Speeding Pointing in Tiled Widgets: Understanding the Effects of Target Expansion and Misprediction

Jaime Ruiz and Edward Lank
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, Canada
{jgruiz,lank}@cs.uwaterloo.ca

ABSTRACT
Target expansion is a pointing facilitation technique where the users target, typically an interface widget, is dynamically enlarged to speed pointing in interfaces. However, with densely packed (tiled) arrangements of widgets, interfaces cannot expand all potential targets; they must, instead, predict the user's desired target. As a result, mispredictions will occur which may disrupt the pointing task. In this paper, we present a model describing the cost/benefit of expanding multiple targets using the probability distribution of a given predictor. Using our model, we demonstrate how the model can be used to infer the accuracy required by target prediction techniques. The results of this work are another step toward pointing facilitation techniques that allow users to outperform Fitts' Law in realistic pointing tasks.

Author Keywords
Pointing, target expansion, Fitts' Law, tiled widgets, human performance

General Terms
Human Factors, Performance

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

Pointing, with a mouse, electronic stylus, touchpad, trackpoint, or trackball, is a frequent task in modern graphical user interfaces. Due to the frequency of pointing, even a marginal improvement in pointing performance can have a large effect on a user's overall productivity. As a result, researchers have explored various techniques to speed pointing, including bubble cursors [5], pointer warping [6], manipulation of motor space [7, 13, 2], and target expansion [10, 14]. While these techniques can speed pointing for sparsely spaced targets on a computer display, McGuffin and Balakrishnan [10] note that these techniques provide less benefit for denser target arrangements, and, in the case of tiled targets, many provide no benefit. Several researchers have explored designing pointing facilitation techniques for tiled target arrangements [10, 14] with, as of yet, no success.

Tiled targets are, however, common in graphical interfaces. For example, widgets are frequently contained by toolbars, ribbons and menus, all multi-widget containers where no space exists between widgets. As well, widgets are not the only candidate targets in interfaces. The cells in a spreadsheet program or the words and letters in a word processing program also constitute legitimate locations for a user to target in a graphical interface. If these programs are displayed in full screen mode, the computer monitor is almost entirely covered with a tiled arrangement of potential targets.

In this paper, we explore the performance of one pointing facilitation technique, target expansion, for tiled targets. Target expansion works by expanding the target a user is pointing at on the display so the target is easier to acquire. However, as noted by McGuffin and Balakrishnan [10] and Zhai et al. [14], when targets are densely arranged on the screen, expanding all targets in a user's path results in no pointing advantage. With dense or tiled target arrangements, performance gains are only possible if one could reasonably predict the trajectory of the cursor such that the system can identify, in real time, the target a user is going to select [10]. To support target expansion for tiled targets, we need some technique for predicting the endpoint of a user's pointing motion in real time.

One sophisticated technique for endpoint prediction is *kinematic endpoint prediction* (KEP) [11, 8]. KEP uses the motion profile of a user to define a region of interest on the display. However, this region of interest is larger than an individual target: Ruiz and Lank [11] show that the region predicted by KEP is defined by a normal probability distribution. The normal distribution is centered on the maximally likely target, and targets surrounding that region have decreasing likelihoods. As a result, while the predicted target may be the actual intended target, surrounding targets are also likely.

Given that KEP cannot predict a single onscreen target, in this work we explore two issues associated with pointing in tiled target arrangements. First, is it possible to expand

IUI'10, February 7–10, 2010, Hong Kong, China.

a small group of targets rather than an individual target to boost predictor accuracy? Second, given expansion of a set of targets, does the use of the KEP improve pointing performance?

It may be the case that the present KEP accuracy is sufficient to observe an improvement in performance in pointing tasks when expanding multiple targets. For example, if the KEP correctly predicts the user's target, the user's intended target will be at the center of the expansion region, and targeting will likely be faster. However, because the KEP predicts a region, offset errors are common, and, with an offset error, the expansion region will be centered on a target other than the user's desired target. If the offset is sufficiently small, the user's desired target will be expanded, but the target expansion will be confounded with target displacement – targets closer to the center of the expansion region will push the user's desired target either toward or away from the user. If the algorithm's prediction is off by a distance greater than the expansion region, a large offset error, the user's desired target will not be expanded *and the target will be shifted.* There is a probable benefit if the KEP algorithm is correct and expands a set of targets centered on the user's desired target. It is highly likely that large offset errors (where the user's desired target is shifted and not expanded) will increase the cost of a pointing task. The effect of small offset errors, where a target is enlarged and shifted on the display, is unknown. In this paper, we seek to determine whether Ruiz and Lank's KEP accuracies result in an *overall* improvement in pointing performance. We also wish to examine the costs of small and large-offset errors.

Through results from two experiments presented in this paper, we show that it is possible to expand a small set of targets on the computer screen to improve pointing performance. We also show that, when expanding a region, the benefits of expansion are affected by the target shift, i.e. the size of the offset error. We find a limit on shift of about 80 pixels on our 24 inch 1920x1200 displays. Finally, we demonstrate that, within the expansion region limit, any endpoint predictor must have an accuracy greater than 56.5% to realize a net benefit from expanding targets.

This paper is organized as follows. First we describe related work on expanding targets and endpoint prediction. Next we describe our design decisions for expanding a group of targets using screen expansion. We then describe and present results from a pilot study conducted to determine if performance benefits can be accrued when enlarging a candidate set of targets. This is followed by an experiment that uses a real time implementation of the KEP as described in Ruiz and Lank [11] for tiled targets. We conclude with a discussion on the implications and future directions of our work.

BACKGROUND

Fitts' Law [4] relates pointing time to target size and distance through a logarithmic term referred to as the *Index of Difficulty* or *ID*:

$$T = a + b\ log_2\ \left(\frac{A}{W} + 1\right) \quad (1)$$

In the above equation, A represents the distance to the target, and W represents the size of the target.

Given the reliability of Fitts' Law to predict movement times in interfaces [9], work in pointer facilitation can be categorized by techniques that reduce the distance to the target, increase the width of the target, or both decrease the distance and increase target width (see [1] for a review of these techniques). In this paper, we focus on techniques that increase the size of the target on the display.[1]

McGuffin and Balakrishnan [10] investigated the potential performance benefits of expanding target size. In their study they demonstrated that users are able to take advantage of enlarged targets even if target expansion occurs with only 10% of the distance left to travel. They also conclude that the required movement time to acquire a target can be accurately modeled by Fitts' law using the expanded target width to calculate the target's index of difficulty. In a replication study, Zhai et al. [14] pointed out that in McGuffin and Balakrishnan's experiment, users could anticipate the enlarging of a target. Therefore, observed benefits may be a result of anticipation of a larger target or may result from the enlarged target. Zhai et al. added additional conditions in which targets would expand, shrink, or remain static to prevent users from anticipating the final target size. Their results support McGuffin and Balakrishnan's results that users are able to take advantage of an expanding target as late as 90% of a movement even if the user is unable to anticipate the final target size. Despite the advantages of expanding targets on the display screen, facilitating the selection of tiled targets remains an open challenge. Unlike in isolated targets, where targets expand to fill empty screen space, in a tiled target arrangement targets cannot expand without either obstructing other neighboring targets or causing neighboring targets to be displaced on the screen.

When one considers target expansion of tiled targets, there exists a question of which targets to expand. Expanding every target is clearly no longer an option, as both McGuffin and Balakrishnan[10] and Zhai et al.[14] note. If every target is doubled in size, the screen space consumed by the tiled target arrangement doubles in size consuming too much screen real estate and will spill off the sides of the computer display. Furthermore, while the desired target becomes larger, all intervening targets also become larger, thus increasing the distance to the user's desired target. The benefits of a larger target are exactly offset by the increase in distance. For example, if all targets doubled in size on a display, to reach his or her desired target the user would need to traverse the intervening targets on the display. These intervening targets also double in size, effectively doubling the distance the user needs to travel to reach his or her goal (this assumes a fully tiled display). As shown in Equation 1, if both A and W are doubled, the overall pointing time remains unchanged from the initial distance and size. It is for precisely these reasons

[1]Targets can also be expanded in motor space (but not visually) [6] or visually (but not in motor space) [3]. While both of these techniques have shown benefits in isolated pointing tasks, we focus on display expansion in this paper.

that, to enable target expansion in tiled target arrangements, a predictor is needed to select a candidate target for expansion.

Recent work on endpoint prediction [8, 11] may enable pointing tasks even for dense or tiled target arrangements by predicting the target of a user's pointing motion. Using established kinematic models of motion, the researchers derived a kinematic endpoint prediction (KEP) algorithm. Lank et al. showed that their KEP was able to predict a user's target 42% of time and an adjacent target an additional 39% of the time (assuming tiled arrangements of targets) given a relatively small set of target sizes and distances. In follow up work, Ruiz and Lank [11] demonstrated that distance of motion has a significant effect on predictor accuracy. Therefore, while the KEP algorithm predicts a user's target 42% of the time for the distances tested by Lank et al., the accuracy decreases as the distance of motion increases. Ruiz and Lank also demonstrated that the region identified by KEP is defined by a normal probability distribution around the predicted endpoint. The standard deviation of the probability distribution is linearly correlated with distance of motion and is approximately 10% of motion distance.

Regardless, one drawback to any prediction technique is misprediction. For example, Lank et al.'s prediction technique results in correct prediction of their target sizes only 42% of the time and is accurate $\pm$ one target an additional 39% of the time. Ruiz and Lank extend this work, and show that the KEP yields a probability distribution over a region. While the KEP could be augmented with domain knowledge or user task models to improve its predictive power, there exists an open question on predictor accuracy. Clearly, more accurate target predictors are always to be preferred over less accurate predictors. However, how accurate must a predictor be to be useful for target expansion with tiled candidate target arrangements? Is the current state of the art in endpoint prediction, the KEP, accurate enough for generalized targeting tasks in interfaces? We seek to address these questions in the remainder of this paper.

TARGET EXPANSION FOR TILED TARGETS

A goal of this paper is to design expanding targets for tiled arrangements, for example, arrangements of targets in toolbars, menus, and ribbons in programs such as spreadsheets and word processors. As mentioned above, enlarging all targets in the user's path will result in no performance gain. Therefore, to enable target expansion in tiled target arrangements, a predictor is needed to select a candidate target for expansion. Given that KEP identifies a region of interest on the display and the region is typically larger then that of the user's intended target, the predicted target may not be the user's intended target.

To overcome the frequency of offset errors in endpoint prediction and increase the likelihood that the user's intended target is enlarged, we propose expanding a candidate set of targets. Expanding a group of targets will obviously improve the probability that any individual candidate target will be included in the expanded set. However, expansion of a group also results in having a greater effect on neighboring targets. In this section we describe our design decisions for expanding a group of targets.

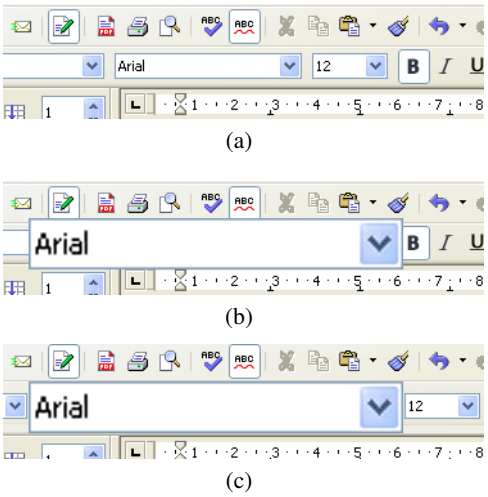


Figure 1. Font and point size selection widgets common in word processing programs. (a) The original unexpanded widget. (b) Expansion of the font widget using occlusion resulting in 100% occlusion of the point size widget. (c) Expansion of the font widget using displacement.

Due to the limited amount of screen space around tiled targets, expansion of a target will have an effect on neighboring targets. There are two possible effects caused by the expansion of a target in tiled targets: occlusion and displacement.

Occlusion has been suggested as a technique to avoid excessive sideways shift of targets at the cost of interfering with the visibility of neighboring targets [10]. Previous designs using occlusion for tiled targets have been limited to targets of equal size. Therefore, doubling the height and width of one target results in a 50% occlusion of the adjacent targets. However, if target sizes differ, up to 100% occlusion of neighboring targets can occur. For example, a majority of word processing applications' toolbars include widgets for font and point size selection similar to the one shown in Figure 1(a). As shown in Figure 1(b), expansion of the font selector widget would result in the complete occlusion of the point size selection widget. Expanding multiple targets exacerbates the occlusion problem. If three targets are expanded to double their original width and height, then 1.5 targets of equal size are occluded on either side.

Due to both the heterogeneous size of widgets found in graphical user interfaces and our desire to expand multiple candidate targets, both of which would result in the total occlusion of possible targets, we choose to examine the use of displacement when expanding multiple targets in tiled arrangements. Displacement occurs when an expanding target causes neighboring targets to shift in order to make room for the target's new size. However, displacement is also not without limitations. Expanding a target that is not the user's intended target, what we call an offset error, results in the user needing to acquire a target that has shifted position. In

the example of the font and point size widgets, expansion of the font selection widget results in the point size widget being displaced 100% of it's width (see Figure 1(c)).

Expanding multiple targets will cause an even larger co-linear displacement compared to expanding a single target, because the group of targets consume more space than a single target. Finally, visual disruption of the display is caused by expanding multiple targets, and this visual disruption may negate the performance benefits of expanding the target size.

Because of these potential risks, we describe two studies of target expansion with endpoint prediction. The first uses a simulated predictor with high accuracy, while the second uses the real-time KEP algorithm as described by Ruiz and Lank [11].

EXPANSION WITH SIMULATED ENDPOINT PREDICTION

The goal of our first experiment is to determine whether or not it is possible to expand a candidate set of targets, and whether any amount of target displacement is possible. To do this, we use a simulated predictor based on the accuracies reported in the original KEP paper of Lank et al. [8]. The original KEP was reported to have accuracies of 42% for the user's desired target and it predicted an adjacent target an addition 39% of the time. Our goal was to determine whether simulating these accuracies would result in a measurable performance improvement in pointing tasks. We also wished to determine how great the cost of displacement is with small offset errors (off-by-one) where the user's target is enlarged and shifted slightly, and large offset errors (off-by-two) where the user's target is not enlarged and is shifted.

In this section, we describe an experiment that shows that, if we expand a candidate set of three targets - the predicted target $\pm$ one target - in a tiled arrangement, it aids target acquisition based on Lank et al.'s initial probabilities. As aspects of this experimental design are replicated in our second study, we spend some time on the specific details of this study.

Method

Apparatus

The experiment was conducted on a generic desktop computer (P4, 2.0GHz) with a 24-inch 1920x1200 LCD display running custom software written in C# using Microsoft .NET. Input was collected using a Wacom Intuos3 five button mouse on a 12x19 inch tablet with cursor acceleration turned off.

Task

To test the performance of expanding a set of targets,we devised an experimental task that mimics that of previous work on target expansion [10, 14, 2]. Our task differs only in its use of seven tiled targets instead of an isolated target and in that target expansion occurs over more than one target.

The task for our experiments was a simple one-dimensional pointing task. Initially a green starting rectangle was randomly displayed close to one of the horizontal boundaries of the screen. The task began when the participant used the cursor to click within the starting location. At that time, seven tiled targets, would appear on the display orthogonal to the starting position. Participants were required to move the cursor to the red target and use the mouse button to click on the target. A successful target acquisition (i.e., clicking within the target region) was indicated by the target changing color. Users were told to acquire the target as quickly and accurately as possible, similar to other Fitts' Law tasks. To prevent users from always targeting the center of the multi-target widget, the location of the desired target was varied between the third, fourth, and fifth target.

Similar to previous work in expanding targets [14, 10], task ID ranged from 3.17 to 7.01 bits. However, our experiment contains fifteen Distance/Width (*D/W*) combinations resulting from presenting each task ID at three different distances. The three distances were chosen to correspond to close movements (512px), average movements (1024px), and distant targeting tasks (1526px). Our fifteen combinations of D/W (in screen pixels) were 512/4, 512/8, 512/16, 512/32, 512/64, 1024/8, 1024/16, 1024/32, 1024/64, 1024/128, 1536/12, 1536/24, 1536/48, 1536/96, and 1536/192.

Experimental Conditions

In our pilot study, there are two conditions. The first condition is a *Static/No Expansion* condition where the target size never changes during the movement. The other condition is an expansion condition where the candidate targets' widths expand by a factor of two when the cursor has covered 80% of the target distance (D). The targets expanded around a predicted target's center. To simulate predictor behaviour, 40% of the time the predicted target and the user's intended target were the same; 20% of the time one target before and 20% of the time one target after the intended target was the predicted target; and 10% of the time two targets before and 10% of the time two targets after were the predicted targets. As shown in Figure 2, mispredictions resulted in the intended target being shifted. If an off-by-two error occurred, the user's target was shifted and was not expanded.

Participants

Twelve adult volunteers, seven male and five female, between the ages of 18-32 (mean=24.7, SD=4.1) participated in the study. All participants were affiliated with a local university and received a $10 gift certificate for a local coffee shop for their participation.

Procedure and Misprediction Conditions

The experiment consisted of three blocks: one practice block (no expansion) and two experimental blocks: no expansion and expansion using the simulated predictor. The practice and no expansion block consisted of 15 D/W combinations presented six times (twice for each possible target location), resulting in 90 tasks per block. The expansion block consisted of each D/W combination being presented to the user ten times resulting in 150 tasks per block. The order of presentation of the combinations was randomized and the order of the experimental blocks was counterbalanced.

For the ten pointing tasks at each D/W combination, we introduce misprediction conditions to simulate the behaviour of Lank et. al's predictor. These conditions correspond to the conditions shown in Figure 2.

- *Correct Prediction:* In 4 of the 10 movements at each D/W combination, the predicted target was the user's intended target.
- *-1 Prediction:* In 2 of the 10 movements at each D/W combination, the predicted target was the target immediately before the user's intended target, along the user's path of motion.
- *+1 Prediction:* In 2 of the 10 movements at each D/W combination, the predicted target was the target immediately beyond the user's intended target.
- *-2 Prediction:* In 1 of the 10 movements at each D/W combination, the predicted target was two targets before the user's intended target, along the user's path of motion.
- *+2 Prediction:* In 1 of the 10 movements at each D/W combination, the predicted target was two targets beyond the user's intended target, along the user's path of motion.

The visual effect of each misprediction condition on the intended targets in screen space is illustrated in Figure 2.

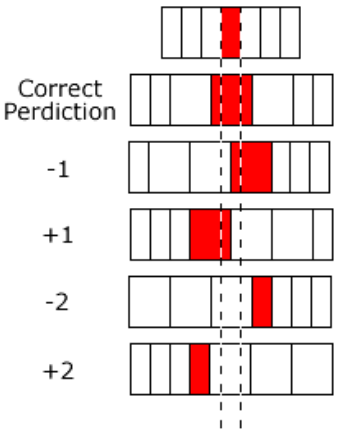


Figure 2. Illustrations of each of the five experimental conditions for expansion condition.

Results

Of the 2880 tasks recorded, 4.1% resulted in the user not correctly hitting the target. These tasks were removed from our analysis.

Figure 3 illustrates the overall movement time for experimental versus control conditions by ID. Figure 4 further segments movement time using each of the expanding conditions (Correct, $+1$, -1, $+2$, -2). Analysis of variance shows a significant effect for condition (expansion or no expansion) ($F_{1,11} = 8.51, p < 0.01$), misprediction condition (Correct, $\pm$ 1, $\pm$ 2) ($F_{4,8} = 27.73, p < 0.01$), and ID ($F_{4,8} = 396.20, p < 0.001$) on movement time. We also

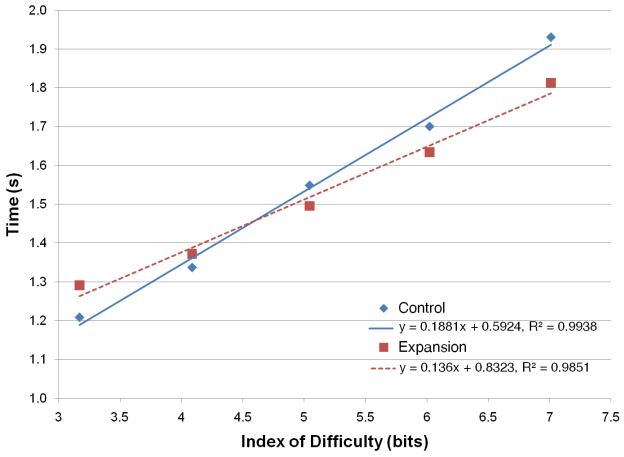


Figure 3. Movement times by Index of Difficulty by expanding condition for the user trial.

observed condition * misprediction interactions ($F_{4,20} = 24.15, p < 0.001$), and condition * ID interactions ($F_{8,16} = 5.97, p < 0.01$) on total movement time.

Analyzing Mispredictions

Results from our user trial indicate that even in the presence of mispredictions, users benefit from target expansion. Analysis of variance for tasks in the expansion condition show a significant effect for ID ($F_{4,8} = 154.18, p < 0.001$), misprediction error condition ($F_{4,8} = 59.21, p < 0.001$), order ($F_{2,10} = 18.17, p < 0.01$), and ID * misprediction interactions ($F_{4,24} = 2.507, p < 0.05$) on movement time. Post Hoc analysis using Bonferroni correction shows the -2 misprediction condition to result in the slowest mean movement time followed by the $+2$ condition. Post Hoc analysis also shows the -1 misprediction condition to be significantly slower than the the correct prediction and $+1$ condition. However, Bonferroni correction does not show a significant effect on movement time for the correct prediction and $+1$ misprediction condition. Finally, in all cases, when the target expands, the user outperforms the control condition. As well, overall the expansion condition outperform the control condition based on the simulated error rates we used. Qualitatively, we note that for all but the lowest IDs (largest, closest targets) the experimental condition was faster than the control condition.

As described by McGuffin and Balakrishnan[10], the maximum expected benefit of expanding a target can be calculated using Fitts' Law with the target's expanded size. Using the target's final width to calculate the task's Index of difficulty, represented as ID_{final}, we plot the effect on movement time by misprediction alongside the maximum expected benefit represented as a solid line (Figure 4). This line was calculated using Fitts' law coefficients from the static condition in Figure 3 assuming that we always predict and expand the correct target. In Figure 4, we see that correct prediction slightly outperforms the maximum expected benefit, and that the +1 condition, also outperforms the expected benefit for all but the lowest ID. The -1 condition performs, on aver-

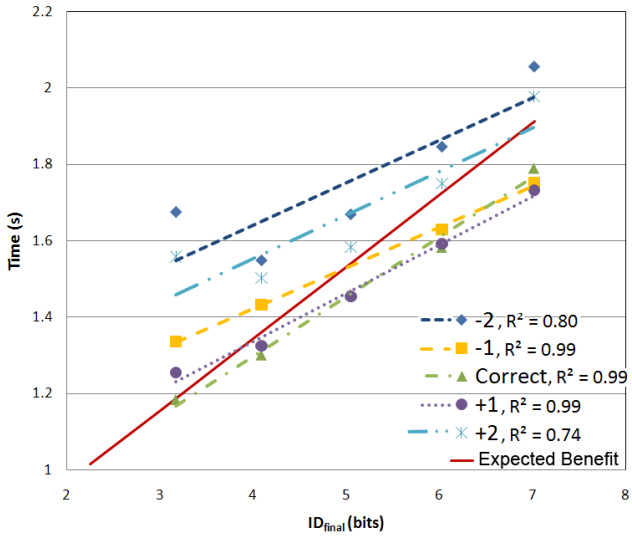


Figure 4. Movement time by final Index of difficulty for the screen expansion condition.

age, at the lower bound (slightly better for high ID tasks and slightly worse for low ID tasks). Finally, the +2 and -2 conditions do perform worse than the optimal expected benefit, but these conditions are relatively rare with current predictor accuracy (10% probability each).

REAL-TIME PREDICTION AND TARGET EXPANSION

Our results in our pilot study demonstrate that expanding a candidate set of targets on the computer screen coupled with optimistic endpoint prediction accuracies improves pointing performance in a tiled target pointing task. We have shown that the screen expansion of multiple targets improves pointing performance and that users are able to capitalize on an enlarged target despite the presence of mipredictions and target shifts. However, as mentioned above, follow up work by Ruiz and Lank [11] demonstrated that while high accuracies could be replicated at the distances used by Lank et al., as distance increased the distributions of predicted endpoints also increased, resulting in lower probabilities of the correct target being predicted. Therefore, at distances further than 512-pixels, even if three targets are expanded, it is much less likely that a user's target will be expanded. To increase the likelihood of target expansion in real world prediction, it is necessary to increase the size of the candidate set as distance increases.

In this section we describe an experiment conducted to examine expanding a region of targets using a real-time implementation of the kinematic endpoint prediction algorithm. In particular, the goal of the experiment is to answer the following questions:

- Will using a real-time predictor and a candidate set of targets, enable us to expand the user's intended target at accuracies defined by a normal probability distribution?
- Is there a limit to the amount of displacement that can occur before performance degrades?
- Finally, is the current state of the art in endpoint prediction accurate enough for enabling expanding targets in interfaces?

Kinematic Endpoint Prediction

To conduct this experiment, we implement the KEP algorithm and incorporate it into our pointing task. The KEP algorithm is described in [11], and is used to predict the endpoint of a user's motion. We summarize the implementation of this predictor here.

The KEP algorithm is a 3-step algorithm to determine user endpoint. The three steps are:

1. Given a partial gesture of length d toward a target, a quadratic equation is used to fit the data points $(d, s(d))$ along the partial gesture. Here d is the distance, and $s(d)$ is the speed at that distance. The quadratic equation is solved for it's 0-roots (x-intercepts). One x-intercept should occur at point $(0, 0)$, the start point. The other occurs at some distance from the start point, d_{calc}.
2. Calculate the *stability* of the predictor by comparing the previous endpoint predicted to the current predicted endpoint. If the prediction is not stable, return a value indicating an accurate prediction is not possible at this point in time.
3. If the predictor is stable, calculate the predicted percentage of gesture length completed by dividing the current distance traveled by d_{calc}. If the percentage is greater than a set threshold, return d_{calc}; otherwise return a value indicating a prediction is not possible at this time.

To return an endpoint, after curve fitting, we first estimate *stability* of our prediction. Our predictor is stable when the following equation holds:

$$\frac{l_n - l_{n-1}}{l_n} \approx 0 \qquad (2)$$

Here l_n is the current value of d_{calc}, and l_{n-1} is its previous value. Typically, this equation only approaches 0. Ruiz and Lank find that, as long as the value of the stability estimate is < 0.02, the prediction is sufficiently stable to give a reliable estimate.

If the prediction is stable, we then calculate where we are in motion. Based on previous work [10, 14], indicates that users are able to take advantage of an expanding target as late as 90% of a movement. We therefore typically predict before 90% of user movement. In our implementation of KEP, if the current distance traveled is 0.89 of d_{calc}, we return a predicted endpoint. For more details on the implementation of the KEP algorithm and the rationale for these design decisions, the interested reader is referred to [11], available on-line at the HCI lab website at the University of Waterloo, and to [8].

In Ruiz and Lank's KEP implementation, they show that the standard deviation of the predicted endpoint of a user's motion can be approximated by taking 10% of the distance a

user travels. As a result, they recommend considering candidate targets within two standard deviations ($\pm 10\%$) highly likely, i.e. 68.2% likelihood of target expansion. We base the expansion region of this experiment on this recommendation, as described below.

Method

Task

The task was the same task as described in pilot study with one exception. Instead of displaying seven targets, the whole screen was tiled with targets. As a result, the KEP predictor could choose any region on the screen as the predicted endpoint instead of limiting predictions to, for example, seven targets, as in the pilot study.

Expanding Conditions

Similar to our pilot study, our experiment has two conditions.

No Expansion/Control: Target size never changed during the movement.

Expansion: Using a real-time implementation of the kinematic endpoint prediction (KEP) algorithm, predictions were made continuously throughout the motion. At 89% of predicted distance (i.e. current distance is 89% of the predicted endpoint) a prediction was acted upon. At that time, the prediction was used to create a candidate set of targets around the predicted target by either determining the number of targets that can occupy a region defined by $\pm 10\%$ of motion distance (D) or ± 1 target, which ever was greater. For example, for an 8-pixel target at a distance of 512 pixels, a candidate set of 13 targets will be expanded because 6 targets fall within 51 pixels on each side of the predicted target. However, for a 64-pixel target at the same distance, only 3 targets will belong to the candidate set since each 64-pixel target is larger than the 51-pixel region defined by 10% of distance traveled. As mentioned above, 10% of motion distance is approximently one standard deviation of the probablity distribution for the region defined by Ruiz and Lank's KEP algorithm. Therefore, we would expect the user's intended target to be contained within a region of $\pm 10\%$ of motion distance (or two standard deviations) approximately 68.2% of the time.

Procedure

The experiment consisted of three blocks: one practice block consisting of no expansion and two experimental blocks: control and expansion. Each block consisted of 15 D/W combinations presented ten times resulting in 150 tasks per block. The order of presentation of the combinations was randomized and the order of the experimental blocks was counterbalanced.

Participants

12 adult volunteers, 8 male and 4 female, between the ages of 21-33 (mean=25.8, SD=3.6) participated in the study. All participants were affiliated with a local university and received a $10 gift certificate for a local coffee shop for their participation.

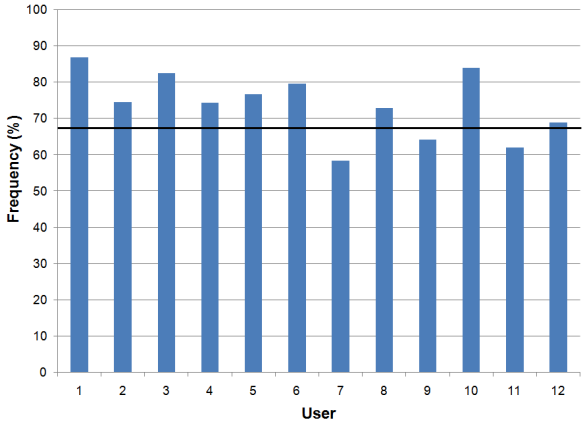


Figure 5. Frequency of the user's target expanding as part of the candidate set by user. The bold horizontal line represents the expected frequency of 68.2%

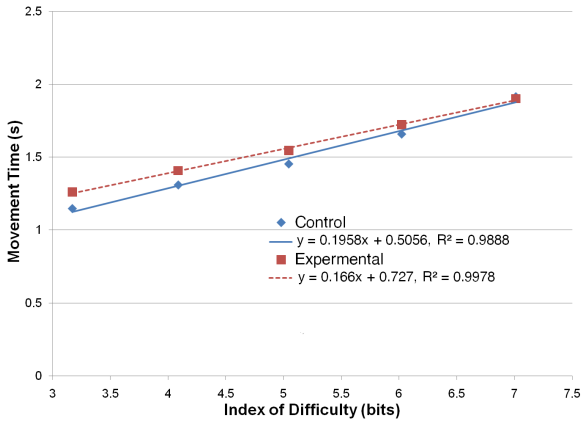


Figure 6. Movement times by Index of Difficulty by condition.

Results

Of the 3600 tasks recorded, 4.1% resulted in the user not correctly hitting the target. There was no signifficant difference between error rates in the control (5.1%) and expanding (3.1%) conditions. Errors were removed from our analysis.

In our experiment, candidate targets consisted of targets occupying $\pm 10\%$ of motion distance around the predicted target. As mentioned above, the region predicted by the KEP algorithm is defined by a normal probablity distribution around the predicted target and approximately 10% of distance represents one standard deviation of the distribution. By expanding a region consisting of $\pm 10\%$ of motion (i.e. two standard deviations), we would expect the user's target to be expanded 68.2% of the time. Results from our experiment indicated that the user's intended target was expanded 73.7% of the time. Figure 5 displays the accuracies by user. As shown in the figure, observed accuracies were typically better than expected, with only three users with frequencies below 68.2% (Users 7, 9 and 11). Accuracies reached as high as 86.9% for one user in our study.

Overall movement times by condition and ID are shown in Figure 6. Results from the experiment indicate that using the kinematic endpoint predictor to identify a candidate set of targets and expanding that set resulted in slower movement times than the control condition. To examine why, we now focus our attention on the analysis of how mispredictions, both small offset mispredictions and large offset mispredictions, affected movement time.

Analyzing Mispredictions

We examine the performance of target expansion for tiled targets by investigating the effect of offset errors on pointing speed. To perform this analysis, we normalize the time taken by motion by subtracting the average time taken in the control condition from the time taken for each individual motion. A value of 0 indicates that the movement speed was identical to the average control condition speed. Negative values indicate the user was faster than their average in the control condition. Positive values indicate that the user was slower.

We also categorize each data point into one of three categories: correct, negative off, and positive off. The correct category represents when the user's intended target was included in the expanded candidate set. Negative and positive off represent when the user's target was not in the candidate set. For Negative off, the algorithm underestimated the user's motion, so the user's intended target was moved farther away from the user. For Positive off, the predicted endpoint was beyond the user's target, so the user's target moved toward the user.

Offset error is measured in two ways, relative to the target's original width and in absolute pixels. Relative offset error defines offset error as a measure of the target's original width. For example, if the KEP predicts the correct target, then the target is not shifted and the relative offset error is 0. If the KEP is off by one target, the relative offset error would be equal to 1. For an error of k targets, the relative offset error is k.

The performance benefit/cost by relative offset error for the correct category is shown Figure 7. As illustrated by the figure, relative displacement has no correlation to the observed benefit/cost. For example, for a 4-pixel target a benefit is observed for relative offset errors as high as 11. Therefore, even though the user's intended target was displaced 11 times the original target's width (44 pixels), the user still capitalizes on the enlarged area of the target. In contrast, for 32-pixel targets, a relative target offset error as low as 1 (32 pixels) negates any benefit of an enlarged target.

Due to the lack of correlation for relative offset error on performance, we focus on absolute offset error, which, as shown in Figure 8, is more strongly correlated with benefit/cost. Absolute offset error is defined as the number of pixels the center of the target was shifted. For example, if the KEP predicts the correct target, then the absolute offset error is 0, the target is not shifted. If the KEP is off by one target, the

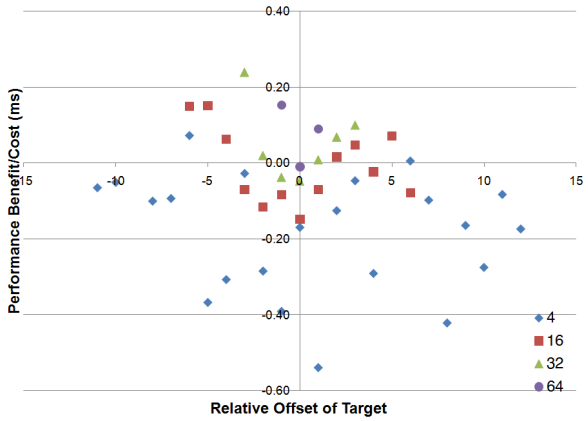


Figure 7. Performance Benefit/Cost by relative displacement of a user's target for targets of size 4, 16, 32, and 64-pixels for the correct experimental condition.

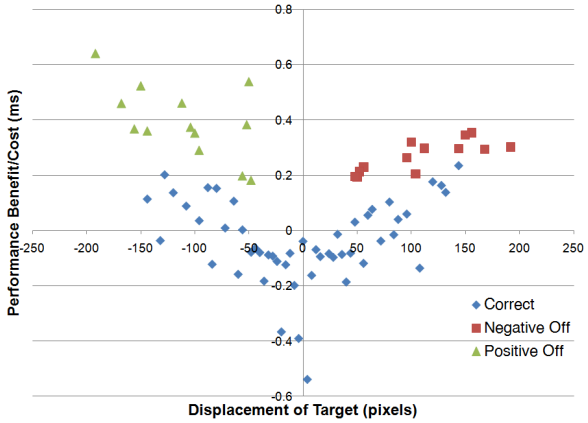
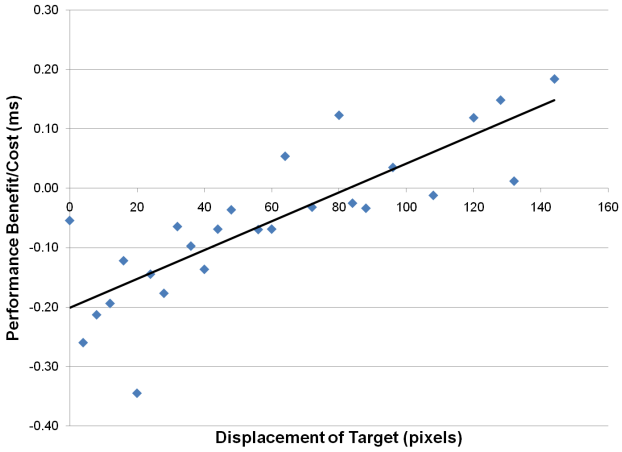


Figure 8. Performance Benefit/Cost by absolute displacement of user's target for each category in the experimental condition.

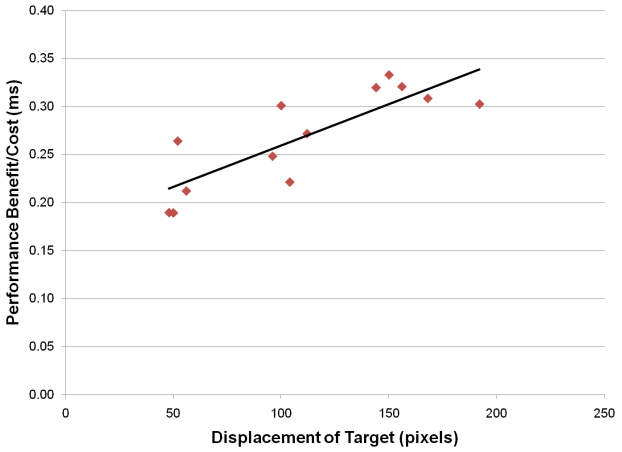
absolute offset error would be equal to W, the target width, and for an error of k targets, the absolute offset error is kW.

Examination of Figure 8 suggests that 0-pixel target displacement (displayed on the horizontal axis) is a point of reflection. Post-hoc analysis using Bonforroni correction confirms this, showing no significant difference between the negative and positive off categories. Therefore, we simplify our categories by taking the absolute value of target offset and combining the positive and negative off category into a single *Error* category. Figure 9 illustrates the normalized time taken versus offset error for each of our resulting categories, Correct, and Error. The solid line represents the best linear fit for each category.

As in the pilot study, we can calculate the expected best performance from expanding the user's intended target by using the Fitts' Law coefficients obtained from the control condition. Using these coefficients, we calculate the maximum performance benefit for our participants to be -0.20. As in the pilot study, when expansion occurs around the user's intended target, resulting in no horizontal displacement, we observed a performance benefit near the maximum benefit



(a) Correct



(b) Error

Figure 9. Performance Benefit/Cost by absolute value of the displacement of user's target for the correct and error categories.

expected by Fitts' Law. However, the performance benefits of the enlarged target quickly degrades as the user's intended target shifts. When the offset error reaches 80 pixels (shown in Figure 9(a)), the time taken to acquire a target matches the control condition. For shifts greater than 80 pixels in our experimental configuration, the user performed worse than control, even if the correct target was expanded.

The performance cost for the error condition is shown in Figure 9(b). As shown in the figure, even at the lowest level of displacement (50 pixels), the performance is worse than the control condition, suggesting a significant cost for target displacement.

DISCUSSION

While our pilot study using a candidate set of three targets and the simulated prediction accuracies reported by Lank et al. [8] resulted in promising results, they did not extend to larger sets of targets using a real-time implementation of the KEP predictor. Results from our experiment demonstrate that there is a limit to the amount a user's target can shift when expanded. On our experimental setup, the limit on target displacement was approximately 80 pixels. If displacement was greater than 80 pixels, then target expansion did not result in a net benefit. Figure 9(a) plots this expected benefit as a linear function. While the 80-pixel limit on displacement is undoubtedly a function of the resolution of our computer monitor and input device, for any display/input device a relatively simple pointing task can be used to calculate the limits on target displacement for a specific user. See, for example, Wobbrock et al.'s work [12] calculating an error function for Fitts' Law pointing tasks, where they use a simple test to calibrate users.

As we note earlier, Ruiz and Lank [11] show that the KEP algorithm can be used to calculate a normal distribution around a predicted endpoint with standard deviation, σ_i, approximately equal to 10% of the user's movement. A normal distribution can be used to calculate specific probability of a value lying between the mean and any arbitrary number of standard deviations, s, using the erf function as follows:

$$p(x) = \frac{2}{\sqrt{\pi}} \int_0^s e^{-x^2} dx \tag{3}$$

Using our 80 pixel limit for our monitor/input device configuration, we can use this to calculate the necessary accuracy of our predictor. At 0 pixels of displacement, we see a maximum benefit, calculated as about $1/5$ of a second, i.e. 0.20 seconds. This benefit shrinks to 0 seconds at 80 pixels, yielding a straight line equation of the form:

$$t_{saved} = -0.20 + \frac{0.20}{80} x \tag{4}$$

Beyond 80 pixels, our targets are not expanded, yielding a constant cost based on 80 pixels of displacement of 0.25 seconds of additional time. We can claim the following:

$$p(x) t_{saved} > 0.25(1 - p(x)) \tag{5}$$

Essentially, the probability of any time saving associated with expansion must *outweigh* the likely cost associated with 80-pixel target displacements when no expansion occurs. The probability of no expansion is exactly equal to $1 - p(x)$, where $p(x)$ is the probability that the target expands. Therefore, 80 pixels is equivalent to the number of standard deviations, s in Equation 3 such that the inequality, Equation 5 holds. Solving this analytically in Maple yields a result that $s = 0.729$. Our displacement limit, 80 pixels, is an arbitrary function of our hardware. However, the predictor accuracy must be greater than 56.5% for our maximum displacement limit. We obtain this calculation by evaluating the integral in Equation 3 with $s = 0.73$.

Considering the current standard deviations associated with the KEP, we see that only for distances less than 800 pixels will the predictor perform with sufficiently high accuracy to result in performance gain on our current experimental configuration. Therefore, for distances presented in Lank et al.'s original study [8], i.e. all distances less than 600 pixels, we would expect to see a net benefit. However, when imple-

menting the real-time predictor on a larger display, only at 512 pixels is the KEP sufficiently accurate to allow a net improvement in performance, and then only if a region of 80 pixels is expanded. Expanding smaller regions will drop the predictor accuracy and limit some of the benefit, resulting in suboptimal benefit for expanding targets.

FUTURE WORK

We are in the process of exploring variants of expansion strategies to determine promising approaches to endpoint facilitation with expanding targets. While the KEP is currently not sufficiently accurate, it may be possible to increase the reliability of the KEP by combining its probabilities with additional information. For example, histories of command usage or user task modeling could allow the calculation of set of priors on the underlying interface widgets. This would allow the KEP to identify the maximally likely target within a candidate set using two independent probability distributions – one from command use and one from motion kinematics. Combining these probability distributions could significantly improve endpoint selection. In addition, in our experiment, we enlarged all candidate set targets around the predicted target. However, if underlying priors indicate that certain targets are highly unlikely, we could either not expand those targets, or we could even shrink less likely targets to make additional space available for expansion of more likely targets. This selective expansion would result in less displacement of expanded, likely target, and less overall displacement of unlikely but possible targets. Another strategy is to increase the size of the targets in proportion to the likelihood of the target as determined by the combined probability distributions, so that targets are of varying sizes. This will result in displacement of the user's intended target within the expansion region, but the displacement should be considerably less than expanding every target in the region the same amount.

Beyond target expansion, we also aim to explore additional pointing facilitation strategies. For example, one problem with display expansion is the visual disruption caused by targets moving on the screen. Motor expansion has been shown to improve performance in pointing tasks, and it may be the case that making targets "sticky" in motor space based on either KEP probabilities, or KEP probabilities in conjunction with underlying priors, could have a higher net benefit. One significant benefit of motor space expansion is that visual disruption of the display is entirely eliminated. We continue to explore techniques that effectively incorporate KEP into realistic pointer facilitation techniques for dense and tiled target configurations.

CONCLUSION

In this paper, we examine the effectiveness of expanding multiple candidate targets with a region on a computer display when the entire screen is covered with potential targets. We show that, if a predictor accuracy is sufficiently high (i.e. if a target is expanded with probability greater than 56.5%) and if the expansion region is sufficiently small (less than 80 pixels in our display configuration, or about 4% of the display resolution), then there is a net benefit to expanding targets when using a real-time kinematic endpoint prediction algorithm in conjunction with expanding targets. As distances increase beyond about 800 pixels on our computer display, however, the error associated with our predictor increases, and endpoint prediction is no longer sufficiently accurate to support improved pointing speeds. Although kinematic endpoint prediction may not be, in itself, sufficient to support endpoint expansion for targets on typical desktop computer displays with resolutions of 1920x1200 pixels or greater, we argue that additional information in the form of underlying priors can be used to refine the underlying target probabilities. Generating an accurate set of target probabilities based on priors and motion kinematics should enable novel target expansion strategies, with the potential for significant improvements in user pointing performance in desktop computer displays.

ACKNOWLEDGMENTS

We thank the participants in our study and our colleagues in the HCI lab at the University of Waterloo for valuable suggestions. Funding for this research was provided by the Natural Science and Engineering Council of Canada (NSERC).

REFERENCES

1. R. Balakrishnan. "beating" fitts' law: virtual enhancements for pointing facilitation. *Int. J. Hum.-Comput. Stud.*, 61(6):857–874, 2004.
2. R. Blanch, Y. Guiard, and M. Beaudouin-Lafon. Semantic pointing: improving target acquisition with control-display ratio adaptation. In *CHI '04*, pages 519–526, 2004.
3. A. Cockburn and P. Brock. Human on-line response to visual and motor target expansion. In *Graphics Interface 2006*.
4. P. M. Fitts. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology*, 47:381–391, 1954.
5. T. Grossman and R. Balakrishnan. The bubble cursor: enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proc. CHI '05*, pages 281–290.
6. Y. Guiard, R. Blanch, and M. Beaudouin-Lafon. Object pointing: a complement to bitmap pointing in guis. In *Graphics Interface 2004*.
7. D. V. Keyson. Dynamic cursor gain and tactual feedback in the capture of cursor movements. *Ergonomics*, 40(12):1287–1298, 1997.
8. E. Lank, Y. Cheng, and J. Ruiz. Endpoint prediction using motion kinematics. In *Proc. CHI '07*.
9. I. S. MacKenzie. Fitts' law as a research and design tool in human-computer interaction. *Human-Computer Interaction*, 7:91 – 139, March 1992.
10. M. McGuffin and R. Balakrishnan. Fitts' law and expanding targets: Experimental studies and designs for user interfaces. *ACM Transactions on Computer-Human Interaction*, 12(4):388–422, 2005.
11. J. Ruiz and E. Lank. Effects of target size and distance on kinematic endpoint prediction. Technical Report CS-2009-25, University of Waterloo, 2009.
12. J. O. Wobbrock, E. Cutrell, S. Harada, and I. S. MacKenzie. An error model for pointing based on fitts' law. In *CHI '08*, pages 1613–1622, New York, NY, USA, 2008. ACM.
13. A. Worden, N. Walker, K. Bharat, and S. Hudson. Making computers easier for older adults to use: area cursors and sticky icons. In *Proc. CHI '97*.
14. S. Zhai, S. Conversy, M. Beaudouin-Lafon, and Y. Guiard. Human on-line response to target expansion. In *Proc. CHI '03*.