Fitts' Law and Expanding Targets: Experimental Studies and Designs for User Interfaces

MICHAEL J. MCGUFFIN and RAVIN BALAKRISHNAN
University of Toronto

Recently, there has been renewed interest in techniques for facilitating the selection of user interface widgets or other on-screen targets with a pointing device. We report research into using *target expansion* for facilitating selection. Widgets that expand or grow in response to the user's focus of attention allow for a reduced initial size, which can help optimize screen space use, and may be easier to select than targets that do not expand. However, selection performance could plausibly suffer from a decreased initial widget size. We describe an experiment in which users select a single, isolated target button that expands just before it is selected. Our results show that users benefit from target expansion, even if the target only begins expanding after 90 % of the distance to the target has been travelled. Furthermore, our results suggest that, for sufficiently large ID values, users are able to take approximately full advantage of the expanded target size. For interfaces with *multiple* expanding widgets, however, subtle problems arise due to the collisions or overlap that may occur between adjacent expanding widgets. We give a detailed examination of the issues involved in both untiled and tiled multiple expanding targets, and present various design strategies for improving their performance.

Categories and Subject Descriptors: H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*graphical user interfaces*; *input devices and strategies*; *interaction styles*; *theory and methods*; H.1.2 [**Models and Principles**]: User/Machine Systems—*human factors*; *human information processing*; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*interaction techniques*

General Terms: Human Factors, Experimentation, Measurement, Design, Theory

Additional Key Words and Phrases: empirical evaluation, expanding targets, expansion, Fitts' law, growing targets, interaction design, interaction modeling, target magnification, widget design

1. INTRODUCTION

Several interfaces and interaction techniques have been described [Furnas 1986; Bederson 2000, for example] in which a widget, or portion of a widget, changes size dynamically to accommodate the user's focus of attention. A larger widget or viewing region can provide the user with more information and/or a greater area for input. Widgets that dynamically grow when pointed at (which we will call *expanding widgets*) can now also be found in a popular operating system [Apple Computer, Inc. 2001] where the icons in the desktop toolbar expand when the

Authors' address: Department of Computer Science, University of Toronto, 10 King's College Road, Room 3302, Toronto, Ontario, M5S 3G4, Canada;
mjmcguff@dgp.toronto.edu, ravin@dgp.toronto.edu, http://www.dgp.toronto.edu
Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, the ACM copyright/server notice, the title of the publication, and its date appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists requires prior specific permission and/or a fee.
© 20YY ACM 0000-0000/20YY/0000-0001 $5.00

mouse cursor is over them.

There are potentially two complementary advantages to using expanding widgets. First, widgets that are small when not in use consume less screen space, meaning user interface elements can be packed more densely. Indeed, as software becomes more complex, with an ever increasing number of commands and buttons, an effective strategy may be to display widgets at a significantly reduced size, and expand them to a usable size only when needed. This would allow more screen real estate to be used for displaying data. Second, whatever a widget's initial size, increasing the widget's size when the user points at it may make the widget easier to select, decreasing selection time. Thus, target expansion may also be effective for facilitating selection.

Unfortunately, a target that is initially small may be more difficult to select, even if the target subsequently expands to a larger size. From Fitts' law [Fitts 1954], we know that targets with a smaller fixed size require more time to select. While Fitts' law has been empirically verified in many interaction scenarios [MacKenzie 1992], these have all been for situations where the target has a constant size. It is unclear how performance is affected if the target changes size while the user is moving towards it, as is the case with expanding widgets. Is the selection time governed by the original size of the target, or the final size, or a combination of both? Furthermore, what is the effect of varying the time at which the target begins to expand?

Without answers to these questions, there is little scientific knowledge to guide the design of interfaces that incorporate expanding widgets. In particular, if selection time is determined by the initial target size, the use of expanding widgets is essentially a tradeoff between saving screen space and the ability of users to select these widgets quickly. On the other hand, if the determining factor is the final target size, we may be able to benefit from expanding targets without compromising performance. If the answer lies between these two extremes, but we can accurately predict the tradeoff, this knowledge will allow designers to make informed decisions. In addition to the implications for interface design, these questions are also interesting from a human motor control standpoint.

There are also secondary issues to address when designing interfaces with *multiple* expanding targets. The target that the user wishes to acquire can change from moment to moment. Hence, the interface must continually determine which of the many targets to expand, and when. Also, closely spaced targets may overlap or otherwise interfere with each other during expansion. In this case, should occlusion be allowed, or should some targets be displaced? What is the best performance that can be expected when multiple expanding widgets are packed in a *tiled* arrangement, with no space between targets?

This paper first presents an empirical study involving selection of isolated expanding targets, with the goal of determining how to predictively model performance. Various factors, such as the time at which expansion occurs, are varied to investigate their influence on performance.

The paper then considers theoretical and design issues surrounding interfaces with multiple expanding targets. Target expansion is compared to various other techniques for aiding selection. The findings from the first study are applied to

generate concrete user interface designs involving multiple targets. An additional pilot study is also presented, aimed at determining the maximum performance that might be expected in the extreme case of multiple, tiled, expanding targets.

2. BACKGROUND

2.1 Fitts' Law

Fitts' law [Fitts 1954; Meyer et al. 1990; MacKenzie 1992] describes the time to acquire a target with a rapid, aimed movement. Given the amplitude A of the motion (i.e. the distance to reach the target), and the width W of the target measured along the axis of motion, the average movement time MT required is

$$MT = a + b \log_2 \left(\frac{A}{W} + K \right) \tag{1}$$

where a and b are empirically determined constants. The logarithm is referred to as the index of difficulty $ID = \log_2(A/W + K)$ which is in bits. K is a constant whose value has been proposed to be zero, 1/2, or 1. $K = 1$ yields the now popular Shannon formulation of Fitts' law, which has the advantages that ID is always non-negative, and has been shown to better fit measured data [MacKenzie 1989; 1992].

The constants a and b vary with factors such as the pointing device and muscles used for input (e.g. mouse, stylus, trackball, gaze tracker), the control-display C:D ratio (i.e. the ratio of distance moved by the physical limb, and the distance moved on the screen by the virtual cursor), and the population of users (e.g. children, adults). To determine a and b for a given set of such factors, typically an experiment is performed where users perform many selections while A and W are varied. Fitting a straight line to the measured MT values yields a and b as the intercept and slope of the line. Once a and b are known, Fitts' law enables prediction of performance in future selections, so long as the factors that influence a and b do not change.

Aimed, rapid movements involve a speed/accuracy tradeoff, and this is reflected in MT being an increasing function of the ratio A/W in Equation 1. Thus, if the distance A to a target is increased by some factor, the selection can still be performed in the same time (i.e. with greater speed) if the accuracy requirement is reduced (i.e. W is increased) by the same factor. Also observe that, if the target's size W is reduced by a factor of 1/2, this results in an increase of approximately 1 bit in ID, and a corresponding incremental increase of MT.

One way to interpret Equation 1, and the task it models, is that the user must home in on the target by reducing the initially large noise (or statistical uncertainty) in the cursor's position until the noise is small enough that the cursor falls within the target. Every reduction by 1/2 of this noise requires b seconds, and corresponds to transmitting 1 bit of information. The number of such reductions necessary is $\log_2(A/W) \approx ID$, and transmitting ID bits requires bID seconds in total. The remaining a seconds in Equation 1 can partly be explained as reaction time and/or the time necessary to complete the selection with a button press. Much more detailed and accurate explanations of the mechanisms behind Fitts' law have been developed [Meyer et al. 1990], however the foregoing is a useful first approximation.

As a final remark, Fitts' law enables the characterization of a given input device

by an index of performance IP (also called throughput), which is the number of bits/second the device allows the user to transmit, independent of the particular target involved. One common convention used for computing throughput is $IP = 1/b$, though this ignores the constant cost of a [Zhai 2002].

2.2 Lower-Level Models of Motor Control

To hypothesize about user performance with expanding targets, it is helpful to consider some of the lower-level motor control theory behind Fitts' law.

Examination of kinematic data for individual target acquisitions reveals that the movement of the user is often not a single, smooth motion, but rather is composed of a sequence of one or more submovements (Figure 1). The first submovement is typically large and fast, covering most of the distance to the target. This may be followed by subsequent, smaller and slower movements, that correct for any undershoot or overshoot of the initial movement.

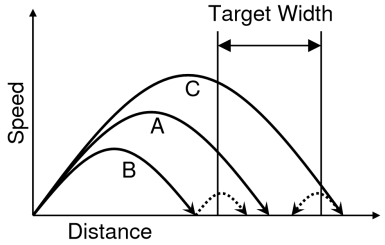

Fig. 1. Possible sequences of submovements toward a target. *A:* A single movement reaches the target. *B and C:* The initial movement undershoots or overshoots the target, requiring subsequent corrective movements (dotted curves).

Perhaps the simplest motor control model proposed to explain Fitts' law is the *deterministic iterative-corrections model* [Crossman and Goodeve 1983; Keele 1968], which postulates that the submovements each have equal duration, each travel a constant fraction of the remaining distance toward the target, and are all executed under *closed-loop* feedback control, i.e. visual or kinesthetic sensory feedback. This model allows Fitts' law to be mathematically derived, however there have also been several serious flaws found with the model [Meyer et al. 1990, pp. 194–195].

An alternative set of theories postulate two phases involved in target acquisition: an initial, open-loop, ballistic impulse; followed by a corrective, closed-loop, "current control" phase. This division into two phases can be traced back to Woodworth [Woodworth 1899, p. 41], however there are more recent and sophisticated versions of it, such as the *stochastic optimized-submovement model* [Meyer et al. 1988], from which it is again possible to derive Fitts' law, as a high-level approximation.

One element common to all these theories is that the later, corrective submovements are performed under closed-loop control. If users make use of *visual* sensory-feedback during these later submovements, it is plausible that users would be able to take advantage of target expansion, even when the expansion occurs late in the execution of the task. There are, however, two limiting factors to consider. First,

if the ID of the task is very low, the user may often be able to acquire the unexpanded target with a single submovement that is planned and executed according to the unexpanded target size. In this case, users might not have any opportunity to take advantage of the expanded target size. Thus, target expansion should probably yield greater benefit for larger ID values, where corrective submovements are more likely to be necessary. Second, the human closed-loop reaction time is roughly 100–200 ms [Zhai et al. 2003], which limits how late expansion can occur for the user to be able to benefit from it.

Within these limiting factors, however, we expect performance of users to be enhanced by target expansion, and the concrete effect of these limits should be revealed through collection of experimental data.

3. EXPERIMENT WITH ISOLATED EXPANDING TARGETS

The models of motor control just discussed predict that the corrective submovements performed toward the end of a target acquisition are done with closed-loop feedback control. Our main hypothesis for expanding targets, then, is that, when corrective submovements are necessary (that is, when ID is large), target acquisition time should be dependent on the *final* target size and not the *initial* one at the onset of movement. Additional questions to be addressed in our experimental investigation are: Can performance with expanding targets be modelled by Fitts' law? Do different methods for target expansion affect performance? At what point should the target begin expanding?

This last question, of when to begin expanding, calls for particular attention. A conservative strategy would be to expand the target sometime during the execution of the initial submovement, and have it completely expanded before the user plans and executes the corrective submovements. From an interface design standpoint, however, it is preferable to delay expansion as much as possible — this would allow for widgets to remain small and not obscure other elements of the display until absolutely necessary. Expanding too late, however, precludes any possible reduction in selection time. Thus, it is important to determine how late the target may expand while still realizing a significant performance advantage.

3.1 Apparatus

The experiment was conducted on a graphics accelerated workstation running Linux, with a 21-inch, 1280×1024 pixel, colour display. A puck on a Wacom Intuos 12×18 inch digitizing tablet was used as the input device. The puck was used to drive the system cursor, and worked in absolute mode on the tablet with a constant linear 1:1 control-display ratio.

Although mice are the most commonly used pointing device by users, we used a tablet in our experiment because the data collected from an absolute positioning device corresponds more directly and reliably to physical limb movements, allowing for potentially more kinds of analysis. Furthermore, the puck used with the tablet had the approximate size and shape of a standard mouse, hence we did not expect users to require a significant amount of time to adjust to using a puck. Users also practiced using the puck and tablet during warmup periods.

3.2 Task and Stimuli

A discrete target selection task was studied where the target's width expands dynamically at some time after the onset of movement. At the start of each trial, a small start box appeared on the left of the screen (Figure 2). Participants had to move the cursor into this box and dwell there for 1 second, at which point a rectangular target appeared on the right of the screen. Participants then had to move the cursor as quickly and accurately as possible onto the target, and indicate completion by clicking the puck button. The target covered the entire vertical extent of the screen, so that pointing only required one dimensional motion along the horizontal axis. Timing began when the target appeared, and ended when the target was successfully selected.

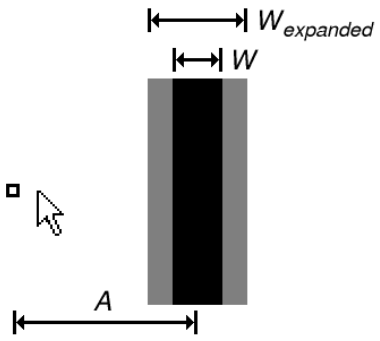

Fig. 2. Stimuli. Users moved from the start box on the left to the target on the right. In the static condition, the target had a width of W. In the expanding condition, the target began with a width W, but expanded to $W_{expanded}$ when the cursor moved past a specified fraction P (the *expansion point*) of the total distance A, where $0 \leq P \leq 1$. The amplitude A was measured from the centre of the start box to the centre of the target.

If the user's first click did not successfully select the target, the occurrence and time of the error were recorded, but the trial was not terminated early. Instead, users were forced to complete each trial successfully, and the time assigned to a trial for the purpose of our analysis was the total time to successfully complete it. A more traditional Fitts experiment would have trials terminate as soon as an error occurs, however this has the potential side-effect that users may be tempted to execute trials faster, even at the risk of committing more errors, due to impatience and the knowledge that errors allow trials to be completed faster. This would confound the error rate and movement times. Indeed, an often reported effect in Fitts' law studies is that error rate increases with ID, and this may partly be due to increased impatience (even if only subconscious) on the part of the user. In contrast, with our design, there is no incentive for the user to commit errors, since every trial must be successfully completed. Our design also has the advantage that the movement times collected could still be analyzed in a traditional manner, e.g. by removing error trials, or by using the time of the first error as the movement time for each error trial. In what follows, however, our analysis uses the total movement times of both non-error and error trials. Thus, in our analysis, the total movement time per

trial incorporates the cost of errors as the time to correct for target misses. We feel this better reflects the real-world scenario where users must spend time correcting for any errors in selection.

In all cases, the expanded target width was twice the initial target width, i.e. $W_{expanded} = 2W$. While we could have varied this parameter, we felt that an expansion factor of 2 was representative of what could be expected in real interface widget design, and was sufficient to address the main questions driving the study.

3.3 Pilot Study

A pilot study was conducted to determine if our direction of research was promising, and also to test which experimental conditions would have significant effects on performance.

In the pilot study, one condition had users select static targets that did not expand at all, to serve as a base case for comparison. The other conditions involved expanding targets, and tested two different methods for expansion: *spatial expansion*, where the widget grows gradually in size over a short period, and *fading-in expansion*, where the target's size is expanded instantly (possibly affording the user a more immediate advantage), but the visual representation of the expanded target is faded in gradually (to avoid a jarring visual effect) using alpha blending.

Furthermore, for each of the expansion methods, three different values for the expansion point P were tested: 0.25, 0.5, and 0.75, corresponding respectively to distances of 25 %, 50 %, and 75 % of A measured from the starting point.

The results of the pilot study showed that the movement times for the expanding conditions were significantly smaller than for the static condition. Regression analyses showed that the data for each condition fit a Fitts' law equation with r^2 values above 0.97, implying that expanding targets can be modelled with some form of Fitts' law. Interestingly, neither the method of expansion, nor the expansion point P had any significant effect on movement time. This last result implies that target expansion can occur as late as 75 % of the way to the target and still result in performance as good as if the target had expanded much earlier.

A more detailed description of our pilot study can be found in McGuffin and Balakrishnan [2002].

3.4 Full Study

3.4.1 *Participants.* Twelve volunteers (9 male, 3 female) participated, aged approximately between 20 and 35 years. All were right-handed and had years of experience with computer mice. Although they had little or no experience with a tablet-based puck, they appeared able to use one as a pointing device without problems.

3.4.2 *Design.* Given that the results of the pilot study showed no difference in performance between the two expansion strategies, we decided to only use the *spatial* expansion method for our full experiment. This was chosen as the preferred technique since it is arguably more consistent with real interfaces, and avoids the visual interference of alpha blending two images as with the *fading-in* method.

Thus, we have two main conditions, *static* and *expanding*.

Similarly, since our pilot results showed no effect on performance when the ex-

pansion point P was changed, we used a single value for P in the full study. We also wanted to demonstrate how late expansion could be performed. A second, smaller pilot study suggested that $P = 0.9$ still afforded the same advantages of expansion; hence this was the value used in the full study.

Targets were made to expand over a 100 millisecond interval. This gives the appearance of a smooth change in target size, but was found to be fast enough to give the user time to react to the expanded target.

For both of the main conditions, in units of 16 pixels, we used four target widths (W = 0.5, 1, 2, and 4 units), and four target amplitudes (A = 8, 16, 32, and 64 units). Fully crossed, these result in sixteen A,W combinations. However, since $P = 0.9$, having conditions where the target initially covers more than 10 % of the amplitude would mean that the user would already be in the unexpanded target before it begins to expand, thus gaining no advantage from the expansion. Accordingly, for both of the main conditions, we eliminated the three easiest A,W conditions (A,W = 8,2; 8,4; 16,4) from the full set of sixteen. We thus have thirteen A,W combinations (8,0.5; 8,1; 16,0.5; 16,1; 16,2; 32,0.5; 32,1; 32,2; 32,4; 64,0.5; 64,1; 64,2; 64,4 in units of 16 pixels) with five levels of task difficulty (ID) ranging from 3.17 to 7.01 bits. We feel this covers a range of IDs relevant to real user interface design. An ID of 7 bits is near the upper limit of what users typically encounter, corresponding to an 8 pixel target at a distance of over 1000 pixels.

The two main conditions were counter balanced between the participants: six users did the static condition first followed by the expanding condition, while the other six did the opposite. The thirteen A,W conditions within each main condition were within-subjects. A repeated measures within-subjects design was used for each condition — participants were presented with five blocks, each consisting of all thirteen A,W combinations appearing five times each in random order within the block. In summary, the experiment consisted of

12 participants
× 2 conditions per participant
× 5 blocks per condition
× 13 A,W combinations per block
× 5 trials per A,W combination
= 7800 trials in total

At the start of the experiment, for each of the two conditions, participants were given a warmup block of trials consisting of a single trial for each A,W condition, to familiarize them with the task and conditions. Data from these warmup trials were not used in our analysis. The experiment was conducted in one sitting and lasted about 50 minutes per participant. Participants were allowed to rest between blocks of trials.

3.4.3 *Hypotheses.* We propose the following hypotheses as plausible and appropriate to test with our experiment:

H1. The expanding condition will result in faster movement times than the static condition.
H2. Performance in both conditions can be accounted for by Fitts' law.

H3. Performance in the expanding condition is dependent largely on the target's final size, not its initial one.
H4. Performance in the expanding condition can be predicted based on the Fitts' law equation generated in the base static condition.

3.4.4 *Results and Discussion.* Repeated measures analysis of variance showed a significant main effect for condition ($F_{1,11} = 1345$, $p < 0.0001$). The overall mean movement times were 1.335 seconds for the static condition and 1.178 seconds for the expanding condition. These results clearly indicate that expanding targets can result in improved performance, confirming hypothesis H1. Figure 3 illustrates.

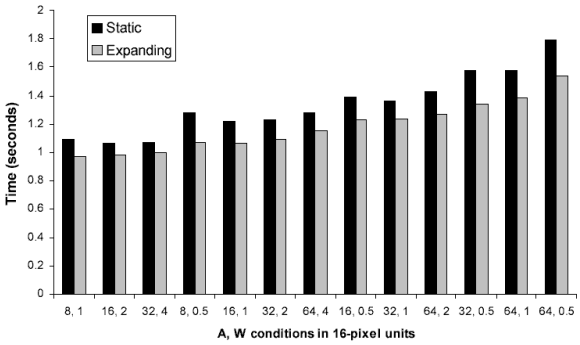

Fig. 3. Comparison of average movement times for static and expanding conditions for each A,W condition tested.

Within each of the two conditions, the movement times were aggregated into 13 average movement times (one for each A,W combination). Linear regression revealed that, within each condition, these average movement times fit a Fitts' law equation with r^2 values above 0.97 (Figure 4). Thus, hypothesis H2 is confirmed.

Given the a and b constants used to fit the data in the static condition, we can estimate a lower bound on movement time in the expanding condition. To acquire an expanding target, the user should take at least as much time as they would to acquire a target that is always expanded:

$$MT \geq a + bID_{expanded} \tag{2}$$

where

$$ID_{expanded} = \log_2\left(\frac{A}{W_{expanded}} + 1\right) = \log_2\left(\frac{A}{2W} + 1\right) \tag{3}$$

and the initial ID of the target is

$$ID = \log_2\left(\frac{A}{W} + 1\right) \tag{4}$$

Solving Equations 3 and 4, we can find $ID_{expanded}$ in terms of ID and substitute into Equation 2, yielding

$$MT \geq a + b\left(\log_2\left(2^{ID} + 1\right) - 1\right) \tag{5}$$

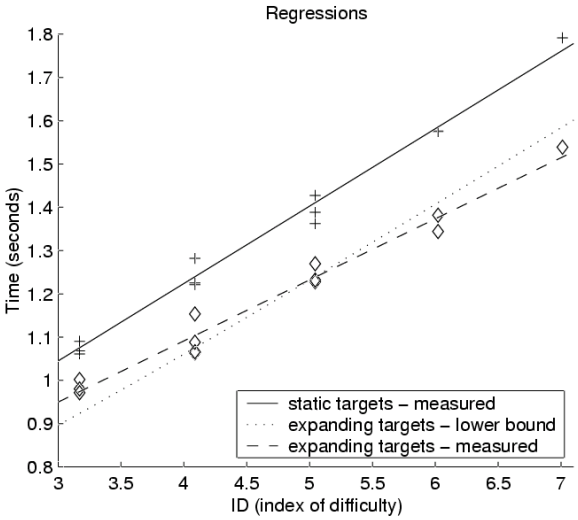

Fig. 4. The solid and dashed lines were obtained via regression of the measured data for both conditions, and correspond to equations $MT = 0.50833 + 0.17888ID$ and $MT = 0.52665 + 0.14109ID$ for static and expanding targets, respectively. The dotted curve is a theoretical lower bound on movement time with expanding targets, corresponding to equation $MT = 0.50833 + 0.17888(\log_2(2^{ID} + 1) - 1)$ which is derived from the regression of the measured data for static targets.

This bound is plotted in Figure 4, and appears to be close to the data measured for the expanding condition, which is a sign of support for hypotheses H3 and H4. The data points and values along the lower bound in Figure 4 are given numerically in the below table.

A,W (16-pixel units)	ID (bits)	Measured Time (seconds)		Expanding Lower Bound (seconds)
		Static	Expanding	
8,1	3.17	1.090	0.971	0.924
16,2	3.17	1.061	0.980	0.924
32,4	3.17	1.068	1.002	0.924
8,0.5	4.09	1.282	1.066	1.075
16,1	4.09	1.221	1.064	1.075
32,2	4.09	1.225	1.088	1.075
64,4	4.09	1.281	1.153	1.075
16,0.5	5.04	1.388	1.228	1.240
32,1	5.04	1.362	1.232	1.240
64,2	5.04	1.427	1.269	1.240
32,0.5	6.02	1.576	1.344	1.411
64,1	6.02	1.575	1.382	1.411
64,0.5	7.01	1.792	1.539	1.586

There was a significant $ID \times$ condition interaction ($F_{4,11} = 30$, $p < 0.0001$), indicating that the performance gains due to target expansion varied depending on the value of ID. To examine this in more detail, and to further test hypotheses

H1, H3, and H4, two sets of t-tests were performed on the data for each ID value. In the first set, for each ID value, the unaggregated movement times for the static condition were compared with those for the expanding condition. This revealed that, for each ID value, the movement times in the expanding condition were significantly faster ($F > 100$ and $p < 0.0001$ in all cases) than those in the static condition, further confirming hypothesis H1.

In the second set of t-tests, we borrowed a technique used by Zhai et al. [2003] and compared movement times according to ID_{final}, i.e. the final ID of the target. For static targets, $ID_{final} = ID$, however for expanding targets, $ID_{final} = ID_{expanded}$. For example, we compared the times in the static condition where $ID = 3.17$ to the times in the expanding condition where $ID = 4.09$, since in both cases, $ID_{final} = 3.17$. In full, for each ID_{final} value ranging from 3.17 to 6.02, a t-test compared unaggregated movement times for the static condition with those for the expanding condition. If users gain the full benefit of expanding targets, we should expect the times within each ID_{final} for expanding targets to not be significantly slower than the times for static targets. The following table summarizes the results:

ID_{final} (bits)	Result of t-test
3.17	expanding targets significantly slower ($F = 6.0, p \approx 0.015$)
4.09	no significant difference ($p \approx 0.36$)
5.04	expanding targets significantly faster ($F = 4.8, p \approx 0.03$)
6.02	no significant difference ($p \approx 0.07$)

These results indicate that, for $ID_{final} \geq 4.09$, the user seems to have benefited fully from target expansion, and in the case of $ID_{final} = 5.04$, performance even *exceeded* the best which could be expected given our rationale for the lower bound on movement time. Although one might expect full benefit from expansion for small values of the expansion point P, where the user has greater time to react to expansion, our data was collected with $P = 0.9$. Strictly speaking, these results only partially support hypotheses H3 and H4. However, we note that the significant difference for $ID_{final} = 3.17$ can be explained by the fact that lower ID values tend to require fewer corrective submovements, thus, as noted in §2.2, we should expect the benefit from expansion to be less at lower ID values. We conjecture that future studies that collect more data, possibly extending the range of IDs further, would find that movement time in the expanding condition tends toward the lower bound in Figure 4, especially for larger ID values. We suspect that the apparent dip in Figure 4 of data below this bound, and the significantly *lower* times for $ID_{final} = 5.04$, are *not* indicative of a real trend below the bound, but are more likely due to noise.

Hypothesis H4 is not strictly confirmed by our data, however Figure 4 appears to show a rough match between the lower bound and the measured data for expanding targets. We thus propose the lower bound as a useful estimate of performance with expanding targets.

After the initial publication of our results [McGuffin and Balakrishnan 2002], Zhai et al.'s [2003] follow-up study collected additional data which appears to suggest that, as ID_{final} increases, the movement time with expanding targets approaches that of static targets [Zhai et al. 2003, Figures 5, 9], which would again imply that

users benefit as much as could be expected from expansion. However, Zhai et al. did not come to a definite conclusion as to whether this apparent trend was supported by statistical analysis.

Figure 5 shows most of the same data as in Figure 3, but with data grouped by W_{final}, to make apparent the similar average movement times for both conditions.

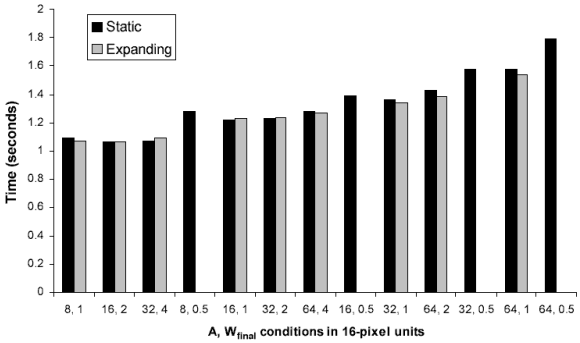

Fig. 5. Same as Figure 3, but with bars arranged according to W_{final} instead of W. For static targets, W and W_{final} are the same, but for expanding targets, W_{final} is equal to $W_{expanded} = 2W$. Not all bars could be paired together (and four of the times for expanding targets are not shown), but for those shown, the movement times appear similar within each pair.

In our experiment, because the static/expanding conditions were not mixed within blocks, participants always knew whether the target they were about to select would expand or not. It is possible that knowing a priori whether or not the target would expand could result in the user planning ahead for the expansion, rather than reacting to the expansion during the motion. However, our experimental design is reflective of how a real system would work: a user familiar with a graphical interface knows which, if any, of the widgets expand prior to selection. Furthermore, our design is intended to create optimal conditions for observing an effect with expanding targets, which is the conservative course of action for a first study, since we wish to test if expansion can improve performance at all. Zhai et al.'s [2003] follow-up study further investigated this issue by testing participants who did not always know a priori whether expansion would occur or not. They found that performance was still enhanced when users did not know if expansion would occur or not, implying that users were adapting in response to visual feedback, rather than simply planning their submovements in anticipation of target expansion.

Following Zhai et al.'s [2003] example, we computed some simple kinematic data to check how much time users had to react to expansion. The average normalized time at which 90 % of distance was covered by the user was 56.6 % (standard deviation 11.2) in the static condition, and 61.1 % (standard deviation 11.3) in the expanding condition. Thus, similar to Zhai et al.'s [2003] finding, although users only had the last 10 % of the distance in which to react to target expansion, they also had approximately 39 % of the total movement time left, so it is not so surprising that performance was enhanced.

We also performed some elementary error analysis of our data. Of the 3900 trials in the static condition, 135 involved an error (i.e. the user's first click did not fall on the target). Within these, the average correction time (i.e. fraction of total trial time after the first click) was 36.8 %. Similarly, of the 3900 trials in the expanding condition, 159 involved errors, within which the average correction time was 38.3 %.

This table shows error rates by ID:

ID	Error Rate	
(bits)	Static	Expanding
3.17	2.89 %	1.67 %
4.09	3.42 %	3.17 %
5.04	3.67 %	6.89 %
6.02	4.00 %	5.17 %
7.01	3.67 %	4.33 %

Unlike Zhai et al. [2003], we do not observe a clearly increasing error rate with ID. This may partly be due to the design of our experiment, which forced successfully completion of all trials, even after an error. As explained in §3.2, we feel that a more traditional design, which does not force successful completion of every trial, may artificially encourage a higher error rate at higher ID values.

The only other significant effect was a learning effect across the blocks of trials ($F_{4,11} = 16$, $p < 0.0001$), which is not unusual in these experimental tasks.

3.5 Summary of Findings

Our results indicate that (1) performance is significantly enhanced by expanding targets, even when expansion occurs after 90 % of the distance towards the target has been traversed, (2) the task of acquiring an isolated expanding target can be accurately modelled by Fitts' law, (3) for sufficiently high ID values, performance is approximately as good as, or better than, the best that could be expected, given our rationale for the lower bound on movement time. This last point means that users benefited fully from expansion for sufficiently high ID, suggesting that the final expanded target size is much more important for determining performance than the initial target size.

4. MULTIPLE EXPANDING TARGETS

Our experimental results may have significant implications for interface design, in particular for the design of buttons, menus, or other selectable widgets. Clearly, an isolated widget that expands to a larger size should be easier for the user to click on. However, when there are many such widgets on the screen, they may collide or overlap during expansion, mutually interfering with each other.

In this section, we classify the different potential designs for multiple expanding targets, identify their pros and cons, and compare them with other techniques for facilitating selection. We then give further details of designs we have developed (and in some cases prototyped in software), and describe a mathematical model and a pilot experiment investigating the most ambitious class of multiple expanding targets.

4.1 Basic Observations

When discussing multiple expanding targets, and other selection facilitation techniques, it is useful to distinguish between *motor space* and *visual space* [McGuffin and Balakrishnan 2002; McGuffin 2002; Zhai et al. 2003; Blanch et al. 2004]. Motor space is the set of all possible positions of the pointing device, or the physical space that the user's limb moves through. Visual space is where visual feedback is displayed, e.g. the set of pixels on a raster display. In a system with a fixed C:D ratio, there is a fixed, linear mapping from the input device's position in motor space to the cursor position in visual space (ignoring translations that occur with relative devices, such as when a mouse is lifted up and repositioned on a desk). In such a system, if a target widget in visual space does not move or change size, its area in visual space corresponds directly to the region in motor space that the user must enter to acquire the target. The distinction between motor and visual space becomes important if, for example, targets in visual space move or change size in response to cursor motion (e.g. as with expanding targets), or if the cursor jumps discontinuously to new positions (e.g. object pointing [Guiard et al. 2004]) or moves in an otherwise non-linear fashion (e.g. semantic pointing [Blanch et al. 2004]).

In our experiment with isolated expanding targets, expansion occurred dynamically in visual space, changing the button's width from W to $W_{expanded} = 2W$. In motor space, however, the target had a fixed expanded size of $2W$; there was no dynamic change in motor space. This is because the set of points in motor space which mapped to the button remained fixed: as soon as the pointing device fell on any of these points, the visual expansion was invoked, and the expanded size was available to the user.

With multiple expanding targets, the targets may similarly have a fixed footprint in motor space (Figure 6, A and B). In fact, if the expansion of targets depends only on the *current position* of the pointing device, then the mapping from motor space to targets must be a static mapping [McGuffin 2002; Zhai et al. 2003]. A static mapping does not present a major problem for untiled targets (Figure 6 A), since the space between targets allows for an expanded size in both visual and motor space. However, if targets are tiled, a static mapping implies there is no room left in motor space for an expanded target size (Figure 6 B).

In Figure 6 B, we have targets that *look* expanded when the cursor is over them, however the expanded size of a button is not fully available to the pointing device. As the user moves away from the centre of an expanded button, the button must contract back to its original size before the cursor reaches the edge of the button's fully expanded area. In motor space, the buttons are no easier to select than non-expanding buttons would be.

For tiled targets, the only way to achieve expanded target size in motor space is to have a mapping that changes dynamically (Figure 6 C), i.e. where expansion depends not only on current cursor position. If the system can somehow anticipate which target is of interest to the user, perhaps by using information in the cursor trajectory [McGuffin 2002; Zhai et al. 2003], then the system could expand a target in both visual and motor space for a brief time, to aide the final stage of the user's movement.

To summarize, we have identified three basic schemes for multiple expanding

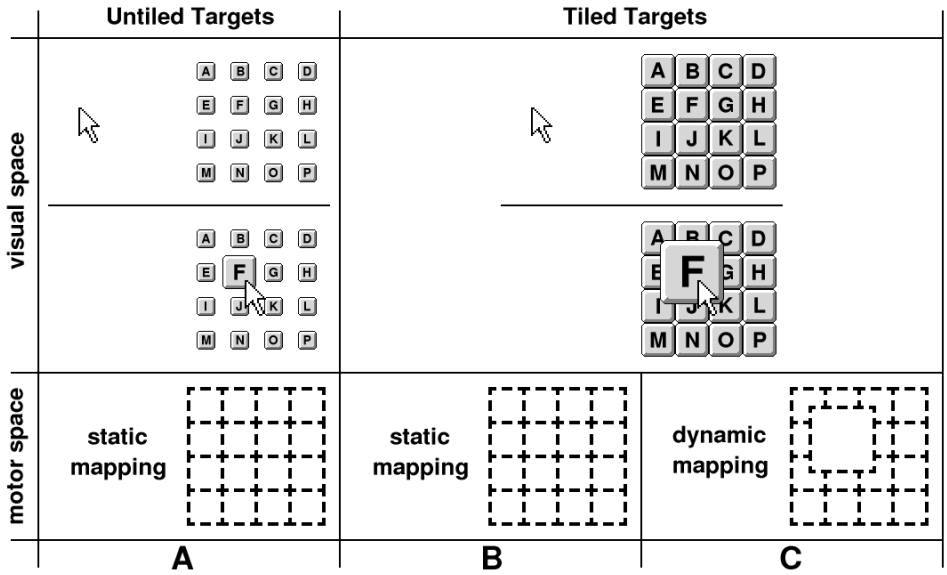

Fig. 6. Three possibilities, *A*, *B*, *C*, for the behaviour of multiple expanding targets. In visual space, we see the state of the targets before and after expansion. The bottom row shows the mapping from motor space to buttons; this mapping may be static, or may change dynamically over time. *A:* Targets are not tiled. The expansion in visual space partially fills the empty space between buttons. In motor space, the footprints of buttons are static and correspond to the expanded button size; at best these footprints tile motor space (as shown). *B and C:* Targets tile the visual space. If expansion depends only on the current cursor position, then the mapping from motor space to buttons is static (*B*), and the footprints of buttons in motor space can be no larger than the unexpanded buttons in visual space. Thus, in *B*, the target is expanded in visual space but not in motor space. Expansion in motor space with tiled targets is only possible if the mapping from motor space to buttons changes dynamically (*C*), e.g. as a function of the button predicted to be desired by the user, perhaps based on cursor trajectory.

targets: **untiled targets** (*A*), **tiled targets without motor expansion** (*B*), and **tiled targets with motor expansion** (*C*).

All three allow widgets to be made smaller (and in the case of *B* and *C*, denser), freeing up space for the display of other data while still allowing widgets to be subsequently magnified in visual space for browsing.

A has the disadvantage that the space freed up, which is in between the buttons, may only be used for output: although data may be displayed between the buttons, the user cannot click on such data (unless the user can somehow deactivate the expansion, perhaps with a special button), because moving the cursor over it causes the nearest widget to expand and occlude the data.

B has the disadvantage of lacking expansion in motor space, meaning that targets are probably no easier to select than non-expanding targets would be. Expansion would of course still help with browsing and recognition of buttons. There is also a possibility that the expansion in visual space alone might help in rapid, aimed targeting tasks, e.g. by making it easier to see when the user is over their desired target, however this is unlikely given previous negative results with visual targeting

feedback [Akamatsu et al. 1995].

C seems to have the combined advantages of allowing for denser controls that are also no harder to select, without the disadvantages of A or B. Unfortunately, it remains to be demonstrated whether C can be successfully implemented. A good prediction algorithm might make C practical, however at the moment it is the most challenging design possibility for expanding targets.

In the following sections, we relate these three schemes to other techniques for facilitating selection, and then discuss each of the three schemes in more detail.

4.2 Relationship with Other Selection Facilitation Techniques

Fitts' law (Equation 1) suggests two non-exclusive ways [Guiard et al. 2004] of making targets easier to select: by decreasing the distance A between the cursor and target (e.g. by moving one toward the other), or by increasing the size W of the target (which might be done indirectly by increasing the cursor's tolerance, as with area cursors). The below table lists various techniques for facilitating selection, broken down by the high-level strategy they employ to reduce A or increase W.

High-Level Strategy	Selection Technique	Type of Hysteresis	Eases selection of tiled targets?
move cursor closer to target (or ease movement toward target)	snap-to cursor [Sutherland 1963] [Feiner et al. 1981] [Bier and Stone 1986]	none	no
	object pointing [Guiard et al. 2004]	cursor	no
	flick gesture [Dulberg et al. 1999]	widget	no
	C:D ratio adaptation [Keyson 1997] [Worden et al. 1997] [Blanch et al. 2004]	cursor or possibly none	possibly
	haptic feedback [Münch and Dillmann 1997] [Oakley et al. 2001] [Oakley et al. 2002]	none	possibly
move target closer to cursor	drag-and-pick [Baudisch et al. 2003]	widget	no
make cursor bigger	area cursor [Kabbash and Buxton 1995] [Hoffmann 1995] [Worden et al. 1997]	none	no
	bubble cursor [Grossman and Balakrishnan 2005]	none	no
make target bigger	untiled expanding targets	none	no
	tiled expanding targets without motor expansion	none	no
	tiled expanding targets with motor expansion	widget	possibly

In the table, "snap-to cursor" refers to a technique where the cursor can temporarily appear to be displaced onto a nearby target, but without truly warping

the cursor. Also, the "bubble cursor" [Grossman and Balakrishnan 2005] is an improved area cursor that automatically expands or shrinks so that exactly one target is contained in the cursor at all times.

All of the techniques listed facilitate selection of a single, isolated target. Many of them can also improve performance with multiple untiled targets, by exploiting the normally unused regions in motor space to make targets effectively larger or closer together.

One major group of techniques to consider are those that create a static mapping from motor space to targets (e.g. snap-to cursors that snap to the nearest target, area cursors, bubble cursors, and untiled expanding targets). With such techniques, at best motor space may be tiled by targets, i.e. their footprints can at best completely cover the available motor space. When motor space is statically tiled in this manner, the regions in visual space that fall between targets may be used to display other data, however this data is for output only and cannot be selected with the pointing device. The major difference between techniques in this category is in their visual feedback, which must indicate to the user which target would be selected from the current cursor position. Possibilities for this visual feedback include: highlighting the nearest target; connecting the cursor to the nearest target via a rubber band; expanding the nearest target so it lies under the cursor (i.e. untiled expanding targets); expanding the cursor so it contains the nearest target (i.e. bubble cursors); drawing the cursor at a temporarily offset position so it is over the nearest target (i.e. snap-to cursors). Although the techniques in this category all have some advantages over status quo pointing, they all create a static mapping, which cannot ease selection of tiled targets. Ideally, we would like to overcome this limitation, e.g. by achieving target expansion in motor space.

Some of the other techniques listed in the table do not statically map motor space to targets, however this does not necessarily give them an advantage over static mappings. The lack of a static mapping is equivalent to possessing a kind of hysteresis (or memory) that makes the mapping dynamic.

The notion of hysteresis has been used before to describe user interfaces [Shoemake 1992], and we define it as follows. A user interface exhibits hysteresis if it is possible for the user to move the pointing device from a point p, though a closed curve back to p, and end up in a different state or configuration than the system was initially in. We distinguish between two kinds of hysteresis: *cursor hysteresis* (Figure 7), where travelling through a closed curve with the pointing device may result in the cursor having a different position in visual space (i.e. due to warping), and *widget hysteresis*, where travelling through a closed curve with the pointing device may result in the widget(s) having a different size or position, in visual space and/or motor space. The kind of hysteresis, if any, associated with each selection technique is listed in the table.

Selection techniques with cursor hysteresis, i.e. that may warp the cursor position, have the disadvantage that they are not suited for direct input devices (e.g. touchscreen, or a tablet PC with a stylus), or with devices used in absolute mode (e.g. stylus on a tablet).

The flick gesture and drag-and-pick techniques are special in that they involve dragging from one location to another, rather than pointing at a single location.

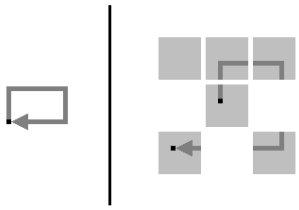

Fig. 7. An example of cursor hysteresis, in this case in the object pointing technique [Guiard et al. 2004]. *Left:* motor space. The pointing device moves from the black dot, through a closed path, back to the same black dot. *Right:* visual space. The cursor skips over empty space between the grey targets, and does not end at the same point it started at.

The table lists them as having widget hysteresis because, after a drag has started, the target activated at a given cursor location (and hence the target's footprint in motor space) depends on the direction from which the cursor was dragged.

The last column of the table indicates which techniques could possibly aid selection of tiled targets. Most techniques cannot, either because they involve a static mapping and/or reduce to status quo pointing in the case of tiled targets. Of the remaining techniques, none have been conclusively shown to improve pointing performance with tiled targets. For example, we don't find it inconceivable that haptic feedback or C:D ratio adaptation, if appropriately designed, might someday help targeting of tiled targets, however to date this has not been demonstrated, and authors have instead reported that these techniques can cause problems if intermediate distractor targets lie along the path to a desired target [Münch and Dillmann 1997; Oakley et al. 2001; Oakley et al. 2002; Blanch et al. 2004].

We suspect that the only way to ease selection of tiled targets is for the system to try to predict the desired target of the user, using more than, for example, just the current cursor position. Prediction might be based on a real-time extrapolation of the cursor's current trajectory, and/or on the frequency of recent selections. The estimation provided by prediction could be used to dynamically change the mapping from motor space to targets, to make the desired target easier to select. This could be used with expanding targets to decide which of a set of tiled targets to enlarge in motor space. It could also be used, for example, to improve haptic feedback: haptic feedback need only be turned on for the target predicted to be desired by the user. In fact, good prediction could be used to enhance any of the techniques listed in the table, and thereby ease selection of tiled targets. (In cases where the last column of the table indicates that techniques *cannot* ease selection of tiled targets, this is based on current descriptions of the techniques in the literature, which in most cases do not involve the kind of prediction we consider here.)

Unfortunately, automatic prediction will not always be correct. There will be a cost associated with mistakes made by the system, e.g. if the system expands the wrong target, making a neighbouring target, which happens to be the one actually desired by the user, more difficult to select. It is plausible that this cost will cancel out or even outweigh the benefit when the prediction is correct.

Given an implemented technique that uses prediction, we could measure the net

benefit in performance for such a system by running a controlled experiment. Although this would give an indication of success or failure, it might also leave many questions unanswered, such as: how to improve the design, how to optimize the design, or (in the case of a negative result) if a successful design is even possible. Ideally, a model of the benefits and costs involved should guide designs and complement experimental evaluation. As will be seen, we have developed a quantitative model of the benefits and costs with tiled expanding targets with motor expansion, which yields an upper bound on the expected net benefit of the technique.

In this section, we have briefly surveyed techniques for aiding selection, paying particular attention to their applicability to tiled targets. Tiled targets are an ultimate challenge for research in selection facilitation; improving performance with them would effectively allow the user to exceed the normal index of performance IP of Fitts' law. However, even selection techniques that cannot aide targeting with tiled targets (Figure 6, A and B) have other benefits, and are also simpler and less risky to use. In the following sections, before we present our model and designs for tiled expanding targets with motor expansion, we first revisit untiled expanding targets, and tiled expanding targets without motor expansion, and consider design possibilities for each.

4.3 Untiled Expanding Targets

Our experimental results indicate that, with untiled targets, simply expanding widgets that are near the cursor should significantly facilitate selection. No sophisticated prediction is necessary, and because expansion need only occur in proximity to the cursor, e.g. within 10 % of A, the user is less likely to be distracted by multiple expanding targets on screen.

Figure 6 A shows a concrete example of untiled buttons, in a grid-like arrangement suggestive of a floating palette. The spacing between buttons would allow data behind the palette to be partially visible when the palette is not in use. If the user moves their cursor over the palette, the nearest button expands to facilitate expansion. This is comparable to existing palettes that use dynamic transparency to show more of the data behind the palette when not in use [Gutwin et al. 2003].

Another example of untiled targets that would benefit from expansion are icons, or other sparsely distributed objects, on a virtual desktop. Although in motor space the icons may be packed together, making selection easier, in visual space the user is free to leave irregularly arranged empty spaces between icons, which can aid spatial memory, and also allows decorative virtual wallpaper to be displayed between icons. Furthermore, although targets would be larger in motor space, they needn't *completely* cover it — there could still be regions where the user could click to invoke a desktop menu. Expanding targets have also been used to aid the selection of small window decorations [Cockburn and Firth 2003].

Untiled expanding targets could also be arranged along the periphery of a window. Figure 8 shows an interface for viewing a 3D mesh with two kinds of expanding widgets. The figure shows how expanding targets not only make selection easier, but can use their expanded size to show the user more data (e.g. an enlarged preview of a camera view) or more information about an option (e.g. a preview of menu contents) just prior to selection. The menu in the upper right corner is also an example of how a single target may contain sub-targets that are tiled.

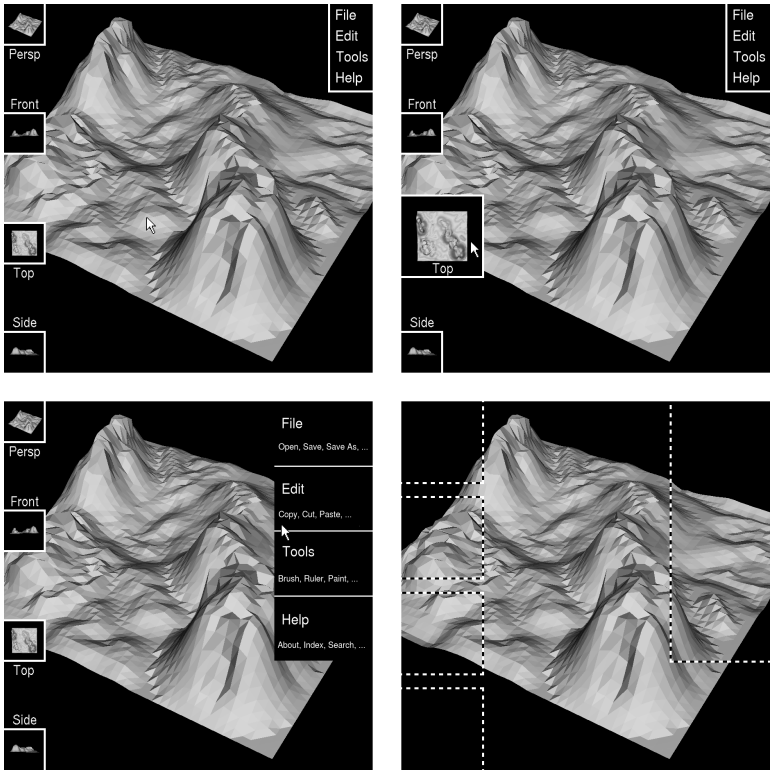

Fig. 8. In this interface, buttons for switching to different views of the mesh are on the left edge of the window, and a menu is in the upper right corner. (Note that, even without expanding targets, the edges and corners of a *screen* are known to be particularly easy to select, however in this case we do not assume that the window covers the entire screen.) *Upper Left:* the cursor is near the centre of the window, and the widgets are in their "rest" state, allowing the mesh being viewed to occupy more screen space. *Upper Right:* the cursor approaches a button, and the button expands, making itself easier to acquire and also showing the user an enlarged preview of the view that would be selected. *Lower Left:* the cursor approaches the menu, which expands and shows previews of the items under each submenu. *Lower Right:* dotted lines show the fixed, expanded size of the widgets in motor space.

Note again that, because there is a static mapping from motor space to buttons, the space covered by the expanded widgets cannot be used to click on the mesh. The space revealed when the widgets are not expanded is used for output only (i.e. displaying the mesh); moving into this space causes expansion of the nearest target. This limitation is shared by most other selection facilitation techniques: in making targets easier to select, they generally make it more difficult or impossible to select empty space between the targets. For example, in object pointing [Guiard et al. 2004], the cursor completely skips over empty space, in an effort to maximize pointing performance. With untiled expanding targets, however, some empty space can still be made accessible to the user (e.g. the central area in Figure 8), if the targets do not completely cover motor space.

Keeping this in mind, the advantages of untiled expanding buttons are that they

do not take up the screen space of large buttons, but at the same time should be as easy to select as large buttons.

4.4 Tiled Expanding Targets Without Motor Expansion

Widgets are often grouped into tiled arrays to save screen space, such as in toolbars or menus containing adjacent buttons or items. This section considers tiled expanding targets where expansion depends only on the current cursor position. Although this prevents true expansion in motor space, such expanding widgets still have useful applications.

For simplicity, we consider one dimensional arrays or strips of widgets (e.g. Figures 9 and 10). It is worth noting that, for such strips, we actually have a combination of the cases in Figure 6: the targets are tiled along the horizontal dimension, but are not tiled along the vertical dimension. Thus, along the vertical dimension, there is no reason we cannot have targets expanded in motor space, to ease selection when approaching a target from above or below (Figure 11). However, the more important issue for this section and the next is that the targets *are* tiled along the horizontal dimension. Thus, when moving sideways through the strip, expansion in motor space is either impossible (in this section) or requires a dynamically changing mapping (§4.5). The observations we make regarding this issue generalize to two dimensional tilings of targets.

In this section, because motor expansion along the tiled dimension is not possible, the main advantage of expansion is in providing enhanced visual information or in showing more data associated with the targets. The principle design question is how to reduce mutual interference between these tiled targets during expansion.

Figure 6 B shows one possibility for visual expansion: the expanded target simply occludes its neighbours. This has the disadvantage that if the user is near but not over the desired target, the expansion of a neighbouring target makes it more difficult for the user to visually identify the desired target. An alternative to allowing any occlusion is to shift neighbouring targets sideways when one is expanded. This is the basis for the following design.

4.4.1 *Imitating the Mac OS X dock.* Consider a strip of buttons where the button closest to the cursor is expanded, and adjacent buttons are moved out of the way to avoid occlusion. Furthermore, to create smooth transitions between successively expanded buttons, neighbouring buttons are also partially expanded. This scheme is used in the Mac OS X dock [Apple Computer, Inc. 2001], as well as in a software prototype[1] we implemented. Note that our prototype improves slightly on the Mac OS X dock in that icons expand *before* the cursor is on top of them when approaching from above or below, thus aiding selection, whereas with the dock, the cursor must be over an icon before it expands. Figures 9 A and B show the prototype's button strip before and after the cursor moves over a button. Unfortunately, when approaching a target from the side, the expansion and contraction of neighbouring icons creates a significant sideways motion, shifting the target's position in visual space and making it more difficult to acquire (Figure 9

[1]Online versions of the prototypes in Figures 9 and 10 are available at `http://www.dgp.toronto.edu/~mjmcguff/research/`

C). This problem is also present in the Mac OS X dock.

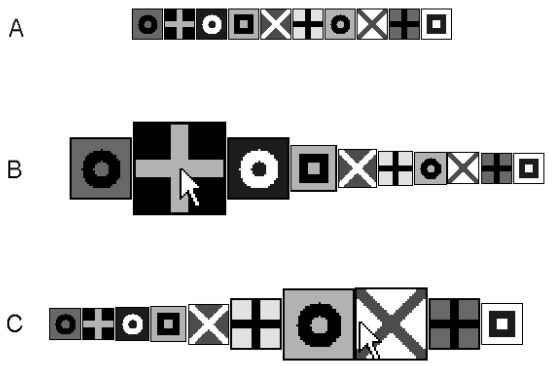

Fig. 9. A design that roughly imitates the dock in Mac OS X. *A:* The buttons are un-expanded when the cursor is far away. *B:* A button is fully expanded when the cursor is over it, and neighbouring buttons are partially expanded and pushed sideways. *C:* A user starting in the state shown in *B* may try to move to the right to select the button with the light X on the dark background. By the time the cursor reaches the desired button's location, the button has moved to the left and the user is now over a different button (one with a *dark* X on a *light* background).

Interestingly, the same shifting problem occurs in 2D when looking at targets through a fisheye lens that is centred at the mouse cursor. As described by Gutwin [2002], approaching a target seen through such a fisheye lens causes the target to move in the direction opposite to the cursor's motion. This is the very same problem that occurs in the Mac OS X dock and the prototype in Figure 9, which can be thought of as 1D fisheye lenses.

As a remedy, Gutwin suggests reducing the magnification of the fisheye lens as a function of cursor speed [Gutwin 2002]. This idea might be adapted to strips of expanding widgets, however we will consider a simpler approach. The next design involves a compromise between occlusion and shifting of neighbouring targets.

4.4.2 *Overlapping Buttons.* To avoid excessive sideways shifting of buttons, we designed a second prototype that allows *limited* overlap between neighbouring buttons (Figure 10). Some sideways shifting of buttons is performed, but only enough to limit the overlap to be of a given amount. Specifically, we use two criteria to determine the layout of buttons. First, the layout generated is such that no button is occluded more than a given percentage, the *Max Occlusion* factor, that can be tuned to adjust behaviour. Second, buttons that are occluded are always expanded at least enough so that their visible area is equal to their original unoccluded area. This ensures a rough lower bound on how difficult they are to see at any given time.

One property of our design is that, even with a *Max Occlusion* factor of 0 % (i.e. no occlusion allowed), which forces buttons to move sideways maximally, our design remains well-behaved in the sense that a fully expanded target will cover all the possible positions that its unexpanded and shifted self could appear in, thus reducing the likelihood of incorrect selections.

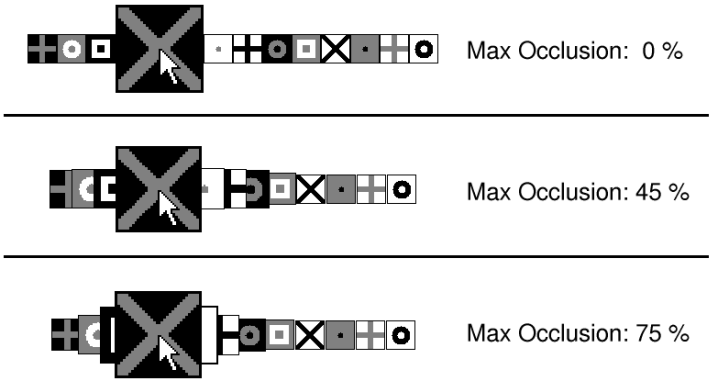

Fig. 10. In this design, limited overlap is allowed between adjacent buttons, which alleviates the problems caused by sideways motion in Figure 9. The *Max Occlusion* factor controls the amount of overlap between neighbouring buttons.

Informal testing with the overlapping buttons design indicates that, with reasonable expansion factors (200 to 400 %), good values for the *Max Occlusion* factor fall between 20 and 50 %. Note that use of transparency and appropriate icon design might further reduce the drawbacks of partial occlusion of targets.

4.4.3 *Summary.* Although the tiled expanding targets just considered are not expanded in motor space (at least, not along the tiled dimension), the visual expansion of targets can be used to display more data or more detailed previews associated with targets, as was sketched in Figure 8, while still allowing targets to be efficiently packed into a small screen space when not in use. Furthermore, if targets are only tiled along one dimension, they can be expanded in motor space along the other dimension, to aid selection along that direction.

Figures 9 and 10 illustrate two designs that are equivalent in terms of motor space (Figure 11), but that differ critically in the feedback given in visual space. Considerations for visual feedback reveal a tension between allowing occlusion of neighbours, which can interfere with the visibility of a desired target, versus shifting of neighbours, which creates moving targets during sideways cursor motion. We feel the design in Figure 10 is a good hybrid in that it allows for an adjustable trade-off between these two effects, and might be further improved using transparency.

4.5 Tiled Expanding Targets With Motor Expansion

We now consider schemes where the mapping from motor space to targets is dynamically updated based on prediction of the user's desired target. Such schemes have the potential of allowing tiled targets to be expanded in motor space, yielding all the potential benefits of expanding targets with seemingly no drawbacks. Such schemes have also not been successfully implemented to date.

The basic idea for such expansion has been described before [McGuffin 2002; Zhai et al. 2003]: as the user moves their cursor, if and when the system's predictor determines which target the user is likely aiming for, the system increases the size of that target in motor (and visual) space. The expanded target must retain its

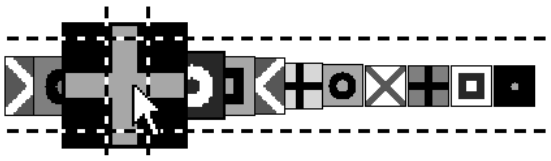

Fig. 11. If targets are tiled along one dimension, and expansion depends only on the current cursor position, expansion in motor space is only possible along the other dimension. Dashed lines delimit the footprint of a button in motor space. The cursor must be within this rectangle to acquire the button. Although the button looks larger than this rectangle in visual space, its full visual size is not available to the user: as soon as the cursor moves off the button's centre, the button begins to contract. The rectangle in motor space is in fact no wider than the unexpanded button, however it is taller, which should ease selection somewhat if the cursor approaches from above or below [Accot and Zhai 2003].

enlarged size long enough that the user may complete their selection with the benefit of the target's expanded size, resulting in widget hysteresis: the configuration of widgets depends not just on the current cursor position, but also on its history. After a sufficient delay, if the user hasn't selected the predicted target, and/or the system has determined that its prediction was wrong, the configuration of targets in motor space may return to its normal state.

In keeping with the previous section, we continue to consider horizontal strips of buttons that are tiled along one dimension. The critical question is whether we can horizontally expand targets in motor space when the user is moving sideways through the strip. When approaching from above or below, expansion in motor space is easy to achieve, even without prediction, since the targets are not tiled in that direction.

4.5.1 *A Model of Expected Benefit.* Before developing a specific predictor of the user's intended target, we have found it useful to quantitatively model a simple expansion scheme in a way that leaves the performance of the predictor as an unknown parameter. The quantification is in terms of ID, in bits.

Figure 12 illustrates a simple expansion scheme where only one target is expanded, and neighbours are occluded with no shifting or partial expansion of them. If the prediction of the user's desired target is correct, expansion of the target will benefit the user. If the prediction is off by one target, however, the desired target is partially occluded by expansion of its neighbour, making selection for the user more difficult.

Both cases amount to a change in the width of the button from W to W', changing the button's index of difficulty from $ID = \log_2(A/W)$ to $ID' = \log_2(A/W')$. (Note that the form of ID used here is similar to Fitts' original form, where the K in Equation 1 is zero. Although $K = 1$ might arguably be better, this would only complicate our analysis without changing the essential results of it, and the two forms are very close for large A.) The number of bits by which a change in width reduces the index of difficulty is $R = ID - ID' = \log_2(W'/W)$.

A given predictor will be correct for some selections, off by one target other times, off by two targets other times, etc. The output of the predictor has a probability

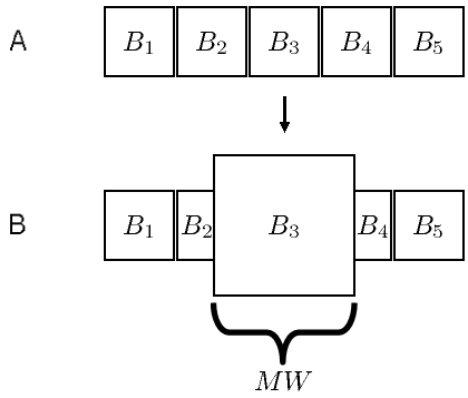

Fig. 12. *A:* Buttons tiled along a strip, each of width W. *B:* The system predicts that the user wishes to select B_3 and expands it in visual and motor space by a factor of M, partially occluding the immediately neighbouring buttons. The ID of B_3 has been reduced by $R = \log_2(M)$ bits, which should help the user select B_3 faster. However, the IDs of B_2 and B_4 have been increased, because in their case, $R = \log_2(1 - (M-1)/2) = \log_2((3-M)/2)$ bits, which is negative. This would hinder the user if either B_2 or B_4 were the real intended target.

distribution associated with it which we can consider to be centred at the true intended target. A reasonably designed predictor should have, at worst, a flat distribution (equivalent to random prediction), with a probability of only $1/N$ of being correct when there are N buttons. A better predictor can be expected to have a distribution that peaks at the correct target and falls off with targets further away.

Let p be the probability that the predictor is correct, and q the probability it is off by one target. It follows that $p + q \leq 1$, and the predictor will be off by more than one target with probability $1 - p - q$. Furthermore, because we expect the distribution to be at worst flat, and anything better to peak at the correct target, we can assume $p \geq q/2$.

Let M be the expansion factor for the target (Figure 12), with $0 < M < 3$ to avoid total occlusion of any target. The net reduction R in ID is a weighted average of the benefit from correct predictions and the penalty from incorrect predictions:

$$\begin{aligned} R &= p\log_2(W_{expanded}/W) + q\log_2(W_{occluded}/W) + (1-p-q)\log_2(W/W) \\ &= p\log_2(M) + q\log_2\left(\frac{3-M}{2}\right) + (1-p-q)\times 0 \\ &= \log_2\left(M^p\left(\frac{3-M}{2}\right)^q\right) \end{aligned} \tag{6}$$

If the values of p and q are known (either at design time, or through live measurements), the interface can adjust the value of M to maximize R. Finding the derivative of the above expression for R with respect to M, and then setting it to zero, reveals that the best M is $M_{optimal} = 3p/(p+q)$. (This further explains why we assume $p \geq q/2$. If $p < q/2$, then $M_{optimal} < 1$, meaning that the predictor is off by one so often that it is better on average to *shrink* the predicted target,

creating more room for both neighbours.)

As an example, if the predictor is correct half of the time and off by one the other half of the time, we have $p = q = 0.5$ and $M_{optimal} = 1.5$, yielding $R \approx 0.085$ bits, which is a small but positive advantage. The model indicates that, despite the penalty from incorrect predictions, an overall benefit from expansion may, on average, be achievable. However, because our model assumes the user's performance depends only on the target's final size in motor space, it is prudent to qualify R as an *upper bound* on the expected reduction in ID. Still, given the results of Zhai et al. [2003], where performance was governed by the target's final size even if the user did not know ahead of time if targets would expand or not, R may be a good indicator of the real world benefit.

Keeping in mind that R is an upper bound, we can substitute $M = M_{optimal}$ in Equation 6 to obtain

$$R = \log_2 \left(M_{optimal}^{p} \left(\frac{3 - M_{optimal}}{2} \right)^{q} \right) = \log_2 \left(\frac{3p}{p+q} \right)^{p} \left(\frac{3q}{2(p+q)} \right)^{q} \quad (7)$$

The above equation is plotted in Figure 13. From the contour plot, we see that to obtain a non-negligible, demonstrable benefit (e.g. $R \geq 0.5$), the predictor must have a fairly high p value and/or a fairly low q. Tradeoffs can be made between the values of p and q, however it is not clear whether a predictor can be implemented that would lead to a non-negligible R.

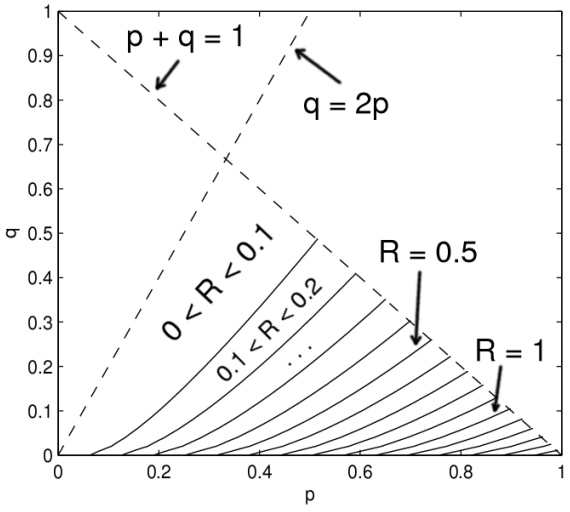

Fig. 13. Contour plot of the reduction R in ID as a function of p and q. We only consider the region below the dashed lines, corresponding to $p + q \leq 1$ and $q \leq 2p$. R is zero along the dashed line $q = 2p$ and increases as p increases and as q decreases, to a maximum of $\log_2(3) \approx 1.585$ bits at $(p, q) = (1, 0)$. Contour lines are plotted at intervals of 0.1 bits.

4.5.2 *Measured Accuracy of a Simple Predictor.* To investigate the level of accuracy that could be expected from a real world predictor, we implemented a simple predictor and measured its performance in a pilot experiment. In the experiment,

participants were asked to select one of a tiled set of targets in trial after trial. During each trial, a separate software component collected mouse motion events and tried to anticipate which target the user was aiming to select. There was no expansion of targets, as we were only interested in measuring the performance of the predictor.

At the start of each trial, the user had to place the cursor in a small start box on the left of the screen and dwell there for 0.5 seconds. Then 8 horizontally tiled targets, each of equal width W, appeared to the right of the start box, with one target highlighted which had to be selected by the user (Figure 14). The distance from the centre of the start box to the left edge of the first target was 50 units (where 1 unit $\approx$ 8 pixels), the width W of targets varied from trial to trial between 2, 3, 5, 7, and 10 units, and the button that was highlighted also varied from trial to trial, which varied the amplitude A of the required motion. The targets were tall enough that they almost covered the entire vertical extent of the screen, so that pointing only required one dimensional motion along the horizontal axis. The user had to successfully click on the highlighted target to complete the trial. Output was displayed on a 19-inch 1280x1024-pixel screen, and the experimental software, written in C++, ran on a 2.4 GHz Pentium4 PC running Microsoft Windows XP.

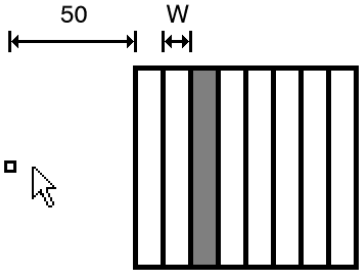

Fig. 14. Stimuli for pilot study of prediction accuracy. Users moved from the start box on the left to the highlighted target (in this case, the third target).

5 users participated in the experiment, all male, all right-handed, all experienced users of mice (though a puck and tablet were used as the pointing device in the experiment, for the same reasons as in §3.1), aged approximately between 19 and 22 years. Each participant performed 3 blocks of trials, with blocks separated by rest periods, and preceded with warm up trials. There were 8 possible highlighted buttons, and 5 possible values for W, creating 40 different A,W conditions. Each of the 40 conditions was repeated 3 times within a block, yielding 120 trials in each block. Thus, a total of 5 users $\times$ 3 blocks/user $\times$ 120 trials/block = 1800 trials were completed. Each participant completed the 3 blocks in approximately 1 to 1.5 hours.

In a real user interface, the prediction of the desired target could rely partially on the history, patterns, and frequency of previously selected buttons. However, prediction based on this information is not really useful in an experimental situation with randomized conditions. It also has the disadvantage of requiring an initial

training phase to learn about the user's habits, and generally runs the risk of performing poorly if the user's habits suddenly change.

We wanted a prediction scheme that would be more generally applicable and entail fewer risks in real interfaces. Thus, we based our prediction algorithm solely on the cursor's trajectory. Every time a new mouse motion event is captured, the algorithm assumes a constant acceleration and quadratically extrapolates the three most recent $(time, x)$ events, ignoring the cursor's y coordinate. It then checks if there is a point in the future where velocity is zero. This point is the predicted final position of the cursor. If such a point exists, and the distance remaining to that point is less than 10 % of the distance from the start box to the predicted point, then the algorithm commits to that prediction as its final output, and no further prediction is attempted for the remainder of the trial. The 10 % threshold was chosen because, in a real interface, if the prediction were driving the expansion of a target, we would want the expansion to occur early enough to allow the user to take full advantage of it.

The results of the experiment revealed that, on average, over all participants, the prediction algorithm was correct with a rate of $p = 0.211$, and off by one with a rate of $q = 0.262$. The other $1 - p - q$ fraction of the time, it was off by more than one target. Examining Figure 13, we see that the point $(p, q) = (0.211, 0.262)$ falls in the region of the contour plot where $0 < R < 0.1$. Substituting $(p, q) = (0.211, 0.262)$ into Equation 7 yields $R = 0.019$ bits, or less than 0.02 bits reduction in ID. Thus, according to our model, if our predictor had been driving the expansion of a target, there may have been a net positive advantage for the user, however this advantage would be so small as to be undetectable in an experimental study, unless perhaps the sample size were prohibitively large. There was some variation in the predictor's performance across users, however even the best performance with a single user was $(p, q) = (0.253, 0.308)$, yielding only $R = 0.024$ bits. Keep in mind also that R is best thought of as an upper bound on the benefit to the user, since our model has not been validated experimentally.

4.5.3 *Discussion.* The predictor we used was one of the simplest that could be used. It is possible that a better predictor could be designed, perhaps building on other work in trajectory prediction or target prediction [Murata 1998; Baldwin et al. 1998; 1999; Münch and Dillmann 1997; Keuning-Van Oirschot and Houtsma 2001], to analyze the cursor trajectory in a more sophisticated manner.

However, even given a better predictor, there are plausible reasons why it may not be of much use when coupled with target expansion. As described in §2.2, the movement involved in a Fitts' targeting task involves an initial impulse that may overshoot or undershoot the target, followed by subsequent corrective submovements as necessary. Target expansion aids the user because the corrective submovements required are fewer and/or smaller. Target expansion is most helpful when one of the early movements toward the target either overshoots or undershoots the target, and the expansion "catches" the cursor anyway, or at least eases correction. However, if such incorrect initial movements are input to a predictor that extrapolates their trajectory, this will likely lead to a wrong prediction. On the other hand, if a movement input to the predictor extrapolates to the correct target, this is precisely when expansion is of the least use to the user, because in

any case such a movement will likely fall on the unexpanded target area.

Another issue is that our model does not take into account that, even if the ID is reduced on average, in practice users may be very frustrated with the system when its predictions are wrong. Poorly designed adaptive user interfaces are often turned off by the user just to eliminate the frustration caused by incorrect adaptation.

Given this reasoning, and especially the poor performance of our predictor, we have so far not pursued experimental studies of tiled targets with motor expansion. Whether selection of tiled targets can be facilitated in practice is still an open question. In addition to improving the performance of the predictor, another way to ease selection of tiled targets may be to improve the rather simple expansion scheme depicted in Figure 12. Our model assumes that the immediate neighbours of the predicted target are occluded during expansion. However, it may be possible to reduce the penalty from incorrect predictions, and thus increase R, by partly shifting neighbouring targets rather than occluding them. Zhai et al. [2003] sketch a design of tiled targets that expand in motor space and where neighbours are shifted out of the way. (Note that, in their description, motor expansion is only done when the motion is *not* along the tiled dimension, however their design might be adapted to always expand targets in motor space.) The following section considers yet another alternative design that may reduce the penalty from incorrect predictions.

4.5.4 *Expansion with a Fixed Edge.* To reduce the penalty from incorrect predictions, an alternative to expanding the predicted target around its centre is to expand it around its edge closest to the predicted *target point* (the predicted location that the cursor will come to rest at). For example, if the system predicts that the cursor will land within the right half of button B at the end of the motion, then B is expanded around its right edge, meaning that the neighbour B^r to the right of B is not occluded or shifted at all. If it turns out that the prediction is incorrect, then the user was most likely really aiming for B^r, whose ID has not changed. The key notion here is that the edge between the two most likely buttons remains fixed during expansion, so that there is no penalty to acquire the second most likely button.

A variation on this would be to partially expand B^r as well (Figure 15), since it is the second most likely button desired by the user. Thus, even when the prediction is off by one, the user will most likely be aided by the expansion. Of course, B's left neighbour B^l (and all subsequent neighbours to the left), and B^r's right neighbour B^{rr} (and all subsequent neighbours to the right), will be occluded and/or shifted, which will penalize the user if the user was not aiming for B or B^r.

Unfortunately, we cannot model the expected benefit of such a design as we did previously, because it is not known how a shift in a target's position affects its ID. Jagacinski et al. [1980] and Hoffmann [1991] studied how Fitts' law changes when a target moves with constant velocity. More recently, Port et al. [1997] developed models of performance at a task where users must *intercept* a moving target within a given "interception zone". Unfortunately, to our knowledge, there have been no studies of tasks where users had to capture a target that begins moving *after* the user has started to move toward the target. Thus, we cannot yet model the index of difficulty for a target that moves as the user approaches it. Although we do not have enough information to quantitatively analyze fixed-edge expansion, this

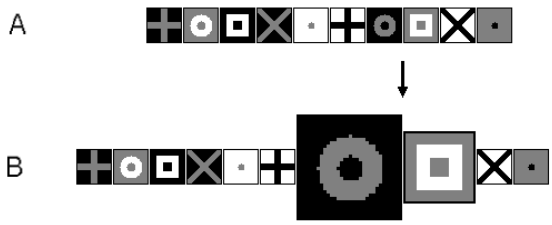

Fig. 15. In this expansion scheme, the edge closest to the predicted target point is held fixed, and the two most likely target buttons are expanded around this edge. All other buttons are moved sideways. *A:* The buttons at rest. *B:* Expansion resulting from a prediction that the cursor's trajectory is heading for the right half of the 4th button from the right.

scheme may prove better than expansion around a button centre, since it reduces the cost of the *two* most likely target buttons rather than just one. In effect, the required tolerance for correct prediction is $2W$ rather than W — as long as the predicted final point is within $\pm W$ of the desired button's centre, the desired button (and one of its neighbours) will be expanded.

Extending this idea further, we could have the n most likely contiguous buttons be treated as a single target, that is expanded around its centre, and within which the user selects a sub-target. Such expansion would be similar to the expansion of the menu in Figure 8. Of course, making n too large would result in most of the targets shifting sideways significantly, even if a small expansion factor is used, and would also mean that prediction and expansion must be done earlier in the movement to give the user the opportunity to take full advantage of the expansion.

Determining which, if any, of these designs is most viable would require more experimental work, and possibly the development of techniques for modelling the cost incurred from having targets shift in position.

5. CONCLUSIONS AND FUTURE DIRECTIONS

5.1 Conclusions

We have presented experimental work that investigates parameters and performance of expanding targets. Our results show that, when users expect the target to expand, they can select a single, isolated expanding target faster than a non-expanding, static target, even if expansion occurs after 90 % of the distance toward the target has been travelled. (This finding was also confirmed in Zhai et al.'s [2003] follow-up study, who also obtained the same result even when users did not know whether expansion would occur or not). Furthermore, for sufficiently high ID values, our data suggests that users benefit fully from target expansion, i.e. performance is approximately as good as, or better than, the best that could be expected given the rationale for our lower bound on movement time. These results indicate that the corrective movements toward the end of a motion can be made to take advantage of fairly late changes in target size. As a rough predictor of performance with expanding targets, we propose using Fitts' law with the final target size, where the intercept a and slope b may be computed from a base set of data

from static targets.

For the case of multiple expanding targets, we have examined three design approaches: untiled targets (§4.3), tiled targets without motor expansion (§4.4), and tiled targets with motor expansion (§4.5). The first two are readily applicable in real user interfaces, however their respective advantages and disadvantages must be kept in mind: untiled expanding targets ease selection but consume more motor space than static widgets; tiled expanding targets without motor expansion allow for a greater area for visual feedback but do not ease selection along the direction(s) they are tiled.

The third approach, of tiled expanding targets with motor expansion, offers potentially the greatest advantages, but has not yet been demonstrated to be workable, and is not ready to be applied in real interface design work. The model presented (§4.5.1) indicates that a net reduction in selection time with tiled expanding targets may be possible, however in practice the benefit may be negligibly small.

5.2 Future Directions

Gathering more data for single expanding targets would allow for a more accurate and complete model of performance to be developed. Our data did not allow us to definitively conclude that performance with expanding targets really tends toward the theoretical bound as ID increases, however we suspect this would be a likely finding in a more extensive study. In addition, data could be collected with $W_{expanded}/W$ ratios other than 2. It would be theoretically interesting to test the condition $W_{expanded}/W < 1$, to see how much the effects of *shrinking* targets mirror the effects of expanding targets.

Some of the designs discussed for multiple expanding targets involve shifting neighbouring targets sideways. Unfortunately, performance with these designs cannot yet be modelled accurately, as we are unaware of any models of selection where the target moves *after* the user has started moving toward the target. Future work could develop a quantitative model for such moving targets, and then apply the model to designs involving multiple moving/expanding targets.

It is plausible that having multiple expanding targets on a screen may cause distraction and reduce performance. Experiments with multiple targets could be designed to test for this, and perhaps determine conditions where the distraction is eliminated, or at least reduced. (A potential design strategy for reducing distraction is to have targets expand in motor space but not in visual space. For example, a system might try to predict which target a user is heading for, and then highlight that target, indicating to the user that they may now "click ahead" rather than complete their motion. If the prediction turns out to be wrong, the user can ignore the highlighting and click on the target they really want. Like a *flick* gesture [Dulberg et al. 1999], this would allow the user to perform selections by simply moving in the direction of a target; however, unlike flicking, the user must also click to complete the selection, and this confirmation step provides the user with the control to prevent misinterpretations.)

Facilitating the selection of tiled targets is still an open challenge; improving performance with tiled targets is equivalent to exceeding the normal index of performance IP in Fitts' law. We have not identified any definitive reason why it should be impossible to achieve this in principle, however future researchers should

be wary that this goal may be practically impossible.

ACKNOWLEDGMENTS

Many thanks to Vikas Jain who implemented the trajectory prediction algorithm, and conducted the experimental evaluation of it described in §4.5.2. Thanks also to Wolfgang Stürzlinger, Joe Laszlo, Gord Kurtenbach, George Fitzmaurice, Azam Khan, Scott MacKenzie, Michel Beaudouin-Lafon, Yves Guiard, Carl Gutwin, Simone Maillard, Etty Shin, and members of the Dynamic Graphics Project lab at the University of Toronto for providing ideas, discussions, help, and reactions to the work as it progressed. We also thank the participants in our study. Finally, thanks to the reviewers of our article, for their patience and careful work in suggesting improvements.

REFERENCES

Accot, J. and Zhai, S. 2003. Refining Fitts' law models for bivariate pointing. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 2003*. 193–200.

Akamatsu, M., MacKenzie, I. S., and Hasbroucq, T. 1995. A comparison of tactile, auditory, and visual feedback in a pointing task using a mouse-type device. *Ergonomics 38,* 4, 816–827.

Apple Computer, Inc. 2001. The "Dock", a feature of the "Mac OS X" operating system. http://www.apple.com/macosx/theater/dock.html Accessed 2001–2004.

Baldwin, J., Basu, A., and Zhang, H. 1998. Predictive windows for delay compensation in telepresence applications. In *Proceedings of the 1998 IEEE International Conference on Robotics & Automation*. 2884–2889.

Baldwin, J., Basu, A., and Zhang, H. 1999. Panoramic video with predictive windows for telepresence applications. In *Proceedings of the 1999 IEEE International Conference on Robotics & Automation*. 1922–1927.

Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B., and Zierlinger, A. 2003. Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch- and pen-operated systems. In *Proceedings of INTERACT 2003*. 57–64.

Bederson, B. 2000. Fisheye menus. In *Proceedings of ACM Symposium on User Interface Software and Technology (UIST) 2000*. 217–225.

Bier, E. A. and Stone, M. C. 1986. Snap-dragging. In *Proceedings of ACM SIGGRAPH 1986*. 233–240.

Blanch, R., Guiard, Y., and Beaudouin-Lafon, M. 2004. Semantic pointing: Improving target acquisition with control-display ratio adaptation. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 2004*. 519–526.

Cockburn, A. and Firth, A. 2003. Improving the acquisition of small targets. In *People and Computers XVII: British Computer Society Conference on Human Computer Interaction (HCI 2003)*. 181–196.

Crossman, E. R. F. W. and Goodeve, P. J. 1983. Feedback control of hand-movement and Fitts' law. *Quarterly Journal of Experimental Psychology 35A*, 251–278. (Original work presented at the meeting of the Experimental Psychology Society, Oxford, England, July 1963).

Dulberg, M. S., St. Amant, R., and Zettlemoyer, L. S. 1999. An imprecise mouse gesture for the fast activation of controls. In *Proceedings of INTERACT '99*. 375–382.

Feiner, S., Nagy, S., and van Dam, A. 1981. An integrated system for creating and presenting complex computer-based documents. In *Proceedings of ACM SIGGRAPH 1981*. 181–189.

Fitts, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology 47,* 6 (June), 381–391. (Reprinted in Journal of Experimental Psychology: General, 121(3):262–269, 1992).

Furnas, G. W. 1986. Generalized fisheye views. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 1986*. 16–23.

Grossman, T. and Balakrishnan, R. 2005. The bubble cursor: Enhancing target acquisition by dynamic resizing of the cursor's activation area. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 2005*. 281–290.

Guiard, Y., Blanch, R., and Beaudouin-Lafon, M. 2004. Object pointing: A complement to bitmap pointing in GUIs. In *Proceedings of Graphics Interface (GI) 2004*. 9–16.

Gutwin, C. 2002. Improving focus targeting in interactive fisheye views. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 2002*. 267–274.

Gutwin, C., Dyck, J., and Fedak, C. 2003. The effects of dynamic transparency on targeting performance. In *Proceedings of Graphics Interface (GI) 2003*. 105–112.

Hoffmann, E. R. 1991. Capture of moving targets: A modification of Fitts' law. *Ergonomics 34,* 2, 211–220.

Hoffmann, E. R. 1995. Effective target tolerance in an inverted Fitts task. *Ergonomics 38,* 4, 828–836.

Jagacinski, R. J., Repperger, D. W., Ward, S. L., and Moran, M. S. 1980. A test of Fitts' law with moving targets. *Human Factors 22,* 2, 225–233.

Kabbash, P. and Buxton, W. A. S. 1995. The "prince" technique: Fitts' law and selection using area cursors. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 1995*. 273–279.

Keele, S. W. 1968. Movement control in skilled motor performance. *Psychological Bulletin 70,* 6 (December), 387–403.

Keuning-Van Oirschot, H. and Houtsma, A. J. M. 2001. Cursor displacement and velocity profiles for targets in various locations. In *Proceedings of Eurohaptics 2001*. 108–112.

Keyson, D. V. 1997. Dynamic cursor gain and tactual feedback in the capture of cursor movements. *Ergonomics 40,* 12, 1287–1298.

MacKenzie, I. S. 1989. A note on the information-theoretic basis for Fitts' law. *Journal of Motor Behavior 21,* 3, 323–330.

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

McGuffin, M. and Balakrishnan, R. 2002. Acquisition of expanding targets. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 2002*. 57–64.

McGuffin, M. J. 2002. Fitts' law and expanding targets: An experimental study, and applications to user interface design. Master of Science (M.Sc.) thesis, Department of Computer Science, University of Toronto, Toronto, Ontario, Canada.

Meyer, D. E., Abrams, R. A., Kornblum, S., Wright, C. E., and Smith, J. E. K. 1988. Optimality in human motor performance: Ideal control of rapid aimed movements. *Psychological Review 95,* 3, 340–370.

Meyer, D. E., Smith, J. E. K., Kornblum, S., Abrams, R. A., and Wright, C. E. 1990. Speed-accuracy tradeoffs in aimed movements: Toward a theory of rapid voluntary action. In *Attention and Performance XIII*, M. Jeannerod, Ed. Lawrence Erlbaum, Hillsdale, NJ, 173–226. http://www.umich.edu/~bcalab/Meyer_Bibliography.html.

Münch, S. and Dillmann, R. 1997. Haptic output in multimodal user interfaces. In *Proceedings of ACM International Conference on Intelligent User Interfaces (IUI) 1997*. 105–112.

Murata, A. 1998. Improvement of pointing time by predicting targets in pointing with a PC mouse. *International Journal of Human-Computer Interaction 10,* 1, 23–32.

Oakley, I., Adams, A., Brewster, S. A., and Gray, P. D. 2002. Guidelines for the design of haptic widgets. In *People and Computers XVI: British Computer Society Conference on Human Computer Interaction (HCI 2002)*.

Oakley, I., Brewster, S. A., and Gray, P. D. 2001. Solving multi-target haptic problems in menu interaction. In *Extended Abstracts of ACM Conference on Human Factors in Computing Systems (CHI) 2001*. 357–358.

Port, N. L., Lee, D., Dassonville, P., and Georgopoulos, A. P. 1997. Manual interception of moving targets: I. Performance and movement initiation. *Experimental Brain Research 116,* 3, 406–420.

Shoemake, K. 1992. ARCBALL: A user interface for specifying three-dimensional orientation using a mouse. In *Proceedings of Graphics Interface (GI) 1992*. 151–156.

Sutherland, I. E. 1963. Sketchpad: A man-machine graphical communication system. In *Proceedings of AFIPS Spring Joint Computer Conference*. 328–346.

Woodworth, R. S. 1899. The accuracy of voluntary movement. *The Psychological Review: Monograph Supplements (also known as Psychological Monographs) 3,* 2 (Whole Number 13) (July), 1–114.

Worden, A., Walker, N., Bharat, K., and Hudson, S. 1997. Making computers easier for older adults to use: Area cursors and sticky icons. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 1997*. 266–271.

Zhai, S. 2002. On the validity of throughput as a characteristic of computer input. Tech. Rep. IBM Research Report RJ 10253, IBM Almaden Research Center, San Jose, California. 12 pages.

Zhai, S., Conversy, S., Beaudouin-Lafon, M., and Guiard, Y. 2003. Human on-line response to target expansion. In *Proceedings of ACM Conference on Human Factors in Computing Systems (CHI) 2003*. 177–184.