High-density cursor: a visualization technique that helps users keep track of fast-moving mouse cursors

Patrick Baudisch, Edward Cutrell, & George Robertson

Microsoft Research, Redmond, WA 98052, USA
{baudisch, cutrell, ggr}@microsoft.com

Abstract: As bigger screens and multi-monitor configurations become more popular, users employ higher mouse accelerations in order to traverse the screen reasonably quickly. The faster the mouse cursor moves, however, the more it seems to jump from one position to the next, as it is updated only at the refresh rate of the monitor. This lack of visual continuity increases the risk of users losing track of the cursor. The *high-density cursor* presented in this paper addresses this issue by filling in additional cursor images between actual cursor positions (temporal supersampling). Unlike existing techniques, such as the Windows *mouse trail*, the proposed technique preserves the responsiveness of the mouse cursor. We report the results of a user study in which the high-density cursor improved participants' performance on a Fitts' law task by up to 7% for target acquisitions across long distances. Moreover, the conditions of the high-density mouse that were tested were subtle enough that they were often not even detected, and were never considered distracting.

1 Introduction

As more users switch to multi-monitor display configurations, as well as with the advent of large screens for individuals (e.g. Czerwinski et al., 2001, Baudisch et al., 2002), users have to move across larger distances on their screens. Users of CAD, image editing, or development environment, for example, may place their tool palettes on a separate screen (Grudin, 2001), which requires them to traverse substantial distances in order to access these palettes.

The increased distances often affect how users handle their mice. To avoid excessive clutching, users who want to get across the screen reasonably fast have to set their mice to higher speeds or acceleration values, which have drawbacks. One problem is that higher mouse speeds make it harder for users to visually track the cursor and to visually reacquire the cursor as it is approaching a target, such as a button the user wants to click. If users lose track of the cursor, they have to spend extra time re-acquiring it.

But it is not only the cursor's speed that makes the cursor hard to track. Because of the way today's operating systems render the cursor, it appears to jump from one position to the next, as illustrated by Figure 1a. This artifact, also referred to as *temporal*
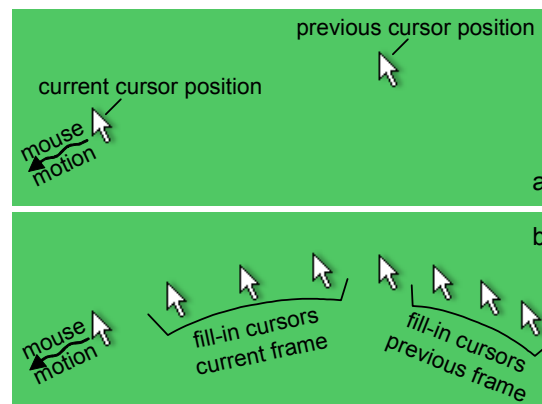
Figure 1: (a) The problem: at high mouse speeds, the mouse cursor seems to jump from one position to the next. (b) *High-density cursor* makes the mouse cursor appear more continuous by inserting additional cursor images between actual cursor positions.

aliasing (Dachille and Kaufman, 2000), becomes a problem as higher mouse speeds make the cursor jump further. On a triple LCD setup, for example, we observed cursor jumps of over 500 pixels—about half the width of a regular computer screen. These large gaps between the individual appearances of the cursor make it hard for users to visually track the cursor, to predict its trajectory, or to detect it as it is

approaching a target. This limits the speed at which users can reliably operate the mouse.

Unlike other rendering-related problems, this one will not go away with increasing processing power. The reason is that the distance the cursor appears to jump is caused by the low refresh rate at which it is displayed. The position of the cursor on the screen can be *visibly* updated only once per screen update. The refresh rate of computer displays, however, has not significantly increased since monitors reached "flicker-free" values around 80Hz. In fact, with the emergence of LCD screens, the frame rate has dropped to values around 60Hz.

The result is that as mouse cursors traveling at increasing speed are rendered on monitors of decreasing refresh rates, the apparent gaps in the path of the mouse cursor get larger and larger.
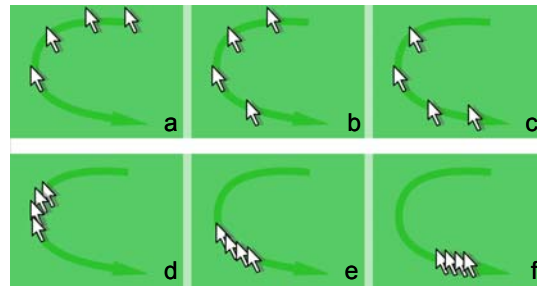
2 High-density cursor

The *high-density cursor* (*hd cursor*) addresses this issue by creating a denser cursor track, as shown in Figure 1b. This is accomplished by filling in additional cursor images into the space between the current cursor position and the previous cursor position. Figure 1b shows the same mouse motion as Figure 1a; both images were created by overlaying two successive images on the computer screen, so-called frames. The left-most cursor image in each frame in Figure 1b is rendered at the newest position read from the mouse device. The other three cursor images of each frame are filled in to make the mouse trail appear denser.

High-density cursor addresses two main objectives. First, it increases the visual continuity between cursor images in order to help users track and extrapolate the path of the cursor. Second, the denser track created by hd cursor makes it easier to visually acquire than a regular cursor. The larger number of cursor images on the screen gives hd cursor more saliency or "visual weight" which makes it easier for users to detect as it approaches a target. For example, there are four times more cursor images in the example shown in Figure 1.

High-density cursor achieves these objectives without sacrificing the responsiveness of the mouse. Techniques that enhance the visibility of the cursor by displaying individual cursor images for a longer period of time (e.g. the Windows mouse trail) result in the appearance of a trail *following* the cursor. Since trailing cursor images continue to move after the user stops moving the mouse, this approach leads to the perception that the mouse does not immediately respond to the user's action. Hd cursor is

free from such artifacts because it obtains its enhanced visual weight from using a new set of cursor images for every frame, rather than keeping the same set of cursor images alive longer. The hd cursor approach removes all cursor images in the following frame, so that trailing effects cannot occur and the perceived responsiveness of the mouse is preserved[1]. Figure 2 shows the two techniques in direct comparison.

Figure 2: (a-c) Three successive screenshots from a mouse movement enhanced with the Windows mouse trail. Trailing cursor images lag behind. (d-f) 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.

In the remainder of this paper, we will briefly discuss the related work, discuss design decisions and options, and present the results of a controlled experiment in which hd cursor significantly increased the target acquisition speed of the participants. We conclude with a brief discussion of the findings and an outlook to future work.

3 Related work

The general problem of long-distance target acquisition has been the subject of a whole series of interaction techniques. Since cursor motion generally follows eye motion (e.g. Card et al., 2001), several techniques have been proposed that use eye gaze to warp the cursor to the target either directly selecting the target (Sibert and Jacob, 2000) or combined with manual cursor motion (Magic pointing, Zhai et al., 1999). Other researchers proposed snapping the mouse to target locations. Dulberg et al. (1999) proposed a flying click or flick for that purpose (see also Moyle and Cockburn, 2002), Swaminathan and Sato (1997) proposed making relevant controls on

---

[1] Responsiveness is hard to communicate using noninteractive media such as images, video, or text. An interactive Flash demo of the hd cursor can be tried out at http://research.microsoft.com/users/baudisch

the screen "sticky". Geißler (1998) presented a technique that allows "throwing" objects to distant screen locations. In order to simplify targeting, some researchers explored techniques expanding targets (McGuffin and Balakrishnan, 2002).

Techniques enhancing the detectability of the mouse cursor have been offered by several products. Kensington Mouseworks (2001) offers an option that enlarges the mouse cursor when the mouse moves rapidly, then slowly shrinks it back to its regular size. The previously mentioned Windows mouse trail was designed to enhance the visibility of the mouse cursor on LCDs with a slow response by leaving cursor images on the screen for two or more frames. Ben Bederson's LiveCursor points in the direction of its motion, thereby providing an additional visual cue for the cursor's motion direction (http://www.cs.umd.edu/~bederson/livecursor).

Motion blur, the general concept hd cursor is derived from, is well known in both photography and computer graphics (Cook, 1984). In photography, motion blur occurs because the shutters of real-world cameras remain open for a non-zero amount of time. This makes the camera capture a finite segment of time on each picture, which makes moving objects appear blurry. In computer graphics, motion blur generally has to be generated explicitly, e.g. by rendering a scene multiple times with little time increments in between and aggregating the results (Dachille and Kaufman, 2000). Here, motion blur is used to reduce temporal aliasing, such as to prevent spokes of a wheel from appearing to rotate backwards. Creating motion blur is computationally expensive though, which has mostly limited the availability of this technique to off-line rendering. In the space of user interfaces, motion blur has been used to improve the perceived responsiveness of graphics applications (Conner and Holden, 1997) and for helping users anticipate motion by using rendering styles inspired by cartoons (Chang, 1993; Thomas & Calder, 2001).

4 Designing high-density cursor

The high-density cursor version described in Section 2 is one specific instantiation in a larger design space. In this section, we state the goals around which we designed hd cursor, describe the design aspects we explored, and the decisions we made.

4.1 Design goals
During preliminary studies, we observed two strategies that participants used for acquiring distant targets. One subset of users started by identifying the target's location, then they looked at the cursor and moved the cursor to the target while visually tracking it. The other subset used the opposite approach, i.e. they started by identifying the cursor location, then looked at the target, and stayed focused on the target while moving the cursor towards it. Hd cursor is shaped according to two simple design objectives derived from these user strategies.

1. Enhance the predictability of the cursor path. Especially when tracking the cursor, better predictability will help users detect motion errors earlier and make better guesses where the cursor went when lost. In this section, we will discuss three properties of hd cursor designed to enhance predictability: enhanced trail density/continuous blur, smooth interpolation of the cursor path, and the preservation of trail density as a cue for cursor speed.

2. Increase the detectability of the cursor. Hd cursor increases the visual weight of the mouse cursor to make it easier to detect. While higher visual weight should also make the cursor easier to track, it will especially help re-acquiring the cursor, as it makes the approaching cursor easier to detect. In the following, we will discuss three properties of hd cursor that are designed to increase its visual weight, i.e. enhanced trail density, enhanced cursor opacity, and cursor scaling.

While striving to achieve these two main objectives, hd cursor was designed to keep side effects to a minimum. Most important, we limited ourselves to changes that do not affect the responsiveness of the mouse cursor. Besides that, we kept a close eye on the tradeoff between visual weight and potential side effects thereof, such as distraction or the occlusion of screen content.

4.2 Design decisions
When discussing design parameters in the following, the cursor traversal shown in Figure 3a will serve us as a reference example.

4.2.1 Motion blur vs. temporal supersampling
In order to reduce the jumps between cursor positions, the hd cursor version described in Section 2 uses temporal supersampling. In an analog world, however, the cursor would not jump at all; it would move continuously. Figure 3b shows a version achieving such an effect by applying continuous motion blur. As this figure illustrates, a naïve implementation of motion blur makes the cursor invisible at higher mouse speeds, as each cursor image gets stretched out across hundreds of pixels, reducing its opacity below what users could detect. Before this technique can be considered, we have to increase the opacity of the blurred cursor to a level

detectable to the user (Figure 3c uses opacity = $1/v^k$, with v denoting the cursor's speed and $0<k<1$, here $k = \frac{1}{2}$).[2]

We implemented prototypes of both temporal supersampling and continuous motion blur. While continuous motion blur leads to unsurpassed visual continuity, a high-density cursor visualization based on temporal supersampling has 4 advantages. First, the latest cursor position is always shown blur-free and in full opacity. Second, the distance between cursor images will typically be a stronger visual cue for mouse speed than trail opacity (see below). Third, a supersampled cursor integrates itself well with the appearance of the normal cursor that users are familiar with today; in our user study, many participants did not even notice the change. Fourth, it is computationally less expensive. Because of these reasons, we focused our research on temporal supersampling rather than continuous motion blur.
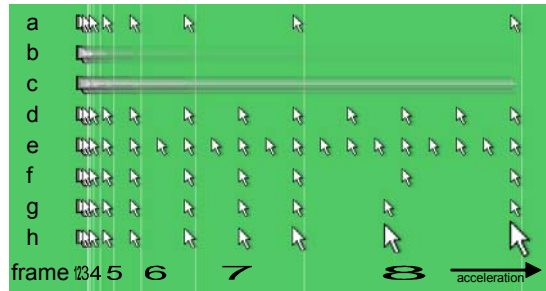
Figure 3: Alternative high-density cursor designs. The user moves the cursor from left to right, accelerating from 0 to 256 pixels per frame. We overlay screen images from 8 successive frames. (a) original cursor, (b) motion blur, (c) blur with enhanced opacity, (d) super sampled cursor, (e) same with half onset threshold, (f) non-linear transition function, (g) interpolated trail density, (h) cursor scaling.

4.2.2   Detectability vs. intrusiveness
Hd cursor's primary methods for increasing the detectability of the cursor is the enhancement of the cursor's trail density as already described. Other enhancements may be applied in addition to that. One method that we have implemented is cursor scaling, which makes cursor images grow as a function of cursor speed (Figure 3h).

---

[2] Note that we have implicitly applied a similar opacity enhancement to the temporal supersampling approach described in Section 2. Temporal supersampling by itself, i.e. rendering n translated cursor images and overlaying the results, results in each image having only $1/n$ opacity. Obtaining opaque cursor images, as hd cursor uses them, requires bringing opacity back to 100%, which corresponds to enhancing opacity by a factor on n.

While higher visual weight will generally make the cursor easier to detect and thus help users visually reacquire the cursor, additional visual weight may also interfere with the user's primary tasks, as it may attract more of the user's attention and occlude more screen content. In the design of hd cursor we addressed this issue by increasing the visual weight of the cursor only when the mouse speed exceeds a certain threshold. In the example shown in Figure 3d, the cursor remains unchanged for the first six frames. Only as mouse speed exceeds the onset threshold, cursor images are filled in (one cursor image in frame 7, three in frame8).

The optional cursor scaling is governed similarly, i.e. a motionless or slow moving cursor is always its normal size, which also prevents potential usability issues with target acquisition. As the cursor slowly passes the last few pixels towards the target, it has already shrunk back to its original size so that users carry out the final part of their target acquisition with a cursor of familiar size (unlike Kensington, 2001). Since the choice of visual weight involves tradeoffs that depend on the user's tasks and preferences, we made all three parameters (density, opacity, and scaling) user-configurable. Figure 3d, for example, shows how a lower onset threshold leads to a denser trail.

4.2.3   Cursor trail density indicates mouse speed
A regular mouse cursor offers an additional visual cue for mouse speed, namely the distance between individual cursor images. Larger distances indicate that the mouse is moving fast; shorter distances indicate a lower mouse speed (see, for example, Figure 3a). Knowledge about the speed is potentially useful, as it may help users detect over- or undershooting early on. It therefore seems desirable to preserve this property.

Filling in additional cursor images can affect this cue. The trail shown in Figure 3e, for example, evinces that there is some acceleration at the beginning, but when hd cursor kicks in (frame 5), all further acceleration becomes unperceivable. To preserve at least some distance-speed proportionality, hd cursor makes distances between cursor images grow with speed, as illustrated by Figure 3f. Our current hd cursor implementation computes the number of cursor images c for a given frame using the following *transfer function* $c(v) = (v/m)^k$ with v denoting the cursor's speed, m the onset threshold, and $k < 1$ being the transfer function constant. If ($v < m$), hd cursor is inactive and $c(v)$ is set to 1. Figure 4 illustrates the distance d between cursor images $d = v/c(v) = m(v/m)^{1-k}$ resulting from this definition.
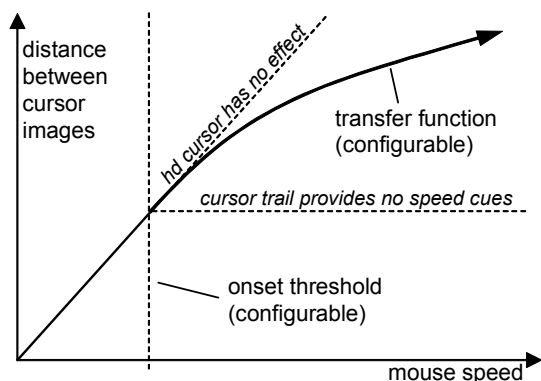
Figure 4: Hd cursor preserves the distance between cursor images as a visual cue for cursor speed by growing cursor distances with mouse speed.

4.2.4   Smooth cursor path interpolation

When hd cursor is used, the peak number of cursor images on the screen will typically exceed the number of mouse location values the mouse device can provide, even if the full hardware sampling rate is used (around 120Hz with some PC mice, for example). The locations of all additional cursor images then have to be determined using interpolation.

The choice of the interpolation function requires additional attention. Simple linear interpolation gives cursor paths a polygon-shaped appearance, which not only exposes users to the presence of the "fake" cursor images—it may also make it harder for users to visually track the cursor across the abrupt corners of the curve. Smooth interpolations are typically created using higher level interpolations, such as cubic splines. Unfortunately, cubic splines require knowledge about path segments on both sides of each segment; in the case of the hd cursor this means past *and future* mouse coordinates. This type of interpolation is unsuitable, as waiting for later samples would require delaying the rendering of the cursor, which in turn would reduce the perceived responsiveness of the mouse.

Hd cursor therefore uses interpolation solely based on past and current cursor positions. Since all cursor images are displayed only for a single frame, it is not possible to modify segments of cursor images already rendered, so that hd cursor renders exactly one curve segment per frame. For our hd cursor implementation, we chose to use a Bezier curve of $n=2$, i.e. $P(z) = P_{t-1}(1-z)^2 + P_c2z(1-z) + P_tz^2$; with P denoting the vector containing the interpolated cursor coordinates, P_{t-1} and P_t being the previous and current mouse coordinates, P_c denoting the Bezier curve's attraction point, and $0<z\leq1$ and. In order to obtain first order continuity, attraction points are placed in extension of the tangent to the previous segment. Alternatively, a spline with a clamped and a relaxed end may be used. For details on Bezier and Spline algorithms, see (Rogers and Adams, 1990).

Hd cursor interpolates cursor density and cursor size similarly, leading to a non-even distribution of cursor images within segments. This interpolation causes the cursor image in the middle of the 8th frame in Figure 3g to be shifted to the left. In order to always provide a cursor image at the "real" cursor position, the interpolation of cursor locations may have to be rounded.

5 Lab user study

To objectively evaluate performance using the high-density cursor, we performed a user study using a Fitts' Law task. Fitts' Law is a standard method for evaluating, optimizing, and studying properties of pointing devices and techniques that is well-accepted by the HCI community (MacKenzie, 1992; Douglas, Kirkpatrick & MacKenzie, 1999). We tested three versions of hd cursor against a control of the default system cursor.

5.1   Pre-study/interfaces

In order to obtain a useful set of candidate interfaces for the study, we conducted an informal survey prior to the actual user study. Fourteen coworkers participated in this pre-study. Participants were given a prototype version of high-density cursor that allowed configuring onset, transfer function, and cursor scaling. After experimenting with hd cursor in a test arena and trying out the individual options, participants had to choose one set of "favorite" settings and report these settings to the experimenter. Returned settings were collected and clustered by similarity.

Onset parameters clearly fell into the two equally popular clusters 12-17 pixels/frame and 35 pixels/frame. We therefore chose to represent these two onset values in the user study. Transfer functions chosen by the participants ranged from constant cursor density to cursor distance growing with the square root of the mouse speed. Participants' choice of transfer function was uncorrelated to onset. Since the square root setting was most popular among participants, we used this value in the study. Preferences for scaling values ranged from no cursor growth to 300% cursor growth. We chose the median (200%) for inclusion in the study.

The resulting three high-density cursor interfaces were defined as follows. The *HD_conservative* interface used an onset of 35 pixels/frame, the square

root transfer function, and no cursor scaling. The *HD_tripleDensity* interface was identical to the *HD_conservative* interface, but used an onset of 12 pixels/frame, giving it a three times denser trail than the *HD_conservative* interface. The *HD_plusScaling* interface was identical to the *HD_conservative* interface with the only difference that during fast motion it scaled the cursor as much as 200%. The *control* interface was implemented as the regular Microsoft Windows mouse cursor.

Figure 5: The three-monitor setup with spreadsheet background used in the experiment.

The experiment was run on a PC running WindowsXP with a triple 18" LCD monitor setup (Figure 5), each having a resolution of 1280x1024 pixels at 60Hz refresh rate, and driven by a Matrox Parhelia graphics card. The high-density cursor interfaces used in the study were implemented in C++. The Microsoft IntelliMouse was set to its highest available mouse speed.

5.2   Hypotheses

Our main hypothesis was that participants would perform faster when using any of the HD interfaces than when using the control interface. Beyond that, we expected participants to perform faster when using the HD_tripleDensity or HD_plusScaling interfaces than when using the HD_conservative interface, as these interfaces add additional visual weight to the cursor.

5.3   Participants

We recruited 7 participants from the community and 5 coworkers between the ages of 25 and 50, for a total of 12 participants. All had experience using multi-monitor computer displays at work or home and all rated themselves as advanced computer users. All had normal or corrected to normal vision with no color blindness and all were right handed and used the mouse in their right hand.

5.4   Method

The Fitts' Law task was administered using a modified version of WinFitts (courtesy of the Dept. of Computer & Information Science, University of Oregon). Participants read a document with general instructions for the task. For each mouse cursor condition, participants were allowed to play with the mouse and acceleration parameters for a short time and then performed a block of practice trials to familiarize them with the task and mouse cursor settings. They then performed a block of trials for that condition. Each block consisted of a single trial for each of the 25 distance-width combinations at 6 different target angles for a total of 150 trials per block. Error conditions (where a target was missed) were repeated in a random order at the end of the block. The Fitts parameters used in the experiment were: Width: 6, 12, 25, 50, 100 mm; Distance: 125, 250, 500, 750, 1000 cm; Angle: 0, 9, 171, 180 189, 351 degrees from start point. This yielded Fitts index of difficulty measures ranging from 1.17 to 7.39 bits (according to the formula ID=$\log_2$(D/W +1). Targets were placed on a background of a spreadsheet consisting of tables containing repeated numbers (Figure 5), to simulate a typical noisy work environment.

5.5   Results

5.5.1   Movement Time

All data analyses for movement times were performed on the log-transformed movement times to normalize the typical skewing associated with response time data. These were converted back to normal time for all figures below to make the results more intelligible. Movement times were first cleaned by removing error trials and outliers (movement times greater than 3 standard deviations larger than the mean for each condition, about 0.2% of all trials). We collapsed across angle to yield the means of 6 repetitions of each distance-width combination for each participant. The error rate was very low for all conditions, ranging from 1.1-1.8%.
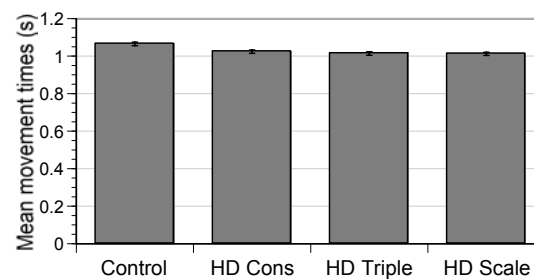
Figure 6: Mean movement times for each condition (+/- standard error of the mean).

We performed a 4 (Condition) x 5 (Distance) x 5 (Width) Repeated Measures ANOVA on the log-transformed movement data. The typical finding of increased movement time as D and W increase was confirmed (i.e., as the task got more difficult: for D, $F(4, 44) = 707$, $p \ll 0.01$; for W, $F(4, 44) = 763$, $p \ll 0.01$). There was also an interaction between Distance & Width, $F(16, 176) = 27.5$, $p \ll 0.01$—as D increased, the size of the target, W, had a smaller effect on movement time.

More interestingly, there was a significant effect for Condition, $F(3, 33) = 4.85$, $p < 0.01$ (see Figure 6). We performed a follow-up series of paired t-tests (a within-subjects comparison of each visualization condition vs. the control) to determine which conditions were different from the control. All three high-density cursor conditions were significantly faster than the control (see Table 1), which confirms our main hypothesis. There were no significant differences between hd conditions, so our second hypothesis was not confirmed by this study.

Condition	Move Time(s)	IP (bits/s)	t (299)	p<<
Control	1.14	3.73	--	--
HD_conservative	1.09	3.88	6.47	0.001
HD_tripleDensity	1.08	3.90	8.10	0.001
HD_plusScaling	1.07	3.91	8.93	0.001

Table 1: Mean movement times and Fitts Index of Performance (IP) for each condition. To the right are paired t-tests of movement times for each visualization condition versus the control (default system cursor).
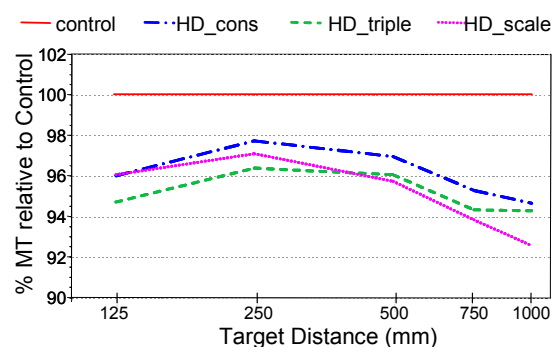
Figure 7: Mean movement time for each condition expressed as the % relative to the control for each distance moved.

It is especially interesting to look at differences between conditions for different distances. Because the range of movement times is much greater than the differences between the conditions, it is difficult to clearly see these differences. To make up for this wide variation, we normalized the movement time to the control condition by showing the movement time at each distance as a percentage of the movement time in the control condition. At the longest distance, the HD_plusScaling visualization resulted in about 7% faster target acquisition times (Figure 7).

5.5.2 Questionnaire & Subjective feedback

After each condition, participants were asked questions about whether they noticed anything different about the cursor or mouse movement from what they were used to and whether they could explain the difference. Very few participants were able to correctly identify the high-density conditions. Only 2 participants described a trace behind the cursor in the HD_conservative condition and 3 in the HD_tripleDensity condition. However, 7 participants detected scaling (HD_plusScaling), as this was much more salient.

In every condition including the control, many participants claimed they observed (nonexistent) changes in the acceleration or targeting of the mouse. They were often very surprised to find that the only difference in the conditions was the visual feedback given to them. Almost all of them commented that they were sure that the acceleration or targeting behavior of the mouse was changing; the mouse just "felt" different. It should be noted that the background (spreadsheet) made it particularly difficult to detect these changes. Of course, this also made it even more difficult to locate the cursor when it was static.

At the end of the last condition, participants were asked which condition was their most and least liked (see Table 2). The HD_plusScaling condition was by far the favorite, receiving 6 votes for "Liked Most" and none for "Liked Least." The Control was the least liked, with 7 votes for "Liked Least" and none for "Liked Most."

Condition	Liked Most	Liked Least
Control	0	7
HD_conservative	2	3
HD_tripleDensity	2	0
HD_plusScaling	6	0

Table 2: Ratings of favorite and least liked conditions in the study. Note that 2 participants reported no preference.

Finally, participants were debriefed concerning exactly what was manipulated in each condition. They then finished the questionnaire with some free

response questions concerning specific problems they had, improvements they'd like to see, and any other thoughts or impressions they may have had.

6 Study summary and conclusions

The differences found in this study were not extremely large, but were statistically significant. As predicted, the largest effect of the high-density cursor visualization was in very large mouse movements, but it was surprising to see across-the-board improvement as well.

Despite their relatively modest size, the findings reported above can be expected to have a significant impact on people's daily work. Target acquisition tasks, like those tested in the user study, occur as part of virtually all everyday computer tasks. The speed up caused by the high-density visualization can therefore be expected to lead to small, but omnipresent time savings, especially as display sizes continue to increase.

Finally, the confirmed unobtrusiveness of hd cursor is of particular importance for the application of the technique in a real-world operating system or mouse driver. As the majority of users were not able to tell the presence of the inserted cursor images, introducing hd cursor into an existing system can be expected to pass without causing legacy issues or need for adjustment among users.

Work on refining hd cursor continues. We are currently running a long term study in which forty-five coworkers have installed hd cursor on their machines. This study combines usage logging and questionnaires to assess the usefulness of hd cursor in daily work, as well as finding which parameters participants prefer when using it over a longer period of time.

Acknowledgements

References

Baudisch, P., Good, N., Bellotti, V., and Schraedley, P. (2002) Keeping Things in Context: A Comparative Evaluation of Focus Plus Context Screens, Overviews, and Zooming. In *Proc. CHI'02*, pp. 259-266.

Card, S. K., Pirolli, P., Van Der Wege, M., Morrison, J., Reeder, R. W., Schraedley, P. and Boshart, J. (2001). Information Scent as a Driver of Web Behavior Graphs: Results of a Protocol Analysis Method for Web Usability. In *Proc. CHI 2001*, pp. 498-505.

Chang, B-W. (1993) Animation: From Cartoons to the User Interface. In *Proc. UIST'93*. pp. 45-55.

Conner, B. and Holden, L. (1997) Providing a Low Latency User Experience in a High Latency Application. In *Symposium on Interactive 3D Graphics 1997*, pp 45 ff.

Cook, R.L (1984). Distributed Ray Tracing. *Computer Graphics 18*(3):137-145.

Czerwinski, M., Tan, D. S., and Robertson, G. G. (2002) Women Take a Wider View. In *Proc. CHI '02*, pp. 195-202.

Dachille, F. and Kaufman, A. (2000) High-Degree Temporal Antialiasing. In Proc. Computer Animation 2000, pages 49-54.

Douglas, S., Kirkpatrick, A., MacKenzie, I.S. (1999), Testing Pointing Device Performance and User Assessment with the ISO 9241, Part 9 Standard, In *Proc. CHI'99*, pp. 215-222.

Dulberg, M.S., St. Amant, R., and Zettlemoyer, L.S. (1999), An Imprecise Mouse Gesture for the Fast Activation of Controls. In *Proc. Interact '99*, pp. 375–382.

Geißler, J. (1998) Shuffle, Throw or Take It! Working Efficiently with an Interactive Wall. In *CHI '98 Summary*, pp.265-266.

Grudin, J. Partitioning digital worlds: focal and peripheral awareness in multiple monitor use. In *Proc. of CHI'01*, pp. 458-465.

Kensington Mouseworks v. 5.61 (2001). http://www.kensington.com/html/1461.html

MacKenzie, I.S. (1992), Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction 1992*. 7: pp. 91-139.

McGuffin, M. and Balakrishnan, R. (2002) Acquisition of expanding targets In *Proc. CHI 2002*, pages 57–64.

Moyle, M. and Cockburn, A. (2002) Analysing Mouse and Pen Flick Gestures. In *Proc. of the SIGCHI-NZ Symposium On Computer-Human Interaction 2002*.

Rogers, D.F. and Adams, J.A. (1990) *Mathematical Elements for Computer Graphics,* McGraw Hill.

Sibert, L.E. and Jacob, R. (2000), Evaluation of Eye Gaze Interaction. In *Proc. CHI'00*, pp. 281-288.

Swaminathan, K. and Sato, S. (1997) Interaction design for large displays. In *Interactions* 4(1):15 – 24.

Thomas, B. H. and Calder P. R. (2001). Applying cartoon animation techniques to graphical user interfaces. *ACM Transactions of Computer Human Interactions*, 8(3):198-222.

Zhai, S., Morimoto, C., and Ihde, S. (1999), Manual and gaze input cascaded (MAGIC) pointing. In *Proc. CHI '99*, 1999, pp. 246-253.