MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Zoom-and-Pick: Facilitating Visual Zooming and Precision Pointing with Interactive Handheld Projectors

Clifton Forlines, Ravin Balakrishnan, Paul Beardsley, Jeroen van Baar, Ramesh Raskar

TR2005-095 October 2005

Abstract

Designing interfaces for interactive handheld projectors is an exiting new area of research that is currently limited by two problems: hand jitter resulting in poor input control, and possible reduction of image resolution due to the needs of image stabilization and warping algorithms. We present the design and evaluation of a new interaction technique, called zoom-and-pick, that addresses both problems by allowing the user to fluidly zoom in on areas of interest and make accurate target selections. Subtle design features of zoom-and-pick enable pixel-accurate pointing, which is not possible in most freehand interaction techniques. Our evaluation results indicate that zoom-and-pick is significantly more accurate than the standard pointing technique described in our previous work.

ACM Symposium on User Interface Software and Technology (UIST)

Zoom-and-Pick: Facilitating Visual Zooming and Precision Pointing with Interactive Handheld Projectors

Clifton Forlines[1], *Ravin Balakrishnan*[2,1], *Paul Beardsley*[1], *Jeroen van Baar*[1], *Ramesh Raskar*[1]

[1]Mitsubishi Electric Research Laboratories
Cambridge, MA
{forlines, pab, jeroen, raskar}@merl.com
www.merl.com

[2]Department of Computer Science
University of Toronto
ravin @dgp.toronto.edu
www.dgp.toronto.edu

ABSTRACT

Designing interfaces for interactive handheld projectors is an exiting new area of research that is currently limited by two problems: hand jitter resulting in poor input control, and possible reduction of image resolution due to the needs of image stabilization and warping algorithms. We present the design and evaluation of a new interaction technique, called *zoom-and-pick*, that addresses both problems by allowing the user to fluidly zoom in on areas of interest and make accurate target selections. Subtle design features of *zoom-and-pick* enable pixel-accurate pointing, which is not possible in most freehand interaction techniques. Our evaluation results indicate that *zoom-and-pick* is significantly more accurate than the standard pointing technique described in our previous work.

Categories and Subject Descriptors: H.5.2 [**User Interfaces**]: Interaction styles, Graphical User Interfaces.

Additional Keywords: Interactive handheld projectors, jittery input, zooming, selection techniques.

INTRODUCTION

Projectors have traditionally been used as static, output-only devices for presenting content in non-interactive manner to a relatively passive audience. However, recent advances in projection technology have led to significant decreases in cost, size (Figure 1, *left*), and power requirements – a trend that shows no sign of slowing down. Researchers may now consider more interactive, mobile, uses for projectors. Consider, for example, the possibilities that arise when projectors become small and light enough to carry in our pockets [23] and operate with a few fingers. Any surface can be become a computationally enabled display, and, unlike current small computing devices like cellphones and PDAs, the projected image will be large enough to enable easy collaborative viewing by more than one person. Raskar et al. [20, 21] explored usage scenarios for handheld projectors and propose a variety of interesting ways in which they can be used in an interactive manner. Of particular interest is their solution to the problems of keystone distortion, rotation, and hand jitter. By using a camera in conjunction with the handheld projector, they compute the pose of the projector relative to the display surface and use that pose information to correct for distortion and rotation and to factor out projector motion. This enables a dynamically updated virtual image (e.g., a standard desktop) to be stably displayed at a fixed location and orientation within the projector's distorted and jittery image plane on the display surface. By displaying a pointer at the center of the projector's image plane, movement of the projector can directly control the pointer's movement across the stabilized desktop image inscribed within. With direct pointer control and a button on the projector, all standard mouse interactions in a WIMP interface are possible, resulting in an *interactive handheld projector* that is simultaneously an input and an output device. Furthermore, since the pointer maps directly to hand movement, the interaction is akin to direct pointing, which is arguably more satisfying than other indirect approaches like using a touchpad or isometric joystick on the projector.

Although appealing in its simplicity and directness, this style of interactive handheld projection has limitations in practice due to the lack of precision of pointer movement as well as low resolution of the stable desktop image within the jittery overall projection. In this paper, we discuss these limitations, develop a new technique called *zoom-and-pick* that alleviates them, present an experiment that evaluates the performance of the new technique, and conclude by exploring design variations motivated in part by our experimental results.

INTERACTIVE HANDHELD PROJECTOR BACKGROUND

In this section, we present a brief review of the method for interacting with a stabilized projected image. Figure 2 illustrates the concept, and the interested reader can find the details of the algorithms used in [4, 21]. First, a camera is used to detect visual markers on the wall and the position of these markers allows us to compute the 3D position of the camera relative to the wall. The black squares in Figure

Figure 1. *Left*, Mitsubishi Pocket Projector. *Right*, prototype handheld projector [4].

UIST'05, October 23–27, 2005, Seattle, Washington, USA.

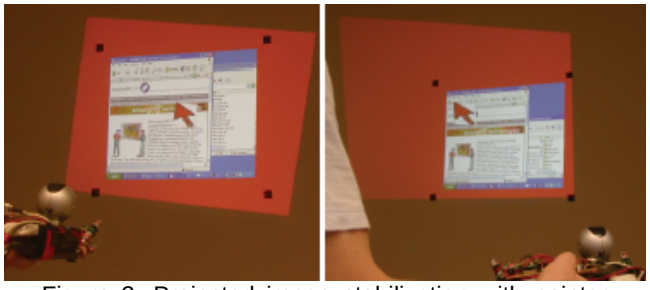

Figure 2. Projected image stabilization with pointer control. A virtual "desktop" image is projected at a fixed location and orientation within the projector's image plane on the display surface. A graphical pointer is displayed at the center of the uncorrected projected image. *Right*, as the projector moves, the pointer moves in sync, but the inscribed desktop image remains stable, thus enabling WIMP interaction. With the exception of the four black markers in the corners of the workspace, all of the graphics displayed on the wall are created by the projector. Images and concept based on [4, 21].

2 and the red LEDs in Figure 5 are examples of the types of markers our prototypes use. Because the camera is rigidly attached to the projector, we can easily find the projector's 3D position, which allows us to correct for rotation, movement, and distortion in order to create a projection that is static on the wall. Figure 2 shows a moving projector. Even while the projector moves, the desktop image remains stable relative to the wall. The distorted polygon surrounding the desktop image shows what an uncorrected projector would display. Point-based interaction is achieved by placing the mouse pointer in the *uncorrected center* of the projector image. The end effect is a static desktop projection with a cursor moving across it in response to the user's pointing motion.

INTERACTIVE HANDHELD PROJECTOR LIMITATIONS

Jittery Pointer

Although the projector pose information enables the generation of a stable desktop image within the jittery overall projected image, the pointer which is mapped directly to the center of the uncorrected projected image remains susceptible to hand jitter. This makes precise pointing difficult. This problem is inherent in all direct freehand pointing techniques, such as laser pointer input [16-18]. As we discuss next in the related work section, attempts have been made to reduce this problem via a variety of techniques, but with only limited success.

Despite the jittery pointer problem, we were encouraged by our observations during demonstrations of the interactive handheld projector prototype at the ACM SIGGRAPH 2004 Emerging Technologies forum. Over 2000 attendees tried the prototype and all were able to easily select 40-pixel targets in a test application, despite only using the device for a few minutes each. That such a large and diverse set of people could immediately use the prototype with no training is extremely promising; however, smaller 12-pixel targets were much harder to select due to hand jitter, thus motivating the need for improved pointing precision.

Pixel Wastage

The technique for rendering an undistorted, stable desktop image at a fixed location and orientation within the jittery overall projection requires that some percentage of the projected image pixels be wasted (Figure 2). If we constrain hand and thus projector image plane motion to a small amount, then few pixels are sacrificed, resulting in a relatively high resolution desktop image. However, if we allow for significant hand and projector image plane motion, then a large amount of pixels must be sacrificed, resulting in a lower resolution desktop image. For example, in order for the pointer displayed at the center of the moving projector image plane to be able to traverse the full extents of the inscribed desktop image in the most direct one-to-one mapping, the desktop image cannot exceed a quarter of the pixels of the projected image. If we wish to use a greater proportion of the projected image, a gain factor must be applied to the pointer movement, further compounding the jittery pointer problem. As such, the user or system designer has to make a tradeoff between image and input resolution. As new technology drives the increase of projector resolution, such pixel wastage will likely not be a major limitation; however, it will still be desirable to provide facile techniques to enable users to improve visual acuity without reducing input acuity.

RELATED WORK

Various researchers have explored the use of laser pointers as input devices for large screen interaction [10, 13, 14, 16-19]. Although appealing in its simplicity and low cost, laser-pointer interaction is hindered by the same hand-jitter problem faced by interactive handheld projection. Myers et al. [16] compared laser pointers to other devices in pointing tasks and found laser pointers to perform the worst, with at best 4-pixel selection accuracy even after predictive filtering. Oh and Stuerzlinger [17] designed a computer-controlled laser pointer with Kalman filtering, but their experiments showed error rates of around 40% when selecting relatively large 40-pixel diameter targets. Matveyev and colleagues [13, 14] describe a more elaborate model for reducing the effects of jitter but do not provide user data measuring the performance of their approach. Peck [19] studied laser-pointer tracking deviations and found significant hand-jitter effects. Olsen and Nielsen [18] cleverly design laser-pointer interaction techniques optimized to avoid hand jitter issues as much as possible. In summary, our review of the literature indicates that the effects of hand-jitter is not easily solved by data-filtering techniques, and a more fruitful approach is to design interaction techniques that deal with hand jitter explicitly in their design. Furthermore, most data-filtering techniques, even if they do work, tend to introduce some lag, which has been shown to be detrimental to performance [24].

Standard pointing-enhancement techniques such as using non-linear transfer functions that change pointer-movement characteristics based on the velocity of the user input (e.g., the pointer enhancement in WindowsXP), work well for relative input devices like mice, touchpads, and isometric joysticks. However, these techniques are not easily applicable to absolute pointing devices like interactive handheld projectors or laser pointers where there is a direct one-to-one correspondence between device movement and pointer control. Using non-linear transfer functions in these scenarios will introduce a dynamically changing offset between device and pointer positions, resulting in a loss of the very directness that makes such pointing techniques so attractive in the first place. Furthermore, if the offset gets too large, the projector or laser pointer may end up outside the tracking envelope. Similarly, pointer prediction techniques [3, 15] have been successfully used to aid stabilization in laboratory settings with relative devices, but are difficult to apply to direct absolute pointing due to the resulting disconnect that arises between actual (i.e., projected image center or laser point) and predicted pointer position.

Our work also draws upon the research in distortion techniques for zooming into areas of interest on a virtual image. Beginning with the early work by Furnass [8] and Sarkar and Brown [22], a variety of focus-in-context techniques have emerged including the perspective wall [12] and the DragMag lens [25]. Carpendale [5, 6] provides a nice unifying treatment of the various techniques.

Most recently, Fitzmaurice et al. [7] describe an interaction technique called tracking menus that has the nice property of allowing a pointer to move freely within a predefined region, but drag the region along in the direction of pointer movement when the pointer moves beyond the edge of the region. This enables a single pointer to perform both local manipulation within a stable region, and also reposition the region within the broader workspace as needed. We leverage this idea to enable pointer stabilization.

ZOOM-and-PICK

In an effort to alleviate both jitter and resolution limitations of interactive handheld projectors, we have developed a new interactive widget, called zoom-and-pick, which allows users to dynamically zoom into areas of interest on the desktop image for higher resolution viewing and has unique design elements to facilitate high-precision pointing operations with a jittery pointer. While Zoom-and-Pick was developed with handheld projectors in mind, we believe that it may also be appropriate for laser-pointer interaction.

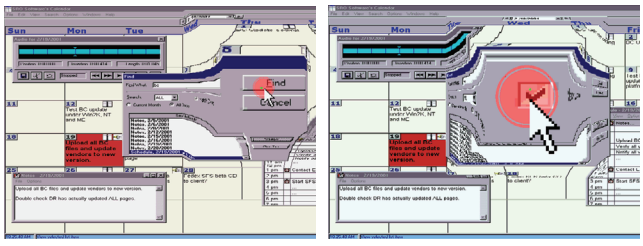

Figure 3. A square fisheye lens surrounds the pointer. A user can adjust the level of lens magnification as they move the pointer around the workspace.

The zoom-and-pick widget design begins with a square fisheye lens [5, 8], with adjustable zoom level, centered about the pointer, as shown in Figure 3. The widget follows the pointer as it's moved around the desktop image. This allows for higher resolution viewing of the contents within the lens. Magnification of the visual space, however, does not alter the device to pointer mapping in motor space, thus pointing precision remains unchanged. Furthermore, a jittery pointer will cause the widget to jitter accordingly, thus making viewing of the zoomed-in area somewhat unpleasant.

Our solution to this precision and jitter problem is inspired by the tracking menu concept of Fitzmaurice et al. [7]. We define a circular "dead" zone within the bounds of the fisheye lens of the zoom-and-pick widget. As long as the pointer remains within this dead zone, the widget is stable and immobile, thus allowing the zoomed-in area within the lens to be viewed comfortably without the ill effects of jitter (Figure 4, *left*). When the pointer moves beyond this zone, it drags the widget with it. At all times, the center of widget represents the hotspot for selection, thus pointer movement within the dead zone does not affect selection accuracy. In our current design, we typically do not display the true pointer when the zoom-and-pick widget is active, allowing users to focus their attention on the hotspot at the

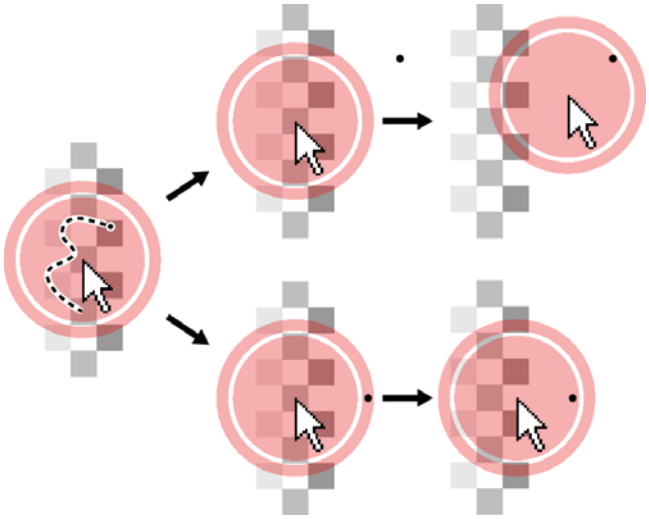

Figure 4. The black dot represents the true pointer while the white mouse-like pointer in the center of the circular widget represents the hotspot for interaction. *On the left*, as long as the true pointer's movement remains within the center dead zone (dotted path), the widget remains stable. When the true pointer moves outside of the bounds of the widget (top path), the widget repositions itself so that the true pointer is within. If the true pointer moves out of the dead zone and into the rim (bottom path), then the widget moves by one pixel and the true pointer is moved back inside the widget.

center of the widget. Up to this point in the design, we have achieved visual magnification and widget stabilization when the pointer is in the dead zone. However, pointer jitter continues to be reflected in the movement of the widget as it is dragged around, and high precision pointing is still not possible.

To facilitate high precision pixel accurate pointing, we further refine the widget's design by defining a small rim around the dead zone. If the true pointer moves beyond the dead zone and into the rim, the widget is moved one pixel towards the true pointer (Figure 4, *bottom path*). If the true pointer moves beyond the dead zone and the rim, the widget is dragged around at a coarse granularity with the same jittery pointer as before (Figure 4, *top path*).

It is important to note a few subtleties in our design. First, the size of the rim is necessarily greater than one pixel, and indeed we make it proportional to the zoom level. To prevent more than one pixel widget movement at a time even if the pointer itself has moved more than one pixel when transitioning from the dead zone into the rim, we warp the pointer back to the dead zone just before the rim boundary. As such, each crossing of the pointer from the dead zone into the rim results in a single pixel movement. However, a small offset is now created between the virtual pointer and its true position. Thus, we have somewhat relaxed the one-to-one device-to-pointer mapping. If a user makes many successive single-pixel movements in the same direction, this offset will get too large and the technique will eventually breakdown. In practice, however, we find that users only make a few single-pixel adjustments in the same direction, followed by a coarser grained adjustment whereupon the pointer regains its true absolute position. A second subtlety is that this technique only works because the pointer data is sampled in a discrete manner. As Figure 4 illustrates, the pointer can only get beyond the rim if it moves fast enough such that it gets from within the dead zone to outside the rim over two successive sampling frames.

In summary, what we end up with is a widget that supports zooming in visual space, stable visualization and selection when the pointer is within the dead zone, high-precision single-pixel widget positioning by moving the pointer slowly from the dead zone into the rim, and quick lower-precision repositioning by moving the pointer quickly beyond the rim.

To achieve a modeless and transient interaction style, we use the projector's orientation about the axis perpendicular to the image place for invocation and zoom control. This orientation is easily computed from the projector pose information. When the projector is rotated past 5 degrees away from the up-vector, in either direction about the axis perpendicular to the image plane, the zoom-and-pick widget fades in. Continuing to rotate from 5 to 45 degrees increases the zoom level linearly. Rotating in the opposite direction towards the upright orientation decreases the zoom level, and the widget fades away when the 5 degree threshold is passed. We note that the zoom-and-pick widget's invocation and zoom adjustment can be achieved in a variety of other ways, including using transducers like buttons or scroll wheels. The use of projector orientation for this purpose, however, results in a nice facile interaction where the user simply moves the projector around for regular coarse grained pointing, rotates and moves a little to zoom-and-pick, and rotates back to return to regular use.

EXPERIMENT

Goals

To validate the efficacy of zoom-and-pick, we conducted an experiment that compares zoom-and-pick with regular pointing using an interactive handheld projector. We expect that zoom-and-pick will enable more precise pointing and that it will also enable selection of targets that are too small to be selected by regular pointing, but it is possible that users may not be able to control the various parameters of the zoom-and-pick widget sufficiently well to achieve this objective. We also wish to investigate if the additional complexity introduced by the widget results in longer pointing times.

Participants

Twelve participants, 2 women and 10 men ranging in age from 18 to 35 years, were recruited mainly from the local university communities and volunteered for the experiment. Participants were paid $20 each and all were regular computer users. Three of the participants were members of our lab; however, like the rest of the participants, none had previous experience with interactive handheld projectors.

Apparatus

The experiment was conducted on a Pentium4 3.2Ghz workstation running WindowsXP. The prototype interactive handheld projector (Figure 5, *left*) was constructed within a box of dimensions 8" x 11" x 2.5" and consisted of a Plus projector model V-1080 with 1024x768 resolution, a rigidly attached Basler 1394 camera, and 4 rigidly attached laser pointers. The prototype weighed 6.5 pounds. Users held and manipulated the projector with a pistol grip handle attached to the bottom of the box. A single trigger button on the handle provided click input. The camera is used to compute pose (position and orientation) of the device. The four laser pens, which were visible red lasers but could be invisible infrared, are used to project distinctive points on the projection surface to ensure reliable high-quality pose computation. Details of the pose computations can be found in [21]. Participants stood during the experiment, 6 feet away from the projection surface, resulting in a projected image of 40" x 30" within which a desktop image of 20" x 15" was rendered (Figure 5, *center*). Rotation of the projector about the axis perpendicular to its image plane controlled the activation and zoom level of the zoom-and-pick widget. A maximum rotation in either direction of 45 degrees was possible, with a maximum zoom level of 25x magnification. Zoom level was linearly proportional to the rotation angle.

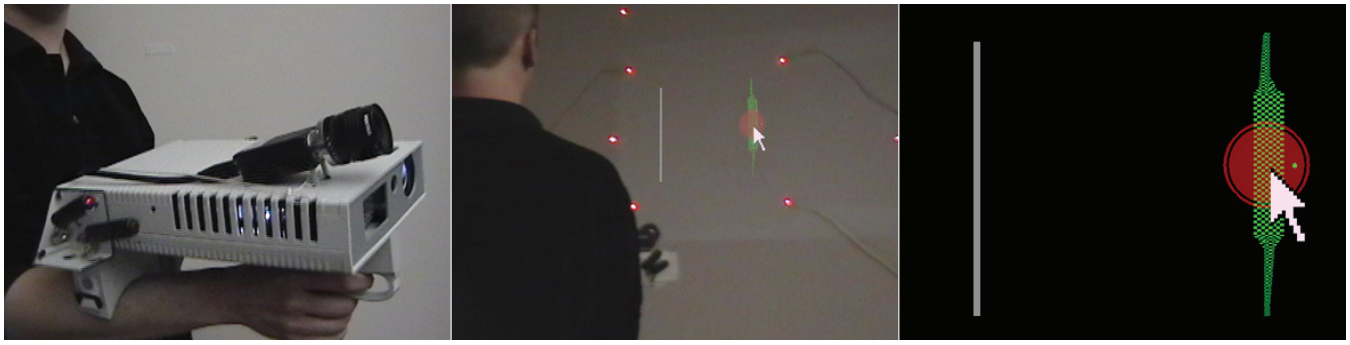

Figure 5. *Left*, Handheld projector. *Center*, User performing the experimental task. Four LEDs on the wall define the work area. *Right*, Close-up of the experimental task.

Task and Stimuli

The task was reciprocal 1D target acquisition, which required participants to point at two targets back and forth in succession. This is a standard Fitts' law [11] task for evaluating pointing devices. The targets were rendered as rectangles with a simple grid texture, equidistant from the center of the display along the left-right axis (Figure 5, *right*). Textured targets were used to enable differentiation from the surroundings when visually zoomed in. The target to be selected was green, and the other light grey. When participants clicked on the green target, the targets would swap colors, as an indication that the participant had to now move to and select the other target. The zoom-and-pick widget was rendered in semi-transparent red.

Design

A repeated-measures within-participant factorial design was used. The independent variables were *technique*: regular pointing and zoom-and-pick, distance between targets *D* (400, 800 pixels), and target width *W* (4, 8, 16, 32 pixels). We chose to use more widths than distances because the primary difference between the techniques is their precision, which would not be affected by distance as much as width. We also chose a fairly wide range of *W*'s because we wanted to find the crossover point where it becomes necessary to use zoom-and-pick. Also, our smallest *W* of 4 pixels was deliberately chosen to be below the hand jitter threshold as measured by Myers et al. [16].

Participants were randomly assigned to two groups of six participants each. The first group performed the experiment with the regular pointing technique first, followed by the zoom-and-pick technique. The second group did it in reverse order.

For each technique, participants performed 3 blocks of trials. Within each block, each of the eight *D*-*W* combinations was presented once in random order to the participant. For each *D*-*W* combination, participants performed 8 reciprocal selections, the first of which was thrown out because of the uneven starting point of the pointer at the start of each set.

Participants could take voluntary breaks between each set of reciprocal trials, and breaks were enforced between conditions. In addition, at the start of each technique, a set of 10 warm-up trials were given to familiarize the participants with the experiment and the relevant technique. Each participant performed the entire experiment in one session, including breaks, in approximately 1 hour. In summary, the design was as follows (excluding warm-ups):

12 participants x
2 techniques (*regular* and *zoom-and-pick*) x
3 blocks per technique x
2x4 = 8 D-W combinations per block x
7 selections per D-W combination
= 4032 selections in total.

In most Fitts'-law-type experiments, a successful target selection occurs when the participant clicks on the target. The pointer's location when the button is released is typically ignored. Real interface tasks, however, often require that the click and release events both occur while the pointer is on the target. With interactive handheld projection there is a concern that there could be excessive amount of pointer movement between the click and release events. As such, it is important to measure user ability to both click and release within the target, and our participants were instructed accordingly. Participants had to successfully click and release within the target before the colors would swap, even if it required multiple clicks. This effectively removes the possibility that participants may try to "race" through the experiment by clicking anywhere. However, after pilot studies indicated that participants had some difficulty selecting the smallest width targets with the regular technique, we altered this design to allow participants to continue to the next trial after three unsuccessful attempts. Our observations of participants throughout the experiment indicated that none tried to subvert the experiment by clicking three times in rapid succession simply to move to the next trial.

Results

Selection-Only Analysis

We first considered trials as successful if participants clicked within the target on the first try. Release events were ignored, thus this analysis resembles the standard Fitts' law target selection paradigm. Selection time was

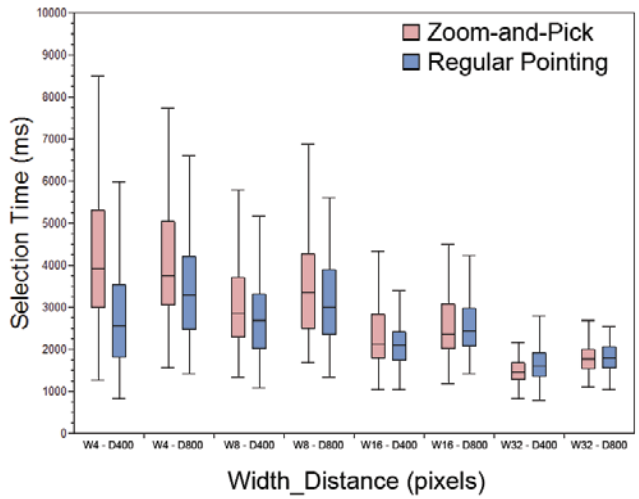

Figure 6. Selection time for both techniques, broken down by each *D-W* combination.

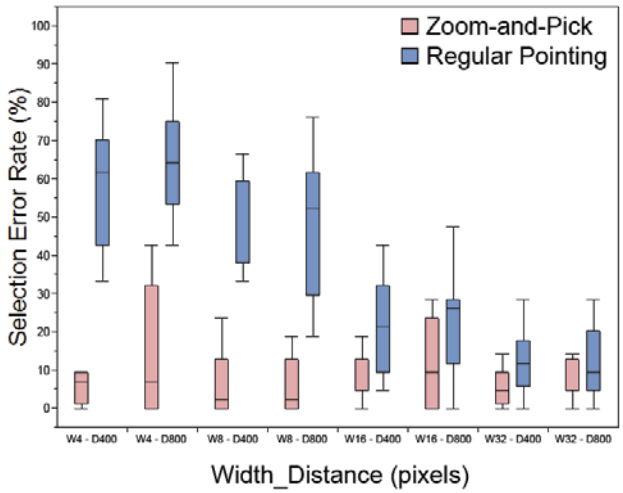

Figure 7. Selection error rate for both techniques, broken down by each *D-W* condition.

calculated as the time to move from the source target to the first click on the destination target. Selection time data reported below do not include trials marked as errors.

Repeated measures analysis of variance showed that order of presentation of the two techniques had no significant effect on selection time ($F_{1,10}$ = 3.30, p = .10) or selection error rate ($F_{1,10}$ = 0.67, p = 0.43). This indicates that any skill transfer was symmetric, and thus a within-participants analysis is appropriate.

There was a significant main effect for technique on selection time ($F_{1,10}$ = 32.00, p < .001), with a mean of 2.51 seconds for regular pointing and 3.01 seconds for zoom-and-pick. This indicates that the added complexity of zoom-and-pick is detrimental to performance.

As might be expected from Fitts' law, there was a significant main effect for target distance ($F_{1,10}$ = 23.51, p = .001) and target width ($F_{3,10}$ = 68.54, p< .001) on selection time. There was no significant interaction between technique and target distance on selection time ($F_{1,10}$ = 0.47, p = .51). However, there was a significant interaction between technique and target width on selection time ($F_{3,10}$ = 20.54, p < .001), with zoom-and-pick performing increasingly better relative to regular pointing as target width increased. No other significant interactions were observed relative to selection time. Figure 6 illustrates these effects.

There was a significant main effect for technique on selection error rate ($F_{1,10}$ = 77.51, p < .001), with a mean of 35.6% for regular pointing and 9.2% for zoom-and-pick. This clearly indicates that participants had significant trouble with using regular pointing for accurate selection. Perhaps unsurprisingly, there was no significant effect for target distance on selection error rate ($F_{1,10}$ = 2.17, p = .17). As might be expected, target width had a significant effect on selection error rate ($F_{3,10}$ = 56.73, p < .001). As seen in Figure 7, selection error rate for zoom-and-pick is fairly consistent across all widths, whereas with regular pointing selection error rate significantly increases as width decreases. A significant interaction between technique and target width on selection error rate ($F_{3,10}$ = 87.79, p < .001) confirms the differences between the two techniques in this regard. No other significant interactions were observed relative to selection error rate.

Some previous Fitts law studies have shown that trials immediately following an error trial are often slower than normal trials. This effect could have unfairly inflated the selection times for the regular pointing trials, which had a very high number of errors. While this may be the case, this inflation would not change the overall selection time ranking of the two techniques – our results already show that regular pointing is superior to Zoom-and-Pick in terms of selection time. For a full comparison between the two techniques, we would ask the reader to look at not only selection time, but also error rates.

In addition to selection time and error rate, we also looked at the amount of zoom that participants used in the zoom-and-pick technique. As Figure 8 shows, participants zoomed more when targets were small, and hardly zoomed at all for the largest target. Figure 9 illustrates the visual target width (i.e., target width multiplied by the zoom level) at the time of selection. If users were optimal in their actions, we would expect that they would zoom just enough to enable selection, such that the resulting visual target width was identical regardless of the original target width. However, as Figure 9 indicates, participants zoomed more than absolutely necessary for the larger targets, opting for truly easy selection rather than completely optimal zoom levels. This also indicates that participants were taking advantage of the fact that it did not take much more effort to zoom a little more with our technique.

There was neither a significant effect for block on selection time ($F_{2,10}$ = 2.34, p = .17) nor on selection error rate ($F_{2,10}$ = 0.41, p = .17). The lack of significant learning effects is probably due to the balancing of learning and fatigue.

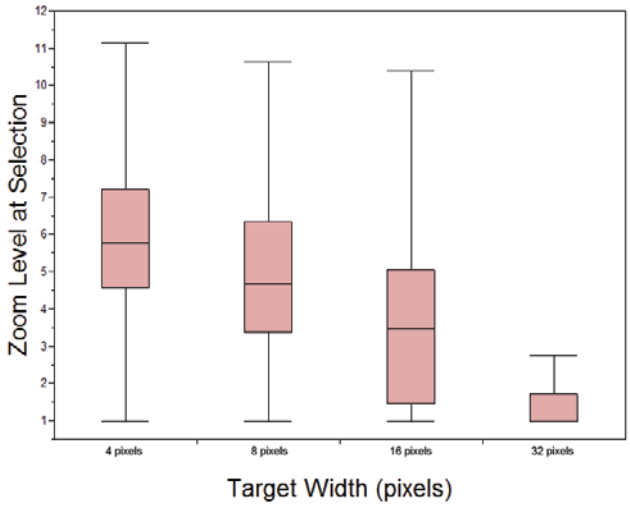

Figure 8. Zoom level at time of selection for the zoom-and-pick technique, broken down by each width.

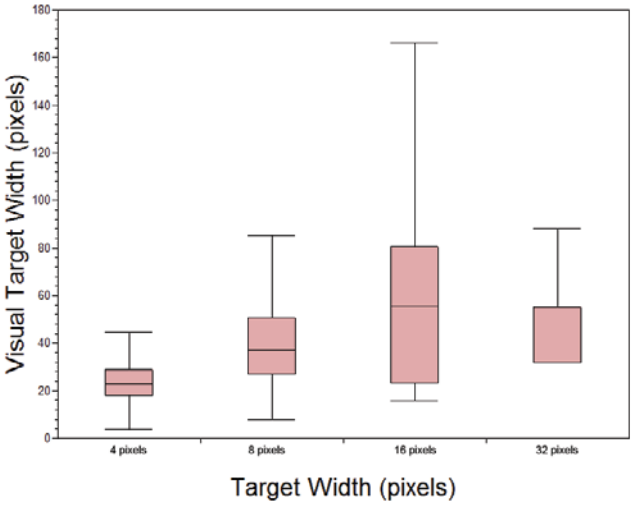

Figure 9. Visual target width (width * zoom level) at time of selection for the zoom-and-pick technique, broken down by each width.

Select-and-Release Analysis

We next looked at the data by considering trials as successful if participants both clicked *and* released within the target on the first try. Release events were thus factored in, thus this analysis resembles the situation in real interfaces in which a button is selected only if both click and release events occur while the pointer is within the target. It also measures participant's ability to release a target within a given spatial threshold, which is important in drag-and-drop or positioning tasks. Select-and-release time was calculated as the time to move from the source target to when the release event was recorded on the destination target. Select-and-release time data reported below do not include trials marked as errors. In comparison to the selection only analysis reported in the previous section, this analysis is more conservative in that more trials will be marked as erroneous.

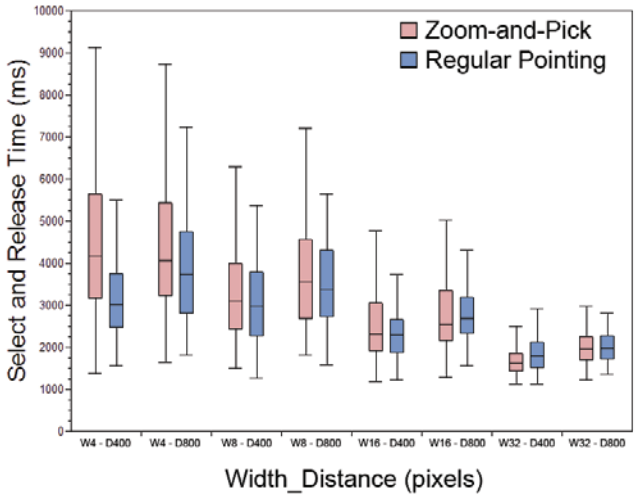

Figure 10. Select-and-release time for both techniques, broken down by each *D-W* combination.

As with the selection only analysis, repeated measures analysis of variance showed that order of presentation of the two techniques had no significant effect on select-and-release time ($F_{1,10}$= 0.59, p = .46) or select-and-release error rate ($F_{1,10}$ = 1.07, p = 0.32). Again, this indicates that any skill transfer was symmetric, and thus a within-participants analysis is appropriate.

There was a significant main effect for technique on select-and-release time ($F_{1,10}$ = 9.66, p < .01), with a mean of 2.96 seconds for regular pointing and 3.33 seconds for zoom-and-pick. Comparing these times to the selection only times of 2.51 and 3.01 seconds for regular pointing and zoom and pick respectively, we see that including the release event does not uniformly increase the time for both techniques.

As with selection only time, there was a significant main effect for target distance ($F_{1,10}$ = 13.83, p = .004) and target width ($F_{3,10}$ = 65.85, p< .001) on select-and-release time. There was no significant interaction between technique and target distance on select-and-release time ($F_{1,10}$ = 0.02, p = .90). As before, there was a significant interaction between technique and target width on select-and-release time ($F_{3,10}$ = 10.8, p < .01), with zoom-and-pick performing increasingly better relative to regular pointing as target width increased. No other significant interactions were observed relative to select-and-release time. Figure 10 illustrates these effects.

As with selection only error rate, there was a significant main effect for technique on select-and-release error rate ($F_{1,10}$ = 126.46, p < .001), with a mean of 53.2% for regular pointing and 15.3% for zoom-and-pick. Comparing these to the means of 35.6% and 9.2% for selection only error rate for regular pointing and zoom-and-pick respectively, we see that incorporating the release event results in a further 17.6% of trials being marked as errors for regular pointing. In contrast, incorporating the release event only results in another 6.1% of trials being marked as errors for zoom-and-pick. This further illustrates the advantages of zoom-and-pick over regular pointing.

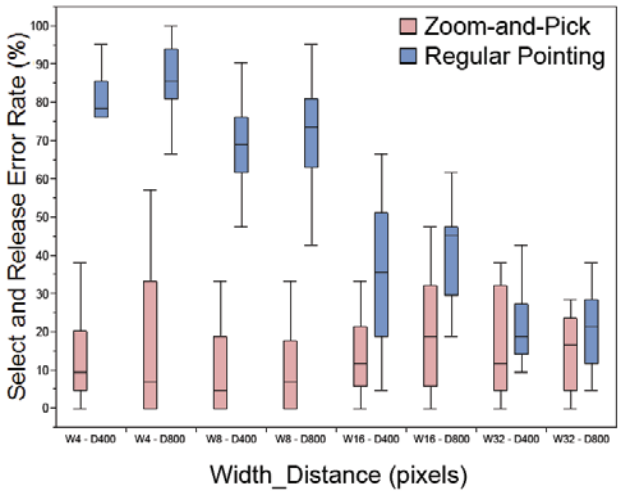

Figure 11. Select-and-release error rate for both techniques, broken down by each *D-W* condition.

Similar trends to selection error rate were observed for the remaining tests. There was no significant effect for target distance on select-and-release error rate ($F_{1,10}$ = 4.64, p = .06). Target width had a significant effect on select-and-release error rate ($F_{3,10}$ = 72.57, p < .001). There was also a significant interaction between technique and target width on select-and-release error rate ($F_{3,10}$ = 110.95, p < .001). No other significant interactions were observed relative to select-and-release error rate. Figure 11 illustrates these effects.

There was a significant effect for block on select-and-release time ($F_{2,10}$ = 5.17, p = .04) with a mean of 3.2 seconds, 3.1 seconds, and 3.0 seconds for blocks one, two, and three respectively. There was no significant effect on select-and-release error rate ($F_{2,10}$ = 1.31, p = .28). The lack of significant learning effects is again probably due to the balancing of learning effects with fatigue.

Discussion

Taking the results as a whole, we see that while zoom-and-pick pays a small penalty relative to regular pointing in terms of selection and select-and-release times, it has a huge advantage in terms of accuracy. In particular, while the accuracy of regular pointing degrades considerably – to the point of being almost unusable – as targets get smaller, zoom-and-pick performs consistently across a variety of target widths. In fact, with a few hours of practicing with the technique, one author of this paper can quickly select individual pixels with almost 100% accuracy. This clearly demonstrates the efficacy of zoom-and-pick's design. Also interesting is the fact that when the release event is taken into account, the advantages of zoom-and-pick become even more apparent.

It is worth noting that even with zoom-and-pick, the selection error rate of 9.2% and select-and-release error rate of 15.3% are both higher than the roughly 4% error rate typically seen in Fitts' style selection tasks. We believe that this high error rate is mostly due to fatigue effects resulting from the weight of the 6.5 pound projector coupled with having to use an unsupported arm. In a sense, by using a fairly crude prototype handheld projector for our experiment, we are effectively looking at a worse-case scenario. A lighter handheld projector would almost certainly result in lower error rates. Even after factoring our the fatigue effects, the extremely high error rates for regular pointing essentially make it untenable as an pointing technique except for very large targets. The results of this formal experiment thus reinforce our informal observation of 2000 people informally using the prototype at the SIGGRAPH 2004 conference as discussed at the start of this paper.

It is worth noting again that our experimental design differs somewhat from the classical Fitts' paradigm where participants could proceed to the next trial regardless of whether or not they selected a target successfully. From past experience in conducting many similar studies, we have observed that participants in these types of experiments often get bored and start racing through the experiment by clicking arbitrarily. By forcing subjects to make a successful click to complete a trial (or at least to try several times as in our relaxed design), it is in the subject's best interest to perform honestly and minimize errors. In other words, they have to self-optimize between going too fast and making too many errors from which they have to recover versus being too accurate and thus taking overly long to complete the experiment. This design is also more ecologically valid as it is a closer approximation to a realistic GUI pointing task, where users typically do not abort a selection simply because they missed it the first time. Our analysis, however, only considers trials that were selected correctly on the first try, and as such from an analysis standpoint we do not differ from the classical design.

DESIGN VARIATIONS

Based on our experiences in using zoom-and-pick and our experimental results, we can consider several design variations to the zoom-and-pick widget

Our original implementation, as evaluated experimentally, mapped the zoom level linearly to projector rotation, with a maximum of 25x magnification. While this setup was suitable for our evaluation, we can imagine situations where users may want even greater zoom levels, or more precise control over it.

One alternative design might be to divide the zoom control into two zones. As illustrated in Figure 12 (*right*), the first zone would allow for linear position control of zoom level. Moving into the second zone would transition smoothly into velocity control of the zoom level, thus allowing for infinite zoom levels.

Another refinement or alternative would be to allow ratchet zooming where the multiple cranks back and forth would successively increase the zoom level. If ratchet zooming were implemented, however, moving backwards would simply reset the ratchet rather than un-zoom the lens. Un-

zooming in this design could be supported by assigning clockwise movements from the up-vector to zooming and counterclockwise movements to un-zooming. In this situation, one could also imagine a "slam to reset" feature where a quick movement in the opposite direction that crosses the up-vector resets the zoom level to zero.

We can also refine the behavior of the widget's rim region. Currently the entire rim behaves in a similar fashion, an alternative design would be to divide the rim into areas, as shown in Figure 12 (*left*). When the true pointer moved into the thick rim areas, it would move the widget by a single pixel. When it moved into the thin areas, it would drag the widget until the user reversed the direction of the pointer.

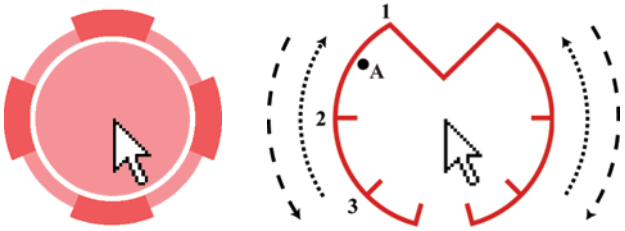

Figure 12. Design variations. (*left*), the widget's rim is broken into multiple sections, each of which cause different behavior when the true mouse pointer is moved into or through it. (*right*), rotation is mapped to linear position control of zoom level when in the 1-2 zone, and transitions to velocity control of the zoom level when in zone 2-3. If ratchet zooming is implemented, moving in the direction of the thick dotted lines would zoom, but moving backwards in the direction of the fine dotted lines would ratchet the widget, allowing for potentially infinite zooming.

Zoom-and-pick was motivated by the need to solve the jitter and visual resolution problem of interactive handheld projectors. However, the widget could also be useful with other jittery input transducers like laser pointers. For the rotation sensing laser tweezers described in [13], our technique should transfer directly to this jittery input device. Since most laser pointers do not provide rotation information, we cannot replicate the implicit mapping of projector rotation to the zoom-and-pick widget control that we used. Instead, we could activate the widget with an explicit trigger like a button press or dwell, and manipulate the zoom level with widget handles that the user would cross over. This crossing style interface [1, 2] requires less precise movements than traditional target clicking, and would thus be well suited to noisy laser-pointer input. A nice property of zoom-and-pick is that its mechanism for precision and coarse-grain widget movement does not need to be altered for laser-pointer use.

CONCLUSIONS AND FUTURE WORK

We have explored issues surrounding the use of interactive handheld projectors – a new class of combined input/output technology that could significantly alter the way we interact with information in mobile environments. After identifying jittery input and low visual resolution as two key problems of current interactive handheld projectors, we developed and evaluated zoom-and-pick, a new interaction technique designed to solve both problems. It is important to note that while the low visual resolution problem will likely resolve itself as projection technology continues to improve, the jittery input problem is a limitation due to human physiology and as such cannot be solved except by appropriate interaction design. Our evaluation of zoom-and-pick showed that it significantly improves the usability of interactive handheld projectors, drastically enhancing user ability to select small targets that are very difficult if not impossible to select with the direct pointer technique developed in [4, 21].

A particularly nice feature of zoom-and-pick is that it allows for pixel accurate positioning despite jittery input. In addition to it's utility in the interactive handheld projector domain, this feature could also be used for other direct hand manipulation techniques such as when using laser pointers as computer input devices.

For our observations of participants using zoom-and-pick, it is apparent that it provides a very facile interaction for moving items around the screen. The basic interaction of zoom, pick, unzoom, drag, zoom, drop, and unzoom is easily understood and performed. Furthermore, as users gain expertise, these steps can be chunked into one fluid continuous gesture, thus leveraging the modeless nature of the technique.

We note that the zoom lens feature is also useful for highlighting regions of interest when discussing something with two or more people in a collaborative setting. This is similar to the "spotlight" technique described by Khan et al. [9].

We also note that the prototype handheld projector used in our experiments was rather heavy and cumbersome, as compared to the most recent models, an example of which is seen in Figure 1 but is unfortunately not yet easily available for evaluation at the time of this research and submission of this paper. Despite this limitation, however, the advantages of zoom-and-pick were strongly demonstrated. The poor performance of traditional mouse interaction lends weight to the argument that traditional GUI widgets need to be reconsidered as they move from the desktop to handheld pointing and interaction devices. We intend to examine and develop appropriate alternatives to these familiar GUI elements as we continue to explore interacting with handheld projectors. It will also be interesting to develop other interaction techniques and usage scenarios that leverage the unique characteristics of this exiting new input/output technology.

ACKNOWLEDGEMENTS

We thank our experiment participants and our colleagues at MERL for valuable discussions and their help in editing this paper.

REFERENCES

1. Accot, J. and Zhai, S. (2002). More than dotting the i's - foundations for crossing-based interfaces. *ACM CHI Conference on Human Factors in Computing Systems*. p. 73-80.
2. Apitz, G. and Guimbretière, F. (2004). CrossY: a crossing-based drawing application. *ACM UIST Symposium on User Interface Software and Technology*. p. 3-12.
3. Baldwin, J., Basu, A., and Zhang, H. (1998). Predictive windows for delay compensation in telepresence applications. *IEEE International Conference on Robotics & Automation*. p. 2884--2889.
4. Beardsley, P., van Baar, J., Raskar, R., and Forlines, C. (2005). Interaction using a handheld projector. *IEEE Computer Graphics and Applications*, 25(1). p. 39-43.
5. Carpendale, M.S.T. and Montagnese, C.A. (2001). A framework for unifying presentation space. *ACM UIST Symposium on User Interface Software and Technology*. p. 61-70.
6. Carpendale, S. (1999). A framework for elastic presentation space. *Department of Computer Science*, Simon Fraser University.
7. Fitzmaurice, G., Khan, A., Pieke, R., Buxton, B., and Kurtenbach, G. (2003). Tracking menus. *ACM UIST Symposium on User Interface Software and Technology*. p. 71-79.
8. Furnas, G. (1986). Generalized fisheye views. *ACM CHI Conference on Human Factors in Computing Systems*. p. 16-23.
9. Khan, A., Matejka, J., Fitzmaurice, G., and Kurtenbach, G. (2005 - in press). Spotlight: Directing users' visual attention on large displays. *ACM CHI Conference on Human Factors in Computing Systems*.
10. Kirstein, C. and Muller, H. (1998). Interaction with a projection screen using a camera tracked laser pointer. *Multimedia Modeling Conference*. p. 191-192.
11. MacKenzie, S. (1992). Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7. p. 91-139.
12. Mackinlay, J., Robertson, G., and Card, S. (1991). The perspective wall: Detail and context smoothly integrated. *ACM CHI Conference on Human Factors in Computing Systems*. p. 56-63.
13. Matveyev, S. and Göbel, M. (2003). The Optical Tweezers: multiple-point interaction technique. *Virtual Reality Software and Technology*. p. 184-188.
14. Matveyev, S., Göbel, M., and Frolov, P. (2003). Laser pointer interaction with hand tremor elimination. *HCI International*. p. 376-740.
15. Murata, A. (1998). Improvement of pointing time by predicting targets with a PC mouse. *International Journal of Human Computer Interaction*, 10(1). p. 23-32.
16. Myers, B., Bhatnagar, R., Nichols, J., Peck, C.H., Kong, D., Miller, R., and Long, C. (2002). Interacting at a distance: measuring the performance of laser pointers and other devices. *ACM CHI Conference on Human Factors in Computing Systems*. p. 33-40.
17. Oh, J.-Y. and Stuerzlinger, W. (2002). Laser pointers as collaborative pointing devices. *Graphics Interface*. p. 141-149.
18. Olsen, D.R. and Nielsen, T. (2001). Laser pointer interaction. *ACM CHI Conference on Human Factors in Computing Systems*. p. 17-22.
19. Peck, C. (2001). Useful parameters for the design of laser pointer interaction techniques. *Extended Abstracts of the ACM CHI Conference on Human Factors in Computing Systems*. p. 461-462.
20. Raskar, R., van Baar, J., Beardsley, P., Willwacher, T., Rao, S., and Forlines, C. (2003). iLamps: geometrically aware and self-configuring projectors. *ACM Transactions on Graphics*, 22(3).
21. Raskar, R., Beardsley, P., van Baar, J., Wang, Y., Dietz, P., Lee, J., Leigh, D., and Willwacher, T. (2004). RFIG lamps: interacting with a self-describing world via photosensing wireless tags and projectors. *ACM Transactions on Graphics*, 23(3). p. 406-415.
22. Sarkar, M. and Brown, M. (1992). Graphical fisheye views of graphs. *ACM CHI Conference on Human Factors in Computing Systems*. p. 83-91.
23. Symbol_Technologies. Laser projection display. www.symbol.com/products/oem/lpd.html.
24. Ware, C. and Balakrishnan, R. (1994). Reaching for objects in VR displays: Lag and frame rate. *ACM Transactions on Computer-Human Interaction*, 1(4). p. 331-356.
25. Ware, C. and Lewis, M. (1995). The DragMag image magnifier. Extended Abstracts of the ACM CHI Conference on Human Factors in Computing Systems. p. 407-408.