Benefits of Merging Command Selection and Direct Manipulation

FRANÇOIS GUIMBRETIÈRE
University of Maryland
and
ANDREW MARTIN and TERRY WINOGRAD
Stanford University

Toolglass [Bier et al. 1993] demonstrated a two-handed command selection technique that combined command selection and direct manipulation. While empirical evaluations showed a speed advantage for ToolGlass, they did not examine the relative importance of two possible factors in its improved performance: (1) the use of two hands and (2) the merging of command selection and direct manipulation.

We conducted a study comparing the relative benefits of three command selection techniques that merge command selection and direct manipulation: one two-handed technique, Toolglass, and two one-handed techniques, namely, control menus [Pook et al. 2000] and FlowMenu [Guimbretière and Winograd 2000]. Participants performed sequences of operations that required both selecting a color and designating the endpoints of a line. Our results show that control menus and FlowMenu are significantly faster than Toolglass. Further analysis suggests that the merging of command selection and direct manipulation is the most important factor in the performance of all three techniques.

Categories and Subject Descriptors: I.3.6 [**Computer Graphics**]: Methodology and Techniques—*Interaction techniques*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Evaluation/methodology; Interaction styles (e.g., commands, menus, forms, direct manipulation)*

General Terms: Experimentation, Human Factors, Measurement, Performance

Additional Key Words and Phrases: Control menu, FlowMenu, Toolglass, two-handed interaction, command selection, direct manipulation

This work was supported in part by the DOE VIEWS program (contract B504665) and the Wallenberg Global Learning Network.

Authors' addresses: F. Guimbretière, Human Computer Interaction Laboratory, Department of Computer Science, University of Maryland, A. V. Williams Building, College Park, Maryland 20742; email: francois@cs.umd.edu; A. Martin; email: mistermartin@gmail.com; T. Winograd, Department of Computer Science, Stanford University, Stanford, California 94305-9035; email: winograd@cs.stanford.edu.

1. INTRODUCTION

Following the research of Guiard [1987] on the asymmetric division of labor in skilled human bimanual action, a number of researchers have experimented with two-handed techniques for human-computer interaction [Leganchuk 1998]. They have argued that the use of two hands (one for positioning tools and one for manipulation) provides benefits for direct manipulation interfaces. Among two-handed interaction techniques, Toolglass, a technique introduced by Bier et al. [1993], has received the most attention and was studied empirically by Kabbash et al. [1994]. Kabbash et al. found that Toolglass provides a significant performance advantage over the more traditional tool palette for a simple connect-the-dots task that combines color selection with line drawing. Yet, in his analysis, Kabbash did not consider another important aspect of the Toolglass design: because the tool is see-through (semitransparent), Toolglass users can select a command and proceed with direct manipulation in a single stroke. While not emphasized at the time, this feature is emerging as a new dimension of the design space. For example, both control menu [Pook et al. 2000] and FlowMenu [Guimbretière and Winograd 2000], two recently introduced *one-handed* techniques, merge command selection with direct manipulation. Because there has not been an equivalent study for a one-handed command selection technique that merges command selection and direct manipulation, it has been difficult to understand the relative benefits of two-handedness and the merging of command selection with direct manipulation.

To understand the relative performance benefits resulting from each of these two factors, we adapted the experiment of Kabbash et al. [1994], using their simple colored line-drawing (connect-the-dots) task to compare the commonly used tool palette, Toolglass, and two one-handed techniques that merge command selection and direct manipulation: control menus [Pook et al. 2000] and FlowMenu [Guimbretière and Winograd 2000] (Figure 1). There are other one-handed techniques that merge command selection and direct manipulation, such as pie menus [Hopkins 1991] and the extension of marking menus proposed by Kurtenbach and Buxton [1991]. We chose control menus and FlowMenu because they are the closest to Toolglass's style of interaction. Furthermore we judged that these two techniques are different enough to provide good coverage of the current design space, and their inclusion would ensure that our results were not specific to just one technique.

Our experimental results replicated the finding of Kabbash et al. [1994] that Toolglass is faster than a tool palette for this task and also showed that both control menus and FlowMenu are faster than Toolglass. These results seem to imply that the key factor in Toolglass's performance is its merging of command selection and direct manipulation, not its two-handedness. Based on our results, we propose new directions for the design of efficient command mechanisms.

2. RELATED WORK

Toolglass [Bier et al. 1993] was one of the first interaction mechanisms to merge command selection and direct manipulation. With Toolglass, the user uses his or her nondominant hand to manipulate a translucent tool palette and his or

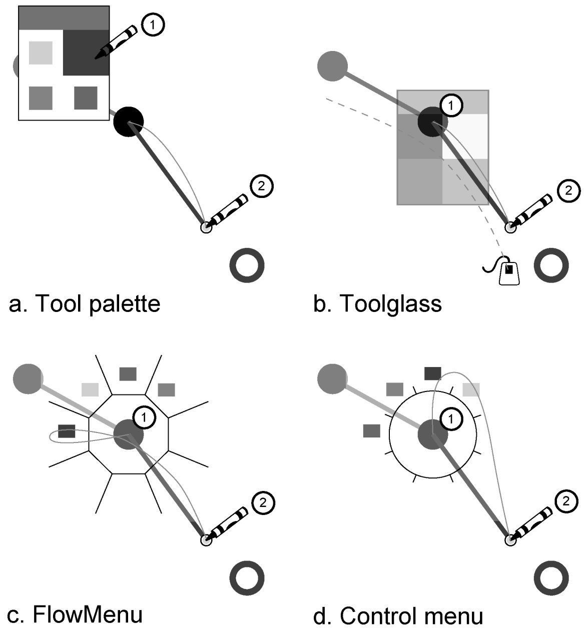


Fig. 1. The four command mechanisms used in this study. Each command mechanism is used to select a color before connecting two dots on the screen by performing a rubber-band line interaction. Tool palette requires the user to first press and release the pen tip on the appropriate color and then connect the dots by pressing the tip of the pen on the starting dot and releasing it on top of the target dot. With Toolglass, a two-handed technique, users move the semitransparent Toolglass with a puck and select the color by starting the connection *through* the correct color. Both control menu and FlowMenu are invoked by clicking the pen command button while pressing the pen tip on the starting dot and then performing a gesture to select the color before completing the connection. The path of the pen on the screen during a connection is shown with a light line. The path of the puck is shown with a light dotted line.

her dominant hand to select commands and perform direct manipulation tasks (Figure 1(b)).

Control menu [Pook et al. 2000] is a radial menu conceptually similar to marking menus [Kurtenbach 1993] that does not require users to release the mouse button (or lift the pen from the screen) to select a command. Instead, control menus use a threshold distance as the triggering mechanism to determine menu selection (Figure 1(d)). This feature allows users to proceed directly from command selection to direct manipulation without interruption.

FlowMenu [Guimbretière and Winograd 2000] is another radial menu that does not require users to release the mouse button (or lift the pen from the screen) to select a command. Instead, FlowMenu requires leaving and reentering the central rest area in specific directions to determine menu selection (Figure 1(c)). Although its principle is similar to control menus, FlowMenu provides additional features such as textual entry and knob interaction (see Guimbretière and Winograd [2000] for further details).

Of these three techniques, only Toolglass has been studied extensively. In Kabbash et al. [1994], the authors compared Toolglass to three other techniques including the conventional tool palette (called *R-tearoff* by Kabbash et al.) and reported significantly better performance in the Toolglass condition.

FlowMenu and control menus are newer techniques and have not yet been studied extensively. A notable exception is a study by McGuffin et al. [2002] that compared FlowMenu, control menus, and FaST Slider for assigning parameter values. In that study, users were asked to observe and adjust the numerical value of eight parameters using FlowMenu, control menus, and FaSTSlider. The study provides qualitative evidence that FlowMenu is more difficult to learn than the other two techniques and that parameter observation and adjustment can be difficult with control menus.

3. HYPOTHESES

To explore how the merging of command selection and direct manipulation influences the performance of interaction techniques, we compared the performance of the following interaction techniques:

(1) Tool palette represents the current status quo. It uses a sequential command assemblage [Kabbash et al. 1994] and neither merges command selection and direct manipulation nor relies on two-handed interaction;
(2) Toolglass uses an asymmetric dependent command assemblage [Kabbash et al. 1994] and merges command selection and direct manipulation by using two hands to perform the task;
(3) FlowMenu and control menus use a sequential command assemblage and merge command selection and direct manipulation using one hand.

Kabbash et al. [1994] results suggest that an asymmetric dependent assemblage [Guiard 1987] is faster than a sequential command assemblage so we expected Toolglass to be faster than tool palettes, FlowMenu, and control menus.

Within the sequential assemblage group, we expect FlowMenu and control menus to be faster than tool palette since they merge command selection and direct manipulation. In doing so, they save users the time it takes to go back and forth between the current locus of attention and the color selection tool.

Predicting the difference between FlowMenu and control menus is somewhat more difficult given the lack of empirical data for both systems. Looking at typical traces (as shown in Figure 2), the most striking difference is the complexity of the overall FlowMenu gesture with the control menu paths being significantly simpler. Based on this observation, we postulated that control menus should be faster than FlowMenu.

4. EXPERIMENT

Like Kabbash et al. [1994] (and previously Dillon et al. [1990]), we used the connect-the-dots task for our experiment. In this task, a series of colored dots are presented one by one to the participant. As soon as a new dot appears, the participant selects the matching color using the technique under study and then connects with a rubber-band line from the last dot in the current path to

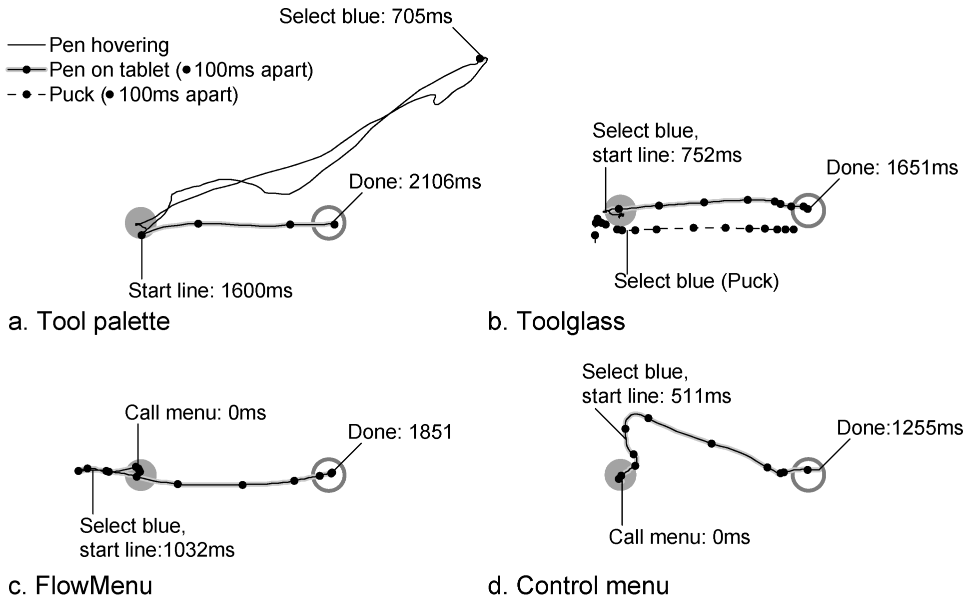


Fig. 2. Typical traces for each technique used in this experiment. The user selects a color (in this case blue) and connects from the grey circle to the white one. The trial clock starts as soon as the previous connection is completed. Annotations show the different steps required for each technique. Numbers associated with each step show the time elapsed in milliseconds. Note that to limit skill transfer between the FlowMenu and Control menu condition, blue was placed on the left in the FlowMenu condition and on top in the control menu condition (see Figure 1).

the new dot. Upon completion of the connection, the next dot appears (or the trial ends).

This task offers a good abstraction of many typical interactions in today's interfaces (such as making an area selection on a canvas or creating a new object in a CAD program) but is simple enough to be amenable to accurate measurement. While it does not cover all possible interaction styles (e.g., it is ill-fitted to characterize freeform drawing applications, see Section 6.1.1), we feel that it covers a large enough class of applications to provide useful insights.

We used a within-subjects design, asking each participant to connect a series of colored dots using all four techniques in turn. The independent variable was the method used to connect dots. The dependent variable was the total task completion time to a correctly colored connection, a good representation of the overall performance, since this measure factors in not only the time taken to perform the task but also the time taken to correct errors.

4.1 Participants

Twelve right-handed, noncolorblind (by self-report) participants (7 men and 5 women) were recruited from a young adult population (18 to 36 years of age). All participants had little or no experience using a pen interface other than on a PDA. In addition, participants had little or no knowledge of control menus, FlowMenu, or Toolglass.

Fig. 3. All conditions used this experimental setting, consisting of a display, the Wacom tablet, its pen, and its puck.

4.2 Experimental Apparatus

We used the setup shown in Figure 3. Interactions were performed on a Wacom Intuos 12″ × 12″ (305mm by 305mm) tablet using the version 1.26 of the WinTab[1] interface package provided by Wacom. This tablet can simultaneously track a pen and a puck. The tablet was used in absolute mode, and both pen and puck shared the same active surface (unified area setting, using the nomenclature of Balakrishnan and Hinckley [1999]). Pilot studies showed that the most comfortable setting mapped the screen area to an area 229mm by 171mm, starting at 88mm below and 76mm to the right of the top left corner of the tablet active area. The gain factor between the tablet and the screen was set to 1.33. Finally, to avoid collisions between the pen and the puck, the puck tracking was offset by 32mm. This setting was picked for best Toolglass performance according to Balakrishnan and Hinckley [1999]. This configuration was selected as an alternative to the configuration (a mouse and a trackball) of Kabbash et al. [1994] in an effort to extend the scope of applicability of the Kabbash et al. results.

The experimental software ran in Windows 2000 on a Dell Precision workstation 610 MT with a single Pentium III (550MHz) and 256MB of memory, using a Nvidia GeForce2 as a display card. The workstation was connected to a Dell UltraScan 1000HS series 17″ monitor with a visible area of 305mm by 229mm, running at a resolution of 1024 × 768. The software logged all interactions performed by users at a sampling rate of about 100Hz. Time measurements were made using the real-time clock provided by the operating system. To limit timing errors, the workstation was disconnected from the network during the experiment and logging data was only committed to disk between sets. To verify the accuracy of our timing method, we also compared the timing provided by

[1]Available at http://www.pointing.com/Wintab.html.

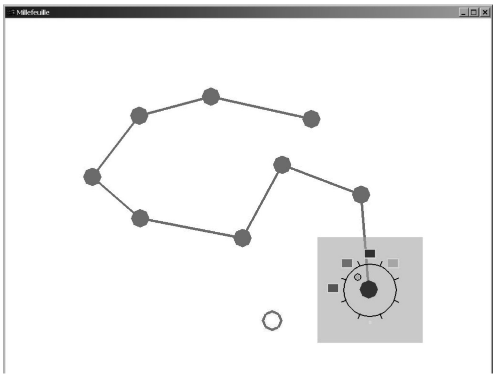


Fig. 4. A typical display for our experiment, shown here with the control menu condition. Previously connected dots were shown in gray, while the current dot is shown in black and the target dot is shown open. The 4 possible colors are: red, green, blue, and yellow. The gray background has been removed for clarity.

our program to the timing provided by counting fields in video footage of some pilot experiments. In all cases, the results were in accord.

4.3 Task and Setting

For all conditions, participants were presented with the same 24 sets of 12 points to connect (11 connections per set). For each set, the computer presented the series of colored dots one by one. Participants were instructed to connect the previous dot to the next dot after selecting the correct color using the control mechanism. They were told to do it as quickly as possible. It was also made clear to the participants that errors would decrease their speed because the interaction was to be repeated until it was successfully completed. New dots were presented as soon as the participant successfully connected the active dot. Consecutive dots were always of different colors as in Kabbash et al. [1994]. The connection time was measured from the appearance of a new dot to successful completion of the line, including time to correct any errors in picking the color or connecting the dots.

To encourage participants to perform at a constant level of performance, after each set, participants were presented with their aggregate time for the completed set and their best time so far for the current experimental condition. If the best time was improved, a rewarding sound was played. Participants could only rest between sets. All conditions were run with users interacting with a pen (and a puck for the Toolglass) on a digital tablet while looking at a monitor (indirect setting, Figure 3). We did not test any configurations with direct interaction (such as a touch screen or stylus on a tablet computer). Future work will be required to determine if the results hold for other configurations.

The screen layout is shown in Figure 4. The path created so far is rendered in gray with the exception of the last dot on the path which is rendered in black[2]. All previous dots in the path are rendered filled. The new target dot is

[2]This differs from the setup used by Kabbash et al. [1994] where the created path was rendered in black. Our setting made the display easier for participants to parse.

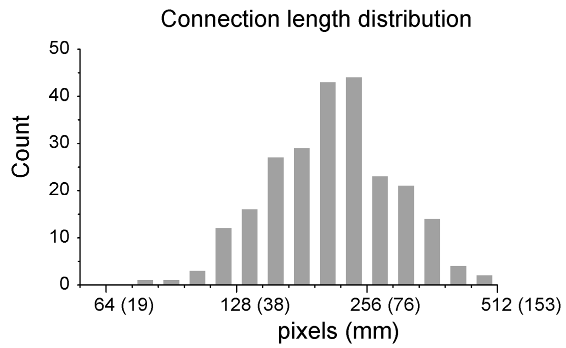


Fig. 5. The connection length distribution over the 24 sets used in the experiment. The same 24 sets were used for all conditions. The data set is identical to the one used by Kabbash et al. [1994]. Only connections for which times were actually measured are reported.

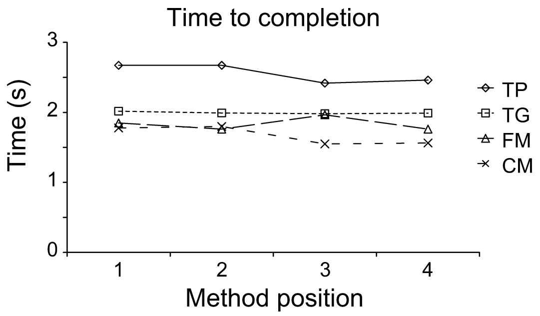
rendered as a circle of the requested color. As soon as a line is started, a line of the selected color is shown on the screen as feedback for the rubber-band interaction. Each dot radius is 11mm. To improve comparability, participants were given the same dot patterns used by Kabbash et al. [1994]. In this set of 24 patterns, the distance between dots varies between 25 and 151mm with the distance distribution shown in Figure 5.

4.3.1 *Tool Palette and Toolglass.* In the tool palette condition, the color tool palette consisted of 4 buttons, each 16mm by 16mm, with a header 32mm wide and 8mm tall at the top, which the participant could use as a handle to move the tool palette. To perform a connection using the tool palette, the participant had to first select the correct color by pressing the tip of the pen and releasing it on the appropriate color button, then pressing the tip of the pen on the last dot of the path, and then draging on top of the new colored target dot.

In the Toolglass condition, the color Toolglass consisted of 4 buttons, each 16mm by 16mm, with a header 32mm wide and 8mm tall at the top. The Toolglass was set to 40% transparency so that the dots were clearly visible underneath it. The Toolglass was moved with a puck. To perform the task using Toolglass, the participant first had to bring the correct Toolglass color area on top of the last dot in the path using the puck, then press the tip of the pen on the last dot of the path, and then dragging on top of the new colored target dot.

In both of these conditions, dots were successfully connected when the pen was lifted from the tablet on top of the target dot.

4.3.2 *FlowMenu and the Control Menu.* In the control menu and FlowMenu conditions, the radius of the menu was 30mm. To perform the task using a control menu or FlowMenu, the participant had to first invoke the menu on top of the last dot of the path by pressing the pen's command button while pressing the pen tip on top of the dot. The menu appeared as soon as the command button was pressed (novice mode), and the command button could be released as soon as the menu appeared. Then the participant had to select a color by either leaving the rest area through the appropriate color's octant (control menu), or leaving the rest area through the appropriate color's octant and then



1	TG	CM	TP	FM
2	CM	FM	TG	TP
3	FM	TP	CM	TG
4	TP	TG	FM	CM

Fig. 6. Skill transfer analysis. For each technique, the graph presents the average task completion time depending on the technique position during each test. Differences are not statistically significant. For reference, the Latin square we used is presented to the right (TP: Tool Palette, TG: Toolglass, FM: FlowMenu, CM: Control Menu).

reentering the rest area (FlowMenu). Finally, the participant had to proceed with the rubber-band interaction to connect to the new colored target dot.

In both of these conditions, dots were successfully connected when the pen was lifted from the tablet on top of the target dot or if the command button was pressed while on top of the target dot to start issuing the next command (as described in Guimbretière and Winograd [2000]).

4.4 Protocol

After a brief description of the experiment, participants were familiarized with the operation of the testing apparatus. For each condition, the correct way to perform the task was explained to the participant, and each participant was given the opportunity to practice on 5 sets of 12 dots that were not among the 24 sets used in the experiment. The order of the experimental conditions for each participant was counterbalanced using a Latin square control for order effects. The color layouts of the control menu and FlowMenu were arranged in different order, as were those of Toolglass and the tool palette, in an effort to limit carryover effects.

After completing all trials, participants completed a questionnaire giving subjective ratings to different aspects of each technique on a scale from 1 (worst) to 7 (best) and providing information about their previous experience with similar systems. The total time for the experiment was about 1.5 hours.

5. RESULTS

All participants completed all sets, typically spending around 22 seconds per set. As in Kabbash et al. [1994], the first connection in each set was removed from the data. As a result, we recorded 240 connections in each of the four conditions for each user. Statistical analyses were performed on averages within each condition and for each user.

To check for skill transfer, we examined the interaction between technique and order [Poulton 1966]. The corresponding data is shown in Figure 6. As expected, the graph shows an overall improvement with position, consistent

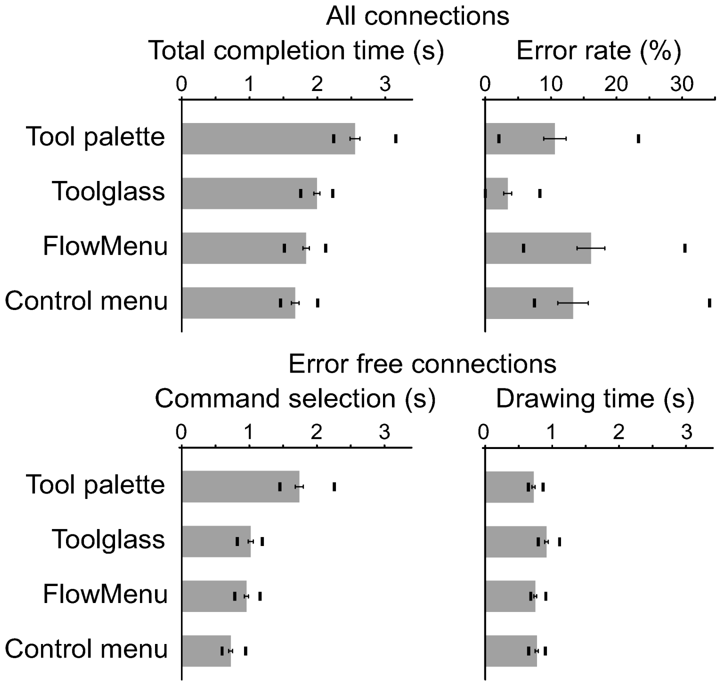


Fig. 7. Quantitative results. For each technique, we show the mean total connection time, the error rate, the mean command selection time (the time elapsed between the start of the trial and the time at which the drawing starts) for connections without errors, and the mean drawing time for connections without errors. Besides mean values and the standard error, the maximum and minimum values for each series are shown. Data used to draw these graphs are tabulated in Table II.

with symmetrical skill transfer. The graph also shows an anomaly when the FlowMenu technique was performed as the third condition or, in our case, just after the Toolglass technique. These anomalies might suggest a possible asymmetric skill transfer between these two conditions. However, an ANOVA determined that these interactions between Technique and Order ($F(9, 32) = .75$, $p = .66$) were not significant.

A box plot showed participant P9 (who used a double click to invoke the control menu) as an outlier (value higher than 1.5*IQR above the upper hinge) for error rate in the control menu condition and subjective enjoyability in the control menu condition, so all P9 data points were removed from those two analyses. All significance levels for pairwise comparison use a Bonferonni correction for multiple comparisons and p values reflect this correction. Results are shown in Figure 7 and Figure 9 with the numerical data tabulated in Table II.

5.1 Task Time

A repeated-measures ANOVA determined that means for total completion times were significantly different across experimental conditions ($F(3, 33) = 73.4$, $p < .0005$). As shown in Figure 7, the tool palette was significantly slower than Toolglass ($p < .0001$). While the relative improvement we observed is

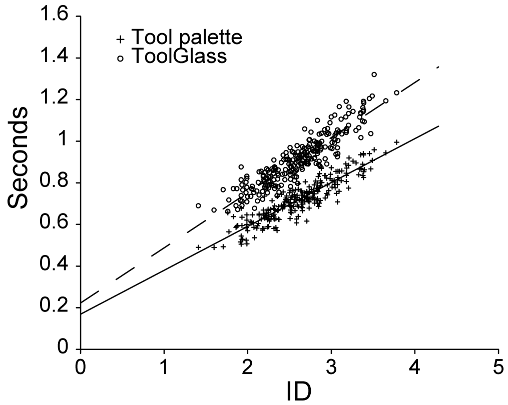


Fig. 8. Relationship between the indices of difficulty and the average drawing time (across users) for the 240 connections of our set. ID was computed as: ID = $\log_2$(D/W + .5) [Welford 1971].

slightly greater than the one reported by Kabbash et al. [1994] (22% compared to the 16% they reported), this result replicates Kabbash's findings.

More surprisingly, our results show that both FlowMenu and the control menu were significantly faster than Toolglass (p = .01, and p < .0005, respectively). This is somewhat surprising given the extended body of work supporting the superiority of two-handed techniques over one-handed techniques (see Leganchuk et al. [1998] for an overview). To understand this finding, it is interesting to examine connections performed without errors in more detail. In the bottom half of Figure 7, we present the command selection time (the time elapsed between the start of the trial and the time at which the drawing starts) and the drawing time for error-free connections. For both variables, a repeated-measures ANOVA determined that mean times differed significantly across conditions, ($F(3, 33) = 194$, $p < .0005$ and $F(3, 33) = 17.8$, $p < .0005$, respectively).

As expected, the tool palette had the slowest command selection time ($p < .0001$), FlowMenu and Toolglass had similar command selection times, and FlowMenu was slower than the control menu ($p < .0005$), probably because it uses a more complex selection gesture. With regard to drawing time, Toolglass was slower than the other techniques ($p < .005$). To explain this difference, we will focus on the difference between Toolglass and the tool palette since their drawing tasks are almost identical. We plotted the average connection time for all users against the index of difficulty (ID = $\log_2$(D/W + .5) [Welford 1971]) for our 240 connections. The result is plotted in Figure 8. As expected, there is a good linear correlation between ID and the time it takes to perform the connections ($r^2 = .83$ for Toolglass and $r^2 = .82$ for tool palette), but the slopes for each population are significantly different ($t_{\alpha(2),476} = 35.26$, $p < .0005$) with Toolglass connections being more difficult to handle. The most commonly observed way users carried out the task with the Toolglass was by moving the Toolglass at the same time as a connection was being made. Thus, it seems that moving the Toolglass with the left hand slows down the tracing part of the task, possibly because the user needs to attend to two tasks at the same time. Contrary to the interpretation of Kabbash et al. [1994], our results suggest that Toolglass's advantage over standard techniques might not come from its two-handedness but from its merging of command selection and direct manipulation.

Finally, we also compared the performance of FlowMenu and control menu and found that control menu was faster but that the difference was not significant ($p = .2$). This result has to be interpreted carefully since the skill transfer analysis presented in the beginning of Section 5 suggests that an interaction might exist between the Toolglass and FlowMenu when they are performed back-to-back. This interaction might artificially increase the difference between control menu and FlowMenu. To understand the possible effect of this interaction, we ran our statistical analysis removing the three participants (4, 8, and 12) for whom FlowMenu was used after Toolglass. For the group that remained, a repeated-measures ANOVA determined that means for total completion times were significantly different across techniques ($F(3, 24) = 45.1$, $p < .0005$, see Table II for numerical values) and that the difference between FlowMenu and control menu was not significant ($p > .99$). These results further suggest that the performance of the two systems might be similar. Further studies will be necessary to examine any differences between these two techniques.

5.2 Error Rates

For the purpose of this analysis, a connection was considered error-free if a participant selected the correct color and performed the connection on the first trial. Due to a limitation in our software, the system did not recognize starting a connection outside of the starting dot as an error. A repeated-measures ANOVA determined that means for error rates were significantly different ($F(3, 30) = 13.2$, $p < .0005$, P9 removed) across techniques. Toolglass was significantly less error prone than any other technique ($p = .01$). One participant was even able to connect all 240 points without any errors using Toolglass. No other differences were significant.

5.3 Subjective Ratings

The questionnaire used to collect subjective ratings asked, for each technique in turn, whether the technique was fast, enjoyable, error prone, and comfortable to use (Figure 9). While a repeated-measures ANOVA determined that means for the *fast* and *error prone* subjective variables differed significantly across techniques ($F(3, 33) = 11.1$, $p < .0005$, $F(3, 33) = 4.8$, $p < .01$, respectively), the only significant pairwise comparisons were found for the fast variable. Overall, subjective ratings for these two variables were in accord with the measured speed and error rate. More precisely, both FlowMenu and the control menu were perceived as being faster than the tool palette ($p < .01$), but only the control menu was perceived as being faster than Toolglass ($p < .05$).

We also recorded the participants' subjective enjoyment and level of comfort for each technique but we did not find any significant differences.

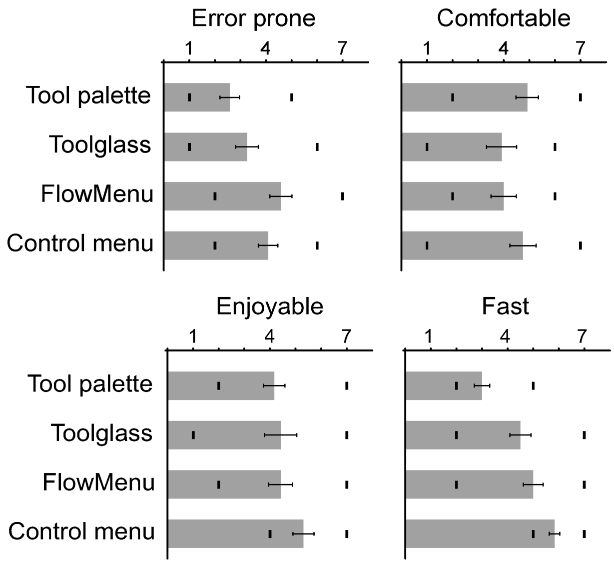


Fig. 9. Subjective ratings. Data were collected using a scale from 1 (Not at all) to 7 (Very much). Besides mean values and the standard error, the maximum and minimum values for each series are shown. Data used to draw this graphs are tabulated in Table II.

Table I.
(Techniques used by Kabbash et al. [1994] and their reported times. In this table, we are using the Kabbash et al. notation. In particular, the tool palette is called R-tearoff by Kabbash et al.)

	Command Selection and Direct Manipulation	
Interaction	Not Merged	Merged
One handed	R-tearoff (2.89s)	
Two-handed	L-tearoff (2.96s) Palette (2.90s)	Toolglass (2.43s)

6. DISCUSSION

The results presented highlight the importance of techniques that can fluidly mix command selection and direct manipulation. Our results suggest that the main advantage of the two-handed Toolglass comes from the fact that it lets the user smoothly merge command selection and direct manipulation.

It is important to note that our findings are consistent with the results reported by Kabbash et al. [1994]. As shown in Table I, it is clear that, although Kabbash tested three two-handed techniques, he observed a significant performance increase only when the technique merged command selection and direct manipulation (Toolglass). These observations reinforce our belief that merging command selection and direct manipulation is the key aspect of Toolglass's superior performance over the standard tool palette technique.

Our study did not find the expected difference between FlowMenu and control menus. We believe that a separate study and a finer-grained analysis (perhaps using the steering law proposed by Accot and Zhai [2002]) will be needed to determine if there is any performance difference between FlowMenu and control menus.

6.1 Interaction Design Considerations

Our work suggests that control menus and FlowMenu might be faster than Toolglass for tasks such as area selection and vector drawing in CAD and illustration programs. This is particularly important because these techniques do not require the special hardware and system changes needed to implement two-handed interaction. Nevertheless, speed is not the only criterion for designing a user interface. Therefore, it is important to understand some of the fundamental differences between these techniques.

6.1.1 *Versatility.* Control menus, FlowMenu, and Toolglass might be functionally equivalent in many situations such as for drawing geometric shapes or for selecting and transforming objects. However, if the application requires freeform drawing, then Toolglass has a distinctive advantage because the Toolglass's see-through metaphor allows the user to start the interaction at the point where the command was invoked. Neither control menus nor FlowMenu provide this flexibility because they require the user to cross a specific boundary away from the point where the menu was invoked. Note that pie menus [Hopkins 1991] and the extension of marking menus proposed by Kurtenbach and Buxton [1991] would face a similar limitation since the selection stroke has to be part of the drawing or the drawing has to be started at some point away from the initial location where the menu was called.

6.1.2 *Design Considerations.* Our results open new avenues in the design of efficient interfaces in situations where two-handed interactions are not practical. For example, for a PDA or Tablet computer, it might be difficult to use both hands at the same time or too expensive to install two tracking devices. Another area that might benefit from our finding is that of interaction design for large interactive surfaces [Guimbretière et al. 2001]. Large interactive surfaces are becoming more prevalent and are designed with a direct interaction mode in mind. For those purposes, we believe that a two-handed interface may still have benefits. However, instead of using the nondominant hand to help select the current command as with Toolglass, one might use it to set the frame of reference for the work or to manipulate tools as in T3 [Kurtenbach et al. 1997]. Guiard [1987] has advocated this setting and our results suggest that this approach might be fruitful.

6.1.3 *Scalability.* Another important point in selecting command mechanisms is scalability. While it is true that the basic operation in our connect-the-dots task resembles a variety of different direct manipulation actions, users often perform similar operations in succession. It is interesting to see how the cost of performing a connection evolves if several points of the same color are presented in succession. In this case, the amortized connection time (CT) can be expressed as a function of the number n of successive points with the same color, the average color selection time (CST) for a technique, and the average

Table II. Means (and Standard Errors) for Each Variable in Our Data Set

	Tool palette	Toolglass	FlowMenu	Control menu
Total connection time (s)	2.56 (0.073)	1.99 (0.044)	1.83 (0.049)	1.67 (0.057)
Total connection time without participants 4, 8, 12 (s)	2.52 (0.090)	2.00 (0.049)	1.79 (0.052)	1.71 (0.069)
Error rate (%)	10.6 (1.7)	3.41 (0.6)	16.1 (2.1)	13.4 (2.3)
Command time for error-free connections (s)	1.74 (0.060)	1.02 (0.037)	0.955 (0.031)	0.724 (0.029)
Drawing time for error-free connections (s)	0.720 (0.022)	0.914 (0.028)	0.747 (0.020)	0.769 (0.023)
Enjoyable (1 [Not at all] – 7 [Very much])?	4.2 (0.42)	4.4 (0.63)	4.4 (0.47)	5.3 (0.40)
Fast (1 [Not at all] – 7 [Very fast])?	3.0 (0.30)	4.5 (0.42)	5.0 (0.39)	5.8 (0.21)
Error prone (1 [Not at all] – 7 [Very much])?	2.6 (0.38)	3.3 (0.45)	4.6 (0.43)	4.1 (0.38)
Comfortable (1 [Not at all] – 7 [Very comfortable])?	4.9 (0.43)	3.9 (0.58)	4.0 (0.49)	4.8 (0.51)

drawing time (DT) as

$$CT = CST + DT \text{(for control menu, FlowMenu and Toolglass)}, \quad (1)$$

$$CT = \frac{CST}{n} + DT \text{(for the tool palette since the color is only selected once)}. \quad (2)$$

Equating (1) and (2) and using numerical values for CST and DT provided in Table II, one can see that performing as few as two operations per tool selection might be enough to make the tool palette faster than Toolglass and FlowMenu, while it would take three to be faster than a control menu. These results assume error-free connections, but, of course, a complete analysis should also take into account that users might be more likely to forget to select a new color after a long streak of dots of one color. This is a general problem with modal interfaces [Raskin 2000]. The relative advantage of a modal interface may also be offset by the fact that, if color positions are stable, both FlowMenu and control menu are subject to learning. As a result, the selection motion might be chunked by expert users [Buxton 1987]. Yet, as shown by Mackay's [2002] study of CPN2000 [Beaudouin-Lafon 2000], some tasks, such as making a copy of a Petri net, naturally lead to high amortization (3.3 commands per command selection for the toolbar). This might help to explain why the toolbar is still such a popular menu technique.

Finally, many applications have far more than four possible commands. All one-handed techniques studied here as well as Toolglass can easily be extended to accommodate a larger set of commands (e.g., the T3 system of Kurtenbach et al. [1997]). The analysis of such systems is made more difficult when dealing with large command sets because of possible confounding effects, such as having to remember how to access a particular command. Future research needs to address these issues.

7. FUTURE WORK

There are always obstacles to applying results from an experimental setting to predict performance in more applied situations. In our case, we were unable to model how users amortize the cost of command selections with a palette by performing several successive actions with one tool. To solve this problem, we are exploring methods to gather quantitative data of real usage behavior for tool palettes in drawing and CAD tools.

We would also like to broaden our understanding of how two-handed techniques can improve other operations. Recent results [Bourgeois and Guiard

2002] show the benefits of two-handed interaction for zooming and panning. Using techniques such as speed-dependent zooming [Igarashi and Hinckley 2000], we would like to see if similar performance can be obtained using only one hand. This will free up the nondominant hand which could be used for other purposes.

Finally, we would like to explore the new design avenues opened by our results. In particular, we would like to pursue interface designs that remove the current limitations of control menus and FlowMenu for freehand applications and propose a new style of two-handed interaction that lets the nondominant hand orient the work while the dominant hand performs the work in the frame of reference set by the nondominant hand.

8. CONCLUSION

In this article, we have presented new evidence for the benefits of mixing command selection and direct manipulation in commonly-performed direct manipulation tasks of modern interfaces. Our results show that these benefits can be obtained not only by using two-handed interaction techniques such as Toolglass, but also by using one-handed techniques such as control menus and FlowMenu. Our analysis of these results should aid in the understanding of the advantages of two-handed command selection and in weighing them against possible drawbacks such as more complex hardware. We have also highlighted the need for further analysis to better understand the real advantage of these techniques for actual patterns of use as observed in large industrial-strength applications.

ACKNOWLEDGMENTS

The authors would like to thank Ken Hinckley, Corinna Löckenhoff, Ben Bederson, Catherine Plaisant, and Juan-Pablo Hourcade for their helpful comments on drafts of this article. We would also like to thank Gordon Kurtenbach for his advice on the initial experimental design and Paul Kabbash for graciously providing us with the dataset he used in his experiment. Finally, we would like to thank our reviewers for their help in improving this article.

REFERENCES

ACCOT, J. AND ZHAI, S. 2002. More than dotting the i's—Foundations for crossing-based interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'02)*. Minneapolis, MN. 73–80.

BALAKRISHNAN, R. AND HINCKLEY, K. 1999. The role of kinesthetic reference frames in two-handed input performance. In *Proceedings of the 12th Annual ACM Symposium on User Interface Software and Technology (UIST'99)*. Asheville, NC. 171–178.

BEAUDOUIN-LAFON, M. 2000. Instrumental interaction: An interaction model for designing post-WIMP user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'00)*. The Hague, The Netherlands. 446–453.

BIER, E. A., STONE, M. C., PIER, K., BUXTON, W., AND DEROSE, T. D. 1993. Toolglass and magic lenses: The see-through interface. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'93)*. Anaheim, CA. 73–80.

BOURGEOIS, F. AND GUIARD, Y. 2002. Multiscale pointing: Facilitating pan-zoom coordination. In *CHI'02 Extended Abstracts on Human Factors in Computing Systems*. Minneapolis, MN. 758–759.

BUXTON, B. 1986. Chunking and phrasing and the design of human-computer dialogues. In *Proceedings of the IFIP World Computer Congress*. Dublin, Ireland. 475–480.

DILLON, R. F., EDAY, J. D., AND TOMBAUGH, J. W. 1990. Measuring the true cost of command selection: Techniques and results. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Seattle, WA. 19–25.

GUIARD, Y. 1987. Asymmetric division of labor in human skilled bimanual action: The kinematic chain as a model. *J. Motor Behav. 19*, 4, 486–517.

GUIMBRETIÈRE, F. AND WINOGRAD, T. 2000. FlowMenu: Combining command, text, and parameter entry. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST'00)*. San Diego, CA. 213–216.

GUIMBRETIÈRE, F., STONE, M., AND WINOGRAD, T. 2001. Fluid interaction with high-resolution wall-size displays. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST'01)*. Orlando, FL. 21–30.

HOPKINS, D. 1991. The design and implementation of PieMenus. *Dr. Dobb's J. 16*, 12, 16–26.

IGARASHI, T. AND HINCKLEY, K. 2000. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST'00)*. San Diego, CA. 139–148.

KABBASH, P., BUXTON, W., AND SELLEN, A. 1994. Two-handed input in a compound task. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'04)*. Boston, MA. 417–423.

KURTENBACH, G. 1993. The design and evaluation of markingMenus. PhD thesis, University of Toronto.

KURTENBACH, G. AND BUXTON, W. 1991. Integrating mark-up and direct manipulation techniques. In *Proceedings of the 4th Annual ACM Symposium on User Interface Software and Technology (UIST'91)*. Hilton Head, SC. 137–144.

KURTENBACH, G., FITZMAURICE, G., BAUDEL, T., AND BUXTON, W. 1997. The design of a GUI paradigm based on tablets, two-hands, and transparency. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'97)*. Atlanta, GA. 35–42.

LEGANCHUK, A., ZHAI, S., AND BUXTON, W. 1998. Manual and Cognitive benefits of two-handed input: An experimental study. *Trans. Hum.-Comput. Interact. 5*, 4, 326–359.

MACKAY, W. E. 2002. Which interaction technique works when? floating palettes, marking menus and toolglasses support different task strategies. In *Proceedings of Advanced Visual Interfaces (AVI'02)*. Trento, Italy. 203–208.

MCGUFFIN, M., BURTNYK, N., AND KURTENBACH, G. 2002. FaST sliders: Integrating marking menus and the adjustment of continuous values. In *Proceedings of Graphics Interface (GI'02)*. Calgary, Alberta, Canada. 35–42.

POOK, S., LECOLINET, E., VAYSSEIX, G., AND BARILLOT, E. 2000. Control menu: Execution and control in a single interactor. In *CHI'00 Extended Abstracts on Human Factors in Computing Systems*. The Hague, The Netherlands. 263–264.

POULTON, E. C. 1966. Unwanted asymmetrical transfer effects with balanced experimental designs. *Psych. Bull. 66*, 1, 1–8.

RASKIN, J. 2000. *The Humane Interface: New Directions for Designing Interactive Systems*. Addison-Wesley Professional Publishing. 33–70.

WELFORD, A. T. 1971. *Fundamentals of Skill*. Methuen, London, UK.

Received April 2004; revised September 2004; accepted October 2004 by Robert Jacob