visual Touchpad* [Malik and Laszlo 2004], which simulates a touch-sensitive surface. On this technique, the user doesn't interact directly with the screen, but rather with a touchpad. The system is composed by two cameras and a panel. The camera capture images of hand gestures over the panel, which can be recognized for interaction purposes. The setup is shown on figure 14. The feeling of direct manipulation is enhanced by rendering the user's hands onto the screen. They overcame occlusion problems by applying transparency to the hand's image. The touchpad area is directly mapped to the area of the screen. This technique was not specifically designed for wall-sized displays, and applying it to them create precision problems, since a small area on the touchpad would correspond to a large area on the display. One example of interaction is shown on figure 15.

Malik et al [Malik et al. 2005] also used a touchpad to interact with

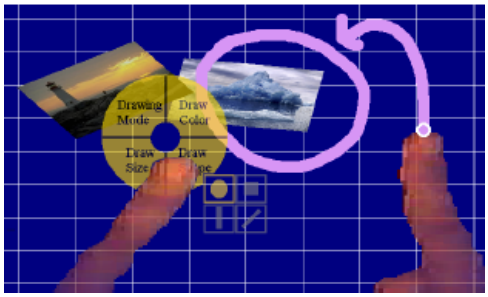

Figure 15: Example of interaction: pie menu for finger-painting. [Malik and Laszlo 2004].

a large display. One main difference from the Visual Touchpad is that the touchpad can be mapped to one small area of the display at a time, providing better precision during manipulation. Moving the mapped region allow the whole display to be within arms reach. They are able to track all ten fingers of a user's bare hands, allowing them to provide bimanual interaction techniques based on finger manipulation and gestures. Some of the techniques were asymmetric, which means that the non-dominant hand will do coarse and less frequent actions, while the dominant hand will do faster and more precise actions. One of the techniques they presented allows fast and precise access to any point of the display. To do so, they mapped the left half of the touchpad to the whole display, so the user can use its left hand to select where to place the workspace region on the screen. The right half of the touchpad is then mapped to the workspace region, and the user uses its right hand to more precisely perform tasks. This mapping is illustrated on figure 6. It's possible to support multiple users by having each user performing interactions on a uniquely identified touchpad.

TouchLight [Wilson 2004] is an example of a touch screen technology that could be adapted to transform available rear-projection displays, such as VisBox, into touch screen displays. TouchLight uses two cameras located behind the screen to compute a touch image, then tracks and identifies gestures using optical flow. The configuration is shown on figure 16, and an example of touch recognition is shown on figure 17.

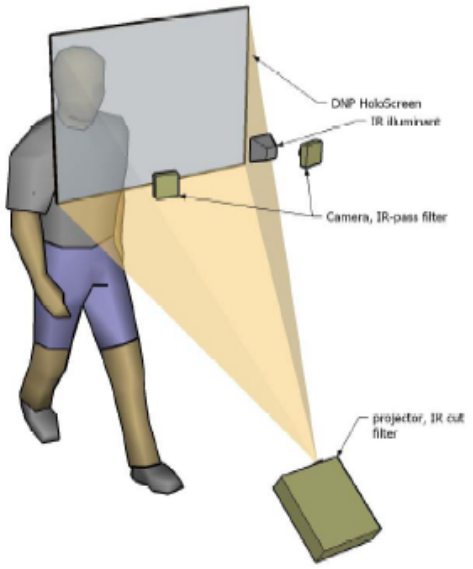

Figure 16: TouchLight physical configuration: screen with two cameras behind the screen. [Wilson 2004].

Figure 17: TouchLight *touch image* capture. (Top row) Images from both cameras. (Bottom) Edges of the image form *touch image*. [Wilson 2004].

8 Open challenges

The research area of interaction with large displays has identified many challenges, and researches have been working on solutions, as this survey has shown. But the work is not complete. Section 2 lists many challenges that are being studies. In addition, there are a few more aspects that need to be investigated.

Limitations of existing techniques. As illustrated on the section about target acquisition (see section 4), many techniques were proposed for simple target selection, but few of them address multiple target selection, and no good approach is offered for targets that are in different areas of the screen. This scenario is common in many applications for large display, and the lack of an effective interaction technique compromises the effective use of large displays. Another important issue insufficiently explored is the topic of manipulation tasks, or in other words, what to do after the targets are selected. Selecting and manipulating groups of objects have not been studied. On the subject of task management, discussed on section 5, much work still need to be done. Most of proposed techniques were based on tasks that are important on single monitor conditions. On large displays, new problems arise. It has been shown that users behave differently when using large or multiple displays. They open more application at a time, and leave many other documents open that support their main task [Ball and North 2005]. So, a technique that enable the user to switch focus between group working windows, for example, could improve usability on large display applications. This report mentions many interaction techniques used for large displays, but there is not a definitive solution for a technique that addresses all of these issues, or at least most of them.

Limited use of 3D interaction techniques. The use of 3D interaction techniques has not been fully explored. These techniques can be of great advantage, by giving users the usability present in direct manipulation even when they are distant from the display. Many techniques have been proposed (use of hand tracking, gestures, eye gaze manipulation, to name a few) but they are not mature. Much more needs to be done, from further exploring these techniques and to the study of new kinds of 3D interactions.

Software and interface design guidelines. Most applications used on large displays are designed based on traditional mouse interaction, even though many other interaction techniques have been investigated (such as gestures, speech recognition, pointing devices, eye tracking, etc). One example is the small target area to be clicked in order to close a window on a desktop environment. In order to

Figure 18: Air Traffic Controllers stand watch in the Air Traffic Control Center on the USS George Washington. [http://gw.ffc.navy.mil].

avoid usability issues, current software and interface design guidelines should be rethought in order to facilitate the use of such other techniques.

Understand the impact of each interaction technique. As this survey shows, there are a large number of interaction techniques being applied to large displays. There need to be more studies to understand why and how these techniques affect the user interaction with large display, and how they compare to each other. Identifying these points will help design better interaction techniques.

Domain-specific techniques. There are many different areas where the use of large displays can be appealing, or even crucial. Some examples are: visual surveillance, air traffic control, design, visual analytics, presentations, collaborative work. Applications can vary a lot from area to area, and it's important to understand the needs of each one of these areas in order to adapt or create new interaction techniques that address each specific need. As an example, let's focus on the area of visual analytics, which is the focus of our class project. On such application, at any given time the display can be showing massive amounts of data, an abundance of icons, or a highly detailed map, for example. It is not sure how techniques for enhancing cursor tracking would perform having such polluted background. Another example is to think on a traffic controler on a aircraft carrier, as illustrated in figure 18. If the application in use is heavily based on widgets, can the user access a menu fast enough when he is in an emergency situation and under extreme pressure? It's important to understand the needs of each area and create domain-specific interaction techniques.

Integrating techniques with working systems. Many large screen interaction techniques are proposed trying to solveonly one kind of interaction problem, and so are tested on simple and controlled environments. It's not sure how well they will perform well on "real world" applications, where highly complex tasks need to be performed. This needs to be further investigated. Another area that deserves attention is exploring how new interaction techniques can be applied to existing working systems. Many existing applications, some of which are being used for many years (like an air traffic control software), could not be redesigned from scratch in order to integrate appropriate large display interaction techniques. There needs to be an investigation of appropriate methods for such integration.

References

BALL, R., AND NORTH, C. 2005. An analysis of user behavior on high-resolution tiled displays. In *Human-Computer Interaction - INTERACT 2005*, Springer Berlin / Heidelberg, 350–363.

BAUDISCH, P., CUTRELL, E., CZERWINSKI, M., ROBBINS, D. C., TANDLER, P., BEDERSON, B. B., AND ZIERLINGER, A. 2003. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch- and pen-operated systems. In *INTERACT*.

BAUDISCH, P., CUTRELL, E., AND ROBERTSON, G. G. 2003. High-density cursor: a visualization technique that helps users keep track of fast-moving mouse cursors. In *INTERACT*.

BAUDISCH, P., CUTRELL, E., HINCKLEY, K., AND GRUEN, R. 2004. Mouse ether: accelerating the acquisition of targets across multi-monitor displays. In *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, ACM Press, New York, NY, USA, 1379–1382.

BENKO, H., WILSON, A. D., AND BAUDISCH, P. 2006. Precise selection techniques for multi-touch screens. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, ACM Press, New York, NY, USA, 1263–1272.

BEZERIANOS, A., AND BALAKRISHNAN, R. 2005. The vacuum: facilitating the manipulation of distant objects. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, New York, NY, USA, 361–370.

BUXTON, W., HILL, R., AND ROWLEY, P. 1985. Issues and techniques in touch-sensitive tablet input. In *SIGGRAPH '85: Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 215–224.

CAO, X., AND BALAKRISHNAN, R. 2003. Visionwand: interaction techniques for large displays using a passive wand tracked in 3d. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 173–182.

CZERWINSKI, M., SMITH, G., REGAN, T., MEYERS, B., ROBERTSON, G., AND STARKWEATHER, G. 2003. Toward characterizing the productivity benefits of very large displays. In *Human-Computer Interaction–INTERACT '03*, IOS Press, 9–16.

DACHILLE, F., AND KAUFMAN, A. 2000. High-degree temporal antialiasing. In *Proceedings of Computer Animation 2000*, 49–54.

FITTS, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement. 381–391.

GROSSMAN, T., AND BALAKRISHNAN, R. 2005. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, New York, NY, USA, 281–290.

GUIARD, Y., BLANCH, R., AND BEAUDOUIN-LAFON, M. 2004. Object pointing: a complement to bitmap pointing in guis. In *GI '04: Proceedings of the 2004 conference on Graphics interface*, Canadian Human-Computer Communications Society, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 9–16.

HUTCHINGS, D. R., AND STASKO, J. 2002. Quickspace: new operations for the desktop metaphor. In *CHI '02: CHI '02 extended abstracts on Human factors in computing systems*, ACM Press, New York, NY, USA, 802–803.

KHAN, A., FITZMAURICE, G., ALMEIDA, D., BURTNYK, N., AND KURTENBACH, G. 2004. A remote control interface for

large displays. In *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 127–136.

KHAN, A., MATEJKA, J., FITZMAURICE, G., AND KURTENBACH, G. 2005. Spotlight: directing users' attention on large displays. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, New York, NY, USA, 791–798.

KLEMMER, S. R., NEWMAN, M. W., FARRELL, R., BILEZIKJIAN, M., AND LANDAY, J. A. 2001. The designers' outpost: a tangible interface for collaborative web site. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 1–10.

LI, K., CHEN, H., CHEN, Y., CLARK, D. W., COOK, P., DAMIANAKIS, S., ESSL, G., FINKELSTEIN, A., FUNKHOUSER, T., HOUSEL, T., KLEIN, A., LIU, Z., PRAUN, E., SAMANTA, R., SHEDD, B., SINGH, J. P., TZANETAKIS, G., AND ZHENG, J. 2000. Building and using a scalable display wall system. *IEEE Comput. Graph. Appl. 20*, 4, 29–37.

MACKENZIE, I. S., AND ONISZCZAK, A. 1998. A comparison of three selection techniques for touchpads. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 336–343.

MACKENZIE, I. S. 1992. Fitts' law as a research and design tool in human-computer interaction. 91–139.

MALIK, S., AND LASZLO, J. 2004. Visual touchpad: a two-handed gestural input device. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, ACM Press, New York, NY, USA, 289–296.

MALIK, S., RANJAN, A., AND BALAKRISHNAN, R. 2005. Interacting with large displays from a distance with vision-tracked multi-finger gestural input. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 43–52.

MATSUSHITA, N., AND REKIMOTO, J. 1997. Holowall: designing a finger, hand, body, and object sensitive wall. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 209–210.

MCGUFFIN, M. J., AND BALAKRISHNAN, R. 2005. Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM Trans. Comput.-Hum. Interact. 12*, 4, 388–422.

NI, T., SCHMIDT, G. S., STAADT, O. G., LIVINGSTON, M. A., BALL, R., AND MAY, R. 2006. A survey of large high-resolution display technologies, techniques, and applications. *vr 0*, 223–236.

POTTER, R. L., WELDON, L. J., AND SHNEIDERMAN, B. 1988. Improving the accuracy of touch screens: an experimental evaluation of three strategies. In *CHI '88: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, New York, NY, USA, 27–32.

ROBERTSON, G., HORVITZ, E., CZERWINSKI, M., BAUDISCH, P., HUTCHINGS, D. R., MEYERS, B., ROBBINS, D., AND SMITH, G. 2004. Scalable fabric: flexible task management. In *AVI '04: Proceedings of the working conference on Advanced visual interfaces*, ACM Press, New York, NY, USA, 85–89.

ROBERTSON, G., CZERWINSKI, M., BAUDISCH, P., MEYERS, B., ROBBINS, D., SMITH, G., AND TAN, D. 2005. The large-display user experience. *IEEE Comput. Graph. Appl. 25*, 4, 44–51.

TAN, D. S., GERGLE, D., SCUPELLI, P. G., AND PAUSCH, R. 2004. Physically large displays improve path integration in 3d virtual navigation tasks. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, New York, NY, USA, 439–446.

ULLMER, B., AND ISHII, H. 1997. The metadesk: models and prototypes for tangible user interfaces. In *UIST '97: Proceedings of the 10th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 223–232.

VOGEL, D., AND BALAKRISHNAN, R. 2005. Distant freehand pointing and clicking on very large, high resolution displays. In *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, ACM Press, New York, NY, USA, 33–42.

WILSON, A. D. 2004. Touchlight: an imaging touch screen and display for gesture-based interaction. In *ICMI '04: Proceedings of the 6th international conference on Multimodal interfaces*, ACM Press, New York, NY, USA, 69–76.

ZHAI, S., BUXTON, W., AND MILGRAM, P. 1994. The silk cursor: investigating transparency for 3d target acquisition. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM Press, New York, NY, USA, 459–464.