The Design and Evaluation of Multitouch Marking Menus

G. Julian Lepinski[1], Tovi Grossman[2], George Fitzmaurice[2]
Autodesk Research 210 King St. East, Toronto, Ontario, Canada, M5A 1J7
[1]{lastname}@dgp.toronto.edu [2]{firstname.lastname}@autodesk.com

ABSTRACT
Despite the considerable quantity of research directed towards multitouch technologies, a set of standardized UI components have not been developed. Menu systems provide a particular challenge, as traditional GUI menus require a level of pointing precision inappropriate for direct finger input. Marking menus are a promising alternative, but have yet to be investigated or adapted for use within multitouch systems. In this paper, we first investigate the human capabilities for performing *directional chording gestures*, to assess the feasibility of multitouch marking menus. Based on the positive results collected from this study, and in particular, high angular accuracy, we discuss our new multitouch marking menu design, which can increase the number of items in a menu, and eliminate a level of depth. A second experiment showed that multitouch marking menus perform significantly faster than traditional hierarchal marking menus, reducing acquisition times in both novice and expert usage modalities.

Author Keywords
Multi-finger input, multi-touch displays, marking menus.

ACM Classification Keywords
H.5.2 [User Interfaces]: Interaction styles.

General Terms
Design, Experimentation, Human Factors.

INTRODUCTION

Multitouch interfaces are characterized as computing interfaces that are touch sensitive, and allow a user to issue multiple touches simultaneously [9, 11, 24]. Multitouch interfaces have become popular in recent years, both in research and in consumer devices. Despite the considerable quantity of research directed towards multitouch technologies, a set of standardized UI components have not been developed [31]. In particular, menu systems specifically designed for multitouch platforms have received little attention.

In traditional desktop computing, menu systems are used to navigate to and access commands. Menus should provide a logical categorization, support efficient access, and save screen real estate for the user's primary work [7].

While traditional menus are widespread in conventional computing interfaces, they are not necessarily well-suited for multitouch displays. Because the activation area of a finger is considerably larger than that of a mouse cursor, touch interfaces often drastically increase the minimum size of widgets needed for accurate selection [4]. For a menu with many items, this may simply require more screen real estate than is available or result in large, visually displeasing menus.

While recent research [4, 23, 27] has presented methods to refine the gross movement of traditional multitouch into finer movements that may be suitable to select from smaller, traditional menu systems, we believe these methods may not provide the same level of performance as new menu systems tailor-made for a multitouch environment.

Another type of menu systems which may be more appropriate for the direct finger input of multitouch systems is marking menus [15]. Marking menus save screen real estate, by only popping-up when being used, require directional accuracy instead of positional accuracy, and support gestural activation which is a desirable mode of interaction to support a "natural" [14] user interface experience. However, a marking menu design, which takes advantage of multitouch input, has not been investigated. Marking menus are a well studied menu design outside the multitouch domain, and their performance gives good reason to examine how they may be applied to multitouch.

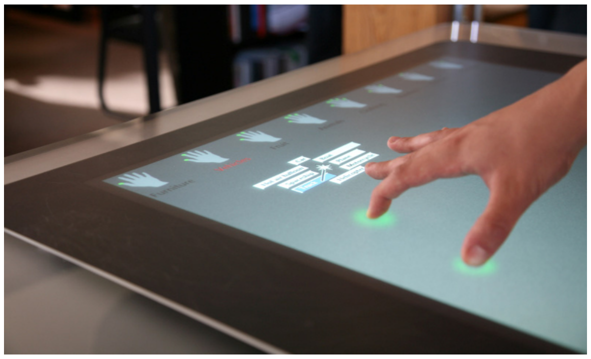
Figure 1. The Multitouch Marking Menu system in use on a Microsoft Surface. The green contact points have been added for illustration purposes.

In this paper, we design and evaluate a new multitouch marking menu system (Figure 1). Our design is guided by an initial empirical experiment, which collects data on users' ability to perform *directional chording gestures*. This initial experiment finds that users can accurately gesture in 8 directions while articulating multifinger chords. Based on this data, we propose our new multitouch marking menu design, which uses chorded input to quickly indicate the top-level menu category. In addition, we outline a new recognition technique to disambiguate between the chords. In a second experiment, we find that our new design outperforms traditional single touch marking menus, when selecting from a set of 64 possible menu items. This result was found for both novice and expert usage modalities.

RELATED WORK

In this section we review the work which relates to our development of a multitouch marking menu technique. In particular we provide an overview of the research done on marking menus, multitouch systems, and chording input.

Marking Menus

Menu Design

Marking menus [15] are a gesture-based menu system which displays menu items in a radial layout around the cursor. The user drags the cursor, or "marks", in the direction of the desired item. In hierarchal marking menus, a mark can select a particular path through a menu hierarchy. Marking menus support both novice and expert modes. In novice mode, after a short dwell time, the menu is displayed and the user can move towards the desired item. In expert mode, the location of the desired item is already known, and the user can quickly perform the marking gesture without waiting for the menu to be displayed.

Numerous alternative designs have been proposed to improve upon the initial designs, such as the hotbox [17], multi-stroke menus [36], zone and polygon menus [36], and flower menus [2]. However, we are unaware of work which applies multitouch input to the design of marking menus.

Empirical Evaluations

One of the most important aspects of marking menus is determining how many menu items should be placed at each level. The angular accuracy which the user can achieve will determine appropriate menu breadth. As the breadth increases, more items are placed at each level, reducing the angle between each item.

Kurtenbach and Buxton [15] recommended that to maintain acceptable levels of accuracy, the breadth of the menu should be 8 with a depth of 2 levels. With this design, the menu can contain 64 items. Zhao and Balakrishnan's multi-stroke marking menus [35] allowed for an additional level of the menu, but the menu breadth was still limited to 8.

These results have been found for mouse and pen input [16]. While hand input has been used to control marking menus, both in free space [18] and on tabletop displays [34], we are unaware of work which investigates the angular accuracy of multi-finger marking menu usage. Our work provides this important empirical data.

Multitouch Systems

Multitouch Interaction

One of the appeals of multitouch interaction is its potential to provide a richer set of inputs to interactive systems. In general, this input is used to manipulate the data itself through "natural" [14] gestures [9, 34], but it can also be used for interface controls, such as command activation [5, 34] and cursor control [10, 22].

In many cases, command activation in multitouch systems can be achieved through gesture sets [9, 34]. While gestures can provide a fun interaction experience, recent research has shown that it may be more challenging than once believed to provide "natural" gestures that user's will be able to immediately learn [12, 33]. Furthermore, while gestures may be practical for a limited set of commands, it would be impractical to develop a library of gestures for applications which possess hundreds of commands. For such a scenario, it would seem to be more practical to use a menu system.

While menus have been developed and used for multitouch systems [5, 34], few actually take advantage of the multiple finger input stream which is available. Wu describes a tool which allows the user to choose a radial menu item with one finger, and then place the item with a second finger [34]. Brandl describes a menu which activates a menu with one hand, and selects the items with the other [6]. Neither of these designs attempt improvements upon traditional marking menus. In our work we integrate multifinger chords into marking menus and provide an empirical comparison to traditional, single point marking menus.

Multitouch Technologies

Numerous technologies have been used for sensing hand and multi-finger input, such as capacitive sensing [9, 24] and vision-based solutions [11, 21, 32]. Each of these technologies provides slightly different input streams, making the design of technology-independent interactions challenging. For example, the SmartSkin system can sense finger positions while they are above the surface [24]. In contrast, FTIR systems only sense contact points [11], but has the ability to also sense pressure [8]. To reduce dependency on a specific hardware platform, our designs do not rely on these additional input streams.

Chording Input

Chorded input involves the simultaneous use of multiple fingers. Chording input has been most commonly seen in the text-entry literature, where it has been shown to significantly increase typing performance [20, 30]. By using different combinations of fingers, the number of possible characters is increased

Previous gesture sets for multitouch technologies have taken into account the *number* of contact points as a mechanism to specify gestures, such as using one finger to rotate and two fingers to scale [12]. Less work has gone into using different *combinations* of fingers [11], which can increase the command vocabulary. A barrier to the use of such chords is the absence of finger identification technology. We describe a simple vision based solution to allow for accurate sensing of an increased number of chords.

DIRECTIONAL CHORDING GESTURES

Recognizing the potential power of using different combinations of fingers as a method of command input, we believe that menus could be efficiently operated through what we define as *Directional Chording Gestures*.

Since chording in text entry occurs on physical devices, the chords cannot be directional, and are only used to press a button (Figure 2a). However, applied to an interactive multitouch surface, chords could also provide directional information (Figure 2b). Applying this principle can provide a large combination of relatively simple gestures. We apply this idea to develop two types of directional gestures: Simple-Stroke and Lift-and-Stroke gestures.

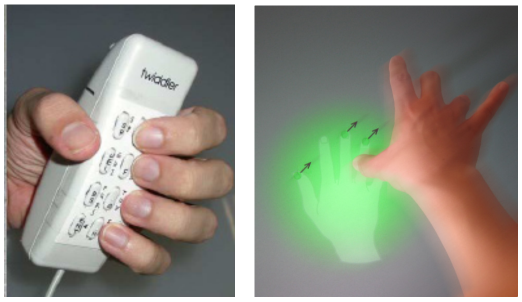

Figure 2. a) A traditional chord used on a physical text input device (Image from [20]). b) Chords on a multitouch device can be directional.

Simple-Stroke Gestures

The chords are initiated by using the various combinations of the 5 fingers on one hand. In total, there are 31 different combinations: 5 single finger chords (one for each finger), 10 two-finger chords, 10 three-finger chords, 5 four-finger chords, and 1 five-finger chord (Figure 3).

To create our *Simple-Stroke Gestures*, we combine each of these chords with a direction. For example, if we allow 8 possible directions (N, NE, E, SE, S, SW, W, NW), we obtain a set of 8 x 31 = 248 gestures. Figure 4 shows the 8 gestures for one of the chords. This design gives us a large gesture set, without any compound strokes [35], or iconic shapes [3].

Lift-and-Stroke Gestures

A requirement of *Directional Chording Gestures* is that the chord itself must first be recognized. This is a challenge, since multitouch technologies typically do not recognize individual fingers. A system may know two fingers are in contact, but not know which two fingers are in contact.

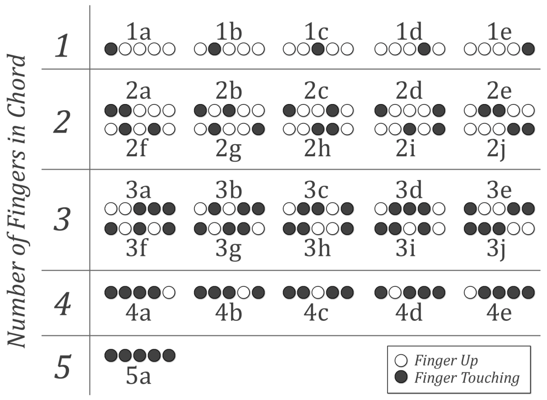

Figure 3. A map of the 31 possible chords. The illustration is for a right hand (thumb is far left, pinky is far right).

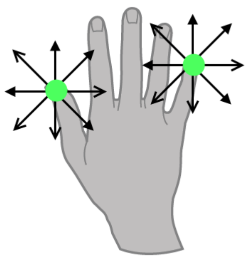

Figure 4. The eight simple stroke gestures for chord [3j]. For any of the directions, the user would move the hand, and both contact points, in parallel.

As a software design solution to this potential limitation, we developed the *Lift-and-Stroke* gestures. These gestures are the same as the Simple-Stroke gestures, except the user is first required to depress all five fingers (Figure 5a). This calibrates the system, so it knows where each finger is located. The user then lifts the fingers *not* required for the gesture (Figure 5b), and then performs the directional stroke (Figure 5c). This solution actually has an additional benefit – the placement of all five fingers could be used to put the system in a command mode, where it would know to accept the menu input. In contrast, before using a Simple-Stroke gesture, the user would first need to enter a command entry mode.

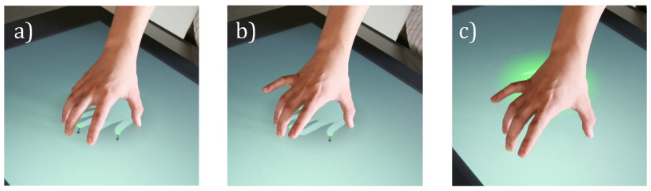

Figure 5. A Lift-and-Stroke gesture for chord [3d]. a) All five fingers are depressed. b) The fingers not required for the chord are released. c) The directional stroke is performed.

EXPERIMENT 1: COLLECTION OF EMPIRICAL DATA

We conducted an experiment to understand how well users can perform directional chording gestures on a multitouch surface. In particular, we wanted to test the effects of different chords (unique combination of fingers) and directions on the angular accuracy and speed of these movements, as well as ascertain which chords are easier to articulate. Furthermore, we wanted to compare the Simple-Stroke gestures to the Lift-And-Stroke gestures. This will be important data to consider when designing our multitouch menus, which we describe after this experiment.

We hypothesized that the directional accuracy and speed of the movements would be affected by both the chord and the direction. Further, we expected there to be a difference in performance and ease of articulation depending on the chord chosen, with 'simple' chords, consisting of fewer fingers, for example, being faster and easier.

Experiment Design

A repeated-measure within participant design was used. The independent variables were *Technique* (Simple-Stroke, Lift-and-Stroke)*, Chord* (1-31)*,* and *direction* (N, NE, E, SE, S, SW, W and NW). Participants performed the experiment in one session lasting approximately 40 minutes. The session was broken up into 62 blocks, 31 of Simple-Stroke and 31 Lift-and-Stroke. Each block consisted of one chord, with all eight directions appearing in a random order. The appearance of the chords was randomized. The *technique* ordering was counterbalanced, with half of the participants starting in Simple-Stroke mode and the other half starting in Lift-and-Stroke mode.

For practice at the start of each experimental mode, participants were given ten randomly chosen chord/direction combinations.

Participants

We recruited ten participants (1 female, 9 male), ranging between 20 and 26 years of age. All participants were right-handed and used their right hand for the study. Participants were between 157cm and 188cm in height and reported no problems with mobility or motion. None of the participants had extensive experience with multitouch systems.

Apparatus

Our experiment was conducted on a Microsoft Surface system which is a combined tabletop multitouch input and rear projected display device. The surface was raised to a height of 86cm, which was a comfortable height for users. The display and interaction area measures 63.5cm by 48cm. The software was implemented in C# and the touch tracking was handled by the multitouch libraries included with the Microsoft Surface SDK.

Procedure

Participants stood in front of the Microsoft Surface in a position where they could comfortably reach the display area. The trial started with a blank screen and a start box at the bottom, which the participant touched to begin a trial.

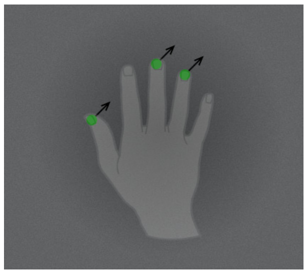

Figure 6. Experiment 1 chord and movement instruction.

Once touching the start box, the chord and direction to perform was displayed in a gesture area (Figure 6), at the center of the screen. The participant then lifted their finger from the start box, moved to this gesture area, and performed the indicated gesture. The gesture area was illustrated with a circle rendered with a gradient fill that faded out at the borders.

For a trial to be considered successful, we required a minimum travel distance of 135 pixels, and an angular accuracy of 45 degrees. We did not perform chord recognition, so the only additional requirement was that the right number of contact points were used. However, an experimenter was present to ensure that users were not "cheating", but performing the correct chord for the trial.

Following a gesture, the screen glowed green or red, indicating whether the trial had been completed correctly or not. In cases where the trial was not completed, the participant repeated the trial, so that we would obtain a complete data set. Users were told to complete the trials as quickly as possible while minimizing errors.

Results

Angular Error

Pilot studies indicated that an accurate way to measure the stroke angle was use the angle between the center-point of the chord when first articulated and the center-point when the first finger was released. Angular error was calculated as the absolute difference between this stroke angle, and the required angle for the trial.

Angular accuracy was very good, with an average angular error of only 5.6 degrees (Figure 7). In total, 98.2% of the gestures were completed with an angular accuracy of less than the 22.5 degrees which would be needed to select between eight radial items.

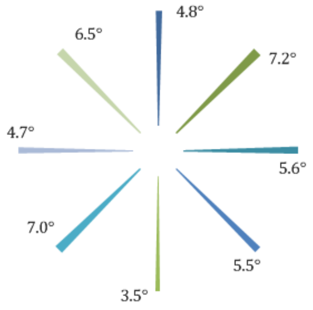

Figure 7. The mean angular error for each direction.

The only variable to have a significant effect on angular error was *Direction* ($F_{7,63} = 6.9$, $p < 0.001$). Our hypothesis that *Chord* would influence angular error was rejected, with a maximal per-chord angular error being 6.4 degrees, and minimal error being 5.0 degrees. This is a positive result, showing that directional chords can be accurately used for command activation.

Trial Completion Time

Trial completion time was measured as the time between lifting the finger from the start box, until lifting the fingers after performing the gesture. Trial completion time was significantly affected by *Technique* ($F_{1,9} = 56.9$, $p < 0.001$), *Chord* ($F_{30,270} = 9.3$, $p < 0.001$), but not *Direction*. To provide a more comprehensive analysis of these results, we divided the trial completion time into two distinct phases: the articulation time, and the movement time.

Articulation Time

Articulation time was defined as the time until the chord was articulated. The *Technique* had a strong effect on articulation time ($F_{1,9} = 84.1$, $p < 0.001$), with average times of 472.9ms for *Simple-Stroke* and 848.4ms for *Lift-And-Stroke* (Figure 8). In general, the act of lifting certain fingers from the surface, while keeping some of the fingers down, was extremely difficult. The *Chord* also had an effect of articulation time ($F_{30,270} = 5.128$, $p < 0.001$). Chords with "gaps", that is, requiring non-consecutive fingers to be depressed, tended to produce higher articulation times. There was also a significant interaction between *Technique* and *Chord* ($F_{30,270} = 4.6$, $p < 0.001$). As seen in Figure 9, there was a greater variation in articulation time for *Lift-and-Stroke*, and less for *Simple-Stroke*.

Movement Time

The movement time was defined as the time taken to perform the stroke, once the chord had been articulated. The movement time was significantly affected by *Chord* ($F_{30,270} = 19.6$, $p < 0.01$), but not *Direction* or *Technique*. In general, chords with more fingers resulted in higher movement times (Figure 9). This was mostly likely due to added friction with more contact points.

Error Rates

Errors were recorded when a trial was not completed successfully on the initial attempt, either because the wrong chord was articulated, or the movement was made in the wrong direction. Error rates were significantly affected by *Technique* ($F_{1,9} = 16.2$, $p < 0.005$), with overall error rates of 16.6% for Simple-Stroke and 24.9% for Lift-and-Stroke. Error rates were also significantly affected by *Chord* ($F_{30,270} = 4.528$, $p < 0.001$), and *Direction* ($F_{7,63} = 2.760$, $p < 0.05$). There were higher error rates for chords involving more fingers. Our observations indicated the errors were often a result of failed tracking by the Microsoft Surface, rather than behavioral errors, and in particular, dropping contact points that had a high velocity. For example, error rates increased linearly with the number of fingers in the chord ($r^2 = 0.95$), with the 5-finger chord (arguably the easiest chord) having the highest error rate ($p < 0.05$, in comparison to 1, 2 and 3 finger chords).

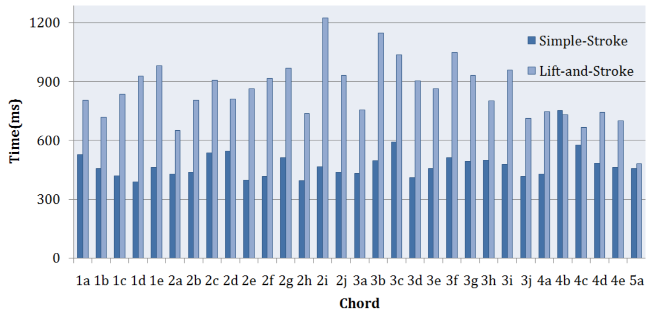

Figure 8. Mean articulation times for the chords, by technique.

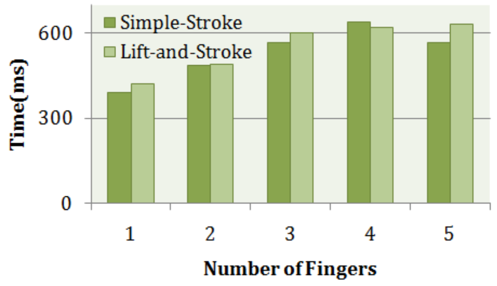

Figure 9. Movement time, by technique and number of fingers.

THE MULTITOUCH MARKING MENU

The data obtained from our first experiment has some important implications to the design of multitouch menus.

On the down side, the Lift-and-Stroke gestures are difficult to perform, and should be avoided. This means that Simple-Stroke Gestures should be used, but a method of recognizing the individual fingers will be needed.

On the upside, we found that the majority of chords were easy to articulate. The exception was chords that had gaps, which tended to produce higher articulation times. Once articulated, the complexity of the chords had negligible effects on the speed and angular accuracy of the directional stroke. This leads us to believe the directional chords, and in particular the Simple-Stroke Gestures, can be used as a command activation mechanism for multitouch systems.

While the Simple-Stroke directional chords seem to be a promising gesture set, we are still in need of a menu system which these gestures can drive. Without a menu system, we only have a large set of unorganized gestures that the user would have to memorize. We place a marking menu system on top of theses gestures, so that they can be structured in an organized fashion, and interactively revealed to the user.

Marking Menu Design Properties

Marking menus have a number of desirable design properties which make them specifically appropriate for use within multitouch applications. In this section, we outline these relevant design properties.

Acceptance of Imprecise Input
Marking menu items are selected through marks, meaning the user never needs to precisely select specific widgets or locations. Instead of positional accuracy, marking menus require angular accuracy. Our first experiment has shown that multitouch input can satisfy these accuracy constraints.

Pop-up Visualization
Marking menus pop-up when activated. This reduces their intrusiveness, as screen real-estate is not taken up, when they are not in use. This is desirable for multitouch applications, since the emphasis should be on the fluid manipulations with the data, and not the UI components.

Scale Independent
The marks required for marking menu selection are scale independent. This means that marking menus can function across a variety of screen sizes. This is important for usage with multitouch technology, as the platforms can vary in size, from large, wall size displays [12], to smaller, personal devices, such as the UnMousePad [25] or iPhone [1].

Location Independent
Because marking menus typically are popped up at any activation point, they can be accessed from any location. This is important for tabletop multitouch systems since a user may be standing anywhere around the display, or multiple users may all want access to the menu system [27].

Gestural Input
Inherent to marking menus are the marks, or gestures, which are used to select menu items. This form of input especially appropriate, as it is consistent with the gesture-based interactions and applications which are being developed for multitouch systems.

Design Specifics
The main idea behind multitouch marking menus is that each level of the menu is accessed with a Simple-Stroke directional chord, instead of a single point mark. The chord indicates a top level menu category, eliminating an entire level of depth from the menu. Here we describe the specific design and implementation details.

Chord Recognition
A limitation of the multitouch devices we are aware of is their inability to disambiguate between different fingers. Our initial hope was that Lift-and-Stroke gestures could be used to calibrate finger locations, eliminating the need for finger recognition, but our initial experiment demonstrated that these gestures are difficult solution.

For simple-strokes, if the technology does not have any capabilities to detect individual fingers, then certain chords will be impossible to disambiguate (for example 2h and 2j in Figure 3). We propose a new recognition technique, which does not provide complete chord disambiguation, but does increase upon the number of chords which could be recognized without any finger disambiguation.

In many vision based multitouch systems, including the Microsoft Surface which we are using, objects in close proximity to the touch screen are visible to the camera. Our solution uses raw images from the tracking cameras to determine a bounding box for the user's hand (Figure 10). We then compare the location of the finger contact points relative to this bounding box. For example, the thumb is almost always at the left edge of the bounding box, and the middle finger is almost always at the top border. This analysis allowed us to effectively increase the number of chords which we could recognize. Although we did not formally study recognition rates, our pilot tests indicated that this strategy could be used to accurately recognize the set of 14 chords illustrated in Figure 11. This set was selected because the chords could be recognized unambiguously based on finger count and their location within the shadowbox.

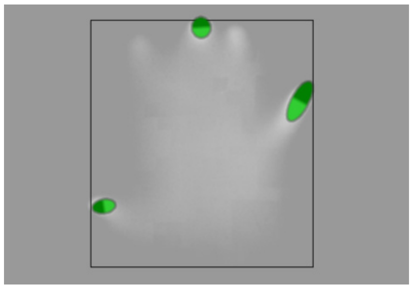

Figure 10. We infer the bounding box for the hand from the raw image captured by the tracking camera.

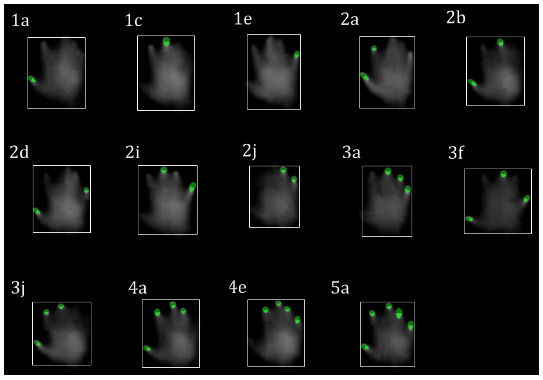

Figure 11. The 14 chords which can be accurately recognized using our shadowbox technique.

Angular Breadth
The angular breadth is the number of directions which are supported by the menu. Previous research has shown that users have difficulty with marking menus with more than eight items displayed, and this has traditionally been the maximum limit of items per menu level [16]. The results of our preliminary study showed that users are also capable of marking with directional chords in eight directions with a very low error rate. As a result, we have retained the maximum limit of eight directions.

Chording Breadth
Because each level of the menu is accessed through a directional chord, the breadth of the menu is also determined by the number of supported chords. Our first experiment showed that some chords are more difficult to articulate (e.g. 4b), but most are acceptable candidates.

The number of chords chosen will depend on the system the menu will be used for, and the number of commands and categories it contains. In our implementation, we used 8 different chords, so that both the chording and angular breadth would be 8. We choose 8 chords which provided strong results in our first experiment, and could be easily recognized with our shadowbox recognition technique. The chord set we settled upon was {1a, 1c, 1e 2a, 2d, 3i, 4a, 5a}. With an angular breadth and chording breadth of 8, our menu system has a total of 64 selectable menu-items.

Menu Depth
The depth of the menu defines the number of levels which the menu contains. Even with a single level, our menu supports 64 items. Since this already replicates the upper limit of what could accurately be achieved with a traditional two-level marking menu [15], we kept the menu depth to 1. This also avoids potential problems caused by including additional menu levels [16]. Thus, all 64 menu items can be accessed through a single, directional chording gesture. We discuss potential designs for increasing menu depths in our future work section.

Menu Item Organization
In traditional marking menus, direction is used to make the final item selection. To maintain this design, our menu organizes items by chord – items in the same category are activated through the same chord. The selection of the item within a category is done through the direction of the chording gesture. As such, the interaction model is to articulate a chord to select a category, and then perform a directional gesture with that chord to select an item.

The Chord Map
We display a chord map, illustrating the eight chord patterns, at the top of the screen relative to the user's location. This chord map could be displayed when the system enters its command mode, so it does not occlude the display during application usage. The chord map shows what category is associated with each chord (Figure 12).

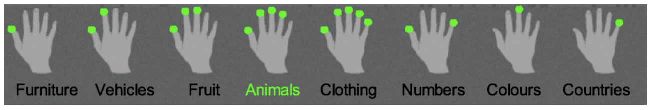

Figure 12. The Chord Map illustrates the 8 chords and their associated categories.

Menu Item Display and Access
Once a chord is articulated, the menu items associated with that chord/category are displayed. To change categories, the user can articulate a different chord, without having to first lift the hand. This allows users to quickly browse multiple categories. In a traditional marking menu, the user would need to constantly back track to do this.

The display of the menu items is the same as traditional marking menus; they are organized in a radial layout around a center point. However, in our design, we offset the entire menu 50 pixels above the top of the shadowbox. Similar to recent techniques for pen-based interaction [6], this alleviates the occlusion problem associated with direct touch interfaces. A cursor is displayed at the origin of the menu, which is offset and controlled by the first finger which contacts the display. The recognition of angular marks made by this offset cursor was done using the same algorithms used by the original marking menus [15].

As with the original marking menus, the menu items are only displayed after a short delay. Thus, an expert user can articulate a chord and perform the directional gesture, in a single fluid movement, without having to display the menu. As with traditional marking menus, menu cancellation is achieved by returning the cursor to the center dead zone.

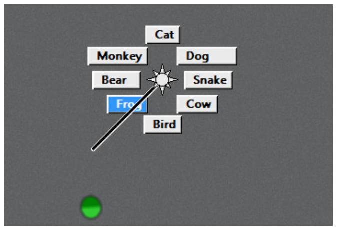

Figure 13. Menu items and an offset cursor are displayed above the contact point (shown in green).

EXPERIMENT 2: MAIN STUDY
Our second experiment was conducted to compare user performance of a traditional hierarchical marking menu (*MM*) to our multitouch marking menu (*MTMM*) design.

We compared performance in both novice and expert user scenarios. The novice user scenario was simulated by giving the user a word to select, requiring them to categorize the word and then use this category to search for the word within the menu system. The expert user scenario was simulated by showing the user the mark they were to draw (either a compound hierarchical mark or a directional chord) before the trial began.

Design
A repeated measures within-participant design was used. The independent variables were *Technique* (*MM*, *MTMM*) and *Mode* (*novice* or *expert*). Each participant performed the experiment in one session lasting approximately 45 minutes. The session was broken up into two sections, one for each technique. Each section was further broken up by the expertise mode, with all trials for one mode being completed before proceeding to the next mode. For each mode, there were four blocks of 16 trials each. These 16 trials exhausted the 8 items in two of the menu categories in

random order. Across the four blocks, all 64 items were selected exactly once. Before the first block for each technique-expertise level combination, participants were given one block of 16 random trials with which to familiarize themselves with that particular mode.

The design was counterbalanced by randomly assigning participants to one of four ordering groups. These groups were divided by which technique was completed first, and which order of expertise mode was applied to the trials for each technique.

Participants

We recruited twelve participants. Six of our participants were female and six were male, all between 20 and 26 years of age. All participants were right-handed, between 157cm and 188cm in height and reported no problems with mobility or motion. Participants were computer users, but none had extensive experience with multitouch.

Apparatus

Our experiment was conducted on the Microsoft Surface in the same configuration as the first experiment.

Procedure

Similar to our first experiment, participants stood in front of the multitouch display within comfortable reach of the working area. Participants were instructed to touch and hold a button in the centre of the screen, while their instruction was shown. Following the release of this button, the instructions would disappear the trial would begin. If the user forgot the instruction, they could press a help button in the corner of the screen, which would restart the trial. The mechanics of the trial depended on the technique and experience level variables.

We developed a menu consisting of 8 generic but identifiable categories (such as "Animals"), and 8 items for each category (such as "Zebra"). The chord map was only displayed after a trial began. For control purposes, we included an equivalent "direction map" for the marking menu technique, which showed which directions were required for each category. In novice mode the instruction presented the target menu item, while in expert mode, the instruction provided a graphical depiction of the required gesture. For the marking menu this consisted of a compound hierarchic mark, while for the multitouch marking menu, this was a directional chording gesture.

Results

Trial Completion Time

Trial completion time was measured as the time between lifting the finger from the start box, until lifting the finger(s) after performing the gesture. Trial completion time was significantly affected by *Technique* ($F_{1,11}$ = 6.4, $p < 0.05$) and *Mode* ($F_{1,11}$ = 326.534, $p < 0.001$). Multitouch marking menus decreased the mean completion time from 4051.3ms to 3932.3ms in novice mode, and from 1103.9ms to 851.7ms in expert mode. There was no significant interaction between *Technique* and *Mode*.

Articulation Time

Articulation time was defined as the time until the chord (or contact point for marking menus) was articulated. The *Technique* ($F_{1,11}$ = 126.6, $p < 0.001$) and *Mode* ($F_{1,11}$ = 226.5, $p < 0.001$) had a significant effect on articulation time. There was also a significant interaction effect between *Technique* and *Mode* ($F_{1,11}$ = 226.5, $p < 0.001$). Multitouch marking menus had a much higher articulation time in novice mode than marking menus (1730ms vs 416ms) because the user first had to find the desired category on the chord map (Figure 14). With the marking menu, users immediately placed their finger down to bring up the top level menu. In expert mode, the articulation times were virtually equivalent (277.5ms for *MM*, 275ms for *MTMM*). We believe part of this "articulation" time could be reduced for multitouch marking menus in the novice mode, with iteration on the design of the chord map, which users sometimes had trouble using to perform their visual search.

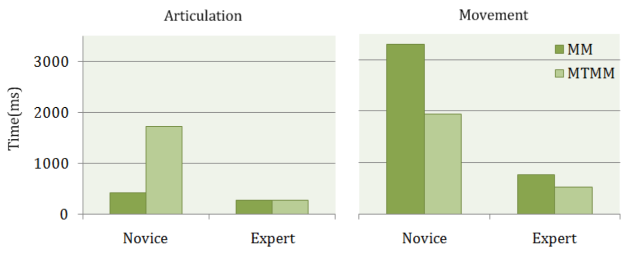

Figure 14. A comparison of Marking Menu and Multitouch Marking Menu trial time performance in Novice and Expert modes.

Movement Time

Movement time was significantly affected by *Technique* ($F_{1,11}$ = 135.709, $p < 0.001$) and *Mode* ($F_{1,11}$ = 189.917, $p < 0.001$). There was no interaction effect between *Technique* and *Mode.* In novice mode, average movement times were 1949ms for *MTMM* and 3298ms for *MM*. In expert mode, average movement times were 528ms for *MTMM* and 767ms for *MM.* This demonstrates the main benefit of the multitouch marking menu technique. The initial menu traversal is replaced by a quick chord posture.

Error Rate

Errors were defined as trials where the participant selected the wrong menu item. Trial error rate was not significantly affected by *Technique (MM or MTMM)* ($F_{1,11}$ = 135.709, $p < 0.001$), but the effect of *Mode (Novice or Expert)* was significant ($F_{1,11}$ = 32.575, $p < 0.001$). The average error rates were 6.4% in novice mode and 14.9% in expert mode. Our observations indicated that the high error rate in expert mode was due to problems with the Microsoft Surface tracking. When contact points moved quickly, as in the expert mode, they could be lost by the tracking system, causing the menu to end prematurely, with the wrong item being selected. The discrepancy between error rates in Novice and Expert modes, despite identical chord sets, supports our belief that errors are not primarily caused by difficulty articulating chords. This could potentially be addressed by allowing dropped contact points to be recaptured before dismissing the menu.

DISCUSSION

Given the variety of multitouch technologies, it is important to discuss how our research can be applied to other platforms. Our technique had two technological requirements. The first needed ability is to identify up to 5 distinct contact points, for each finger of that hand. This should not be problematic, as the most common multitouch technologies used today all have this ability. The second ability is to disambiguate fingers. Using our shadowbox recognition technique, we are able to reliably infer some finger identities. While other vision-based systems should be able to do this, capacitive systems may not. However, a capacitive system will still be able to identify the 5 chords defined by the number of contact points.

Aside from these two properties, we do not assume any other information streams, such as tracking state input [21] or pressure [8]. If such data streams become standard for multitouch technologies, it may be interesting to consider how they could be utilized within our design. For example, the chording map could be displayed when the user hovers over the display surface.

Our results may also need to be reconsidered for different display surface configurations, such as size and orientation. For example, different chords may be harder to articulate on a vertical multitouch surface.

Because our menu design provided a large menu breadth (64), we limited the menu depth to a single level. With such a large breadth, the menu size would explode if a second level were added. For example, if the second level had the same breadth, the total number of menu items would be $248^2 = 61504$. Obviously this is beyond the needs of any multitouch application, and would suffer a drawback that the user would need to change the chord articulation between levels. A more viable alternative would be to keep the chord constant throughout the compound stroke, giving the second level a breadth of 8, defined by the second direction. With this restriction, the menu could contain up to 64x8 = 512 items.

LIMITATIONS

We did not investigate mode switching techniques to transition between application usage and the command activation mode needed for menu usage. This would be necessary to prevent interference with main operations. A thorough investigation needs to be conducted, similar to those carried out for pen-based applications [19]. We foresee the use of mode-switching techniques such as reserved command-zones, bimanual input, or invocation gestures as potential candidates.

We also need to consider how the technique can be made self-revealing for first-time users. Our hope is that the chord map may be beneficial in such scenarios, but this will need to be studied further.

In addition, we did not explicitly investigate fatigue and other potential ergonomic issues arising from long-term usage of our technique. Our studies indicated that an hour of continuous use did not cause undue strain on the user, but this could be investigated further.

Another potential limitation is the relatively high error rates measured in our studies. While we believe these were due to tracking errors, it is still an issue which needs to be addressed. For example, our results could guide hardware developments of multitouch systems to ensure maximum velocities could be accurately tracked.

FUTURE WORK

For the sake of our evaluation, we used a fixed user location. However, in a real usage scenario, the user could be standing anywhere around the display. This would not be problematic for our technique, since the menus pop-up in place, and do not need to be accessed from a specific location. Furthermore, given recent development in finger orientation recognition [29], we would be able to appropriately rotate the shadowbox to maintain our accurate recognition of the chords.

Our implementations were also restricted to a single user scenario. However, a benefit of multitouch platforms is their appropriateness for collaborative usage [27]. In a multi user scenario, our technique could still be used given the location and angle independence described above. In addition, territorial research shows that in general, users will work in their own personal spaces [26], so contact point conflicts should not be problematic.

In our work, we focused on dominant-hand usage, for both our empirical investigation of directional chords, and usage of the multitouch marking menu. However, it may be desirable is some scenarios to support non-dominant hand usage, for example, if the dominant hand is using a pen [5].

Finally, while our shadowbox recognition technique supported accurate identification of chords, it did not provide identification of all 31 possible chords. Furthermore, it may not be robust to significant changes in the user's hand posture. For example, if the hand was postured with only index finger pointing, the middle finger would no longer be at the top of the bounding box. Thus, future work could look at more advanced vision based techniques, such as recognition of hand postures [21], to improve the chord recognition.

CONCLUSION

We presented an investigation into menu techniques for multitouch user interfaces. Informed by a first experiment on directional chording gestures, we developed multitouch marking menus. The design allows users to specify a top-level category *and* menu item with a single stroke. Our second experiment showed a statistically significant performance increase for multitouch marking menus over traditional marking menus, reducing execution times in both novice and expert usage modes. These results indicate that multitouch marking menus could be an efficient menu system for use within multitouch applications.

REFERENCES

1. Apple iPhone, http://www.apple.com/iphone/
2. Bailly, G., Lecolinet, E., and Nigay, L. (2008). Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization. *ACM AVI*. 15-22.
3. Bau, O. and Mackay, W. E. (2008). OctoPocus: a dynamic guide for learning gesture-based command sets. *ACM UIST*. 37-46.
4. Benko, H., Wilson, A.D., Baudisch, P. (2006). Precise selection techniques for multi-touch screens. *ACM CHI*. 1263-1272.
5. Brandl, P., Forlines, C. Wigdor, D., Haller, M., Shen, C. (2008). Combining and Measuring the Benefits of Bimanual Pen and Direct-Touch Interaction on Horizontal Interfaces. *ACM AVI*. 154-161.
6. Brandl, P., Seifried, T., Leitner, J., Haller, M., Doray, B., and To, P. (2009). Occlusion-Aware Menu Design for Digital Tabletops. *ACM CHI*. 3223-3228.
7. Card, S. K. (1982). User perceptual mechanisms in the search of computer command menus. *ACM CHI*. 190-196.
8. Davidson, P. L. and Han, J. Y. (2008). Extending 2D object arrangement with pressure-sensitive layering cues. *ACM UIST*. 87-90.
9. Dietz, P. and Leigh, D. (2001). DiamondTouch: a multi-user touch technology. *ACM UIST*. 219-226.
10. Esenther, A. and Ryall, K. (2006). Fluid DTMouse: better mouse support for touch-based interactions. *AVI*. 112-115.
11. Fingerworks, Inc. (2008). User's Guide. http://www.fingerworks.com/gesture_guide_mouse.html
12. Freeman, D., Benko, H., Ringel-Morris, M., Wigdor, D. (2009). ShadowGuides: Visualizations for In-Situ Learning of Multi-Touch and Whole-Hand Gestures. *ACM ITS*.
13. Han, J. Y. (2005). Low-cost multi-touch sensing through frustrated total internal reflection. *ACM UIST*. 115-118.
14. Jacob, R, A. Girouard, L.M. Hirshfield, M.S. Horn, O. Shaer, E.T. Solovey, and J. Zigelbaum. (2008). Reality-Based Interaction: A Framework for Post-WIMP Interfaces. *ACM CHI*. 201-210.
15. Kurtenbach, G., Buxton, W. (1993). The limits of expert performance using hierarchic marking menus. *ACM Interact/CHI*. 482-487.
16. Kurtenbach, G., Sellen, A., Buxton, W. (1993). An empirical evaluation of some articulatory and cognitive aspects of marking menus. *Journal of Human Computer Interaction*, 8(1):1-23.
17. Kurtenbach, G., Fitzmaurice, G. W., Owen, R. N., and Baudel, T. (1999). The Hotbox: efficient access to a large number of menu-items. *ACM CHI*. 231-237.
18. Lenman, S., Bretzner, L., and Thuresson, B. (2002). Using marking menus to develop command sets for computer vision based hand gesture interfaces. *ACM NORDCHI*. 239-242.
19. Li, Y., Hinckley, K., Guan, Z., and Landay, J. (2005). Experimental Analysis of Mode Switching Techniques in Pen-based User Interfaces. *ACM CHI*. 461-470.
20. Lyons, K., Starner, T., Gane, B. (2006). Experimental evaluations of the Twiddler one-handed chording mobile keyboard. *Human-Computer Interaction*. 21(4):343-392.
21. Malik, S., Ranjan, A., Balakrishnan. R. (2005). Interacting with large displays from a distance with vision-tracked multi-finger gestural input. *ACM UIST*. 43-52.
22. Matejka, J., Grossman, T., Lo, J., and Fitzmaurice, G. (2009). The design and evaluation of multi-finger mouse emulation techniques. *ACM CHI*. 1073-1082.
23. Potter, R.L., Weldon, L.J., Shneiderman, B. (1988). Improving the accuracy of touch screens: an experimental evaluation of three strategies. *ACM CHI*. 27-32.
24. Rekimoto, J. (2002). SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. *ACM CHI*. 113-120.
25. Rosenberg, I. D., Grau, A., Hendee, C., Awad, N., and Perlin, K. (2009). IMPAD: an inexpensive multi-touch pressure acquisition device. *ACM CHI EA*, 3217-3222.
26. Scott, S., Carpendale, S., and Inkpen, K. (2004). Territoriality in collaborative tabletop workspaces. *ACM CSCW*. 294-303.
27. Shen, C., Vernier, F.D., Forlines, C., Ringel, M. (2004). DiamondSpin: An Extensible Toolkit for Around-the-Table Interaction. *ACM CHI*. 167-174.
28. Vogel, D. and Baudisch, P. (2007). Shift: a technique for operating pen-based interfaces using touch. *ACM CHI*. 657-666.
29. Wang, F., Ren, X., Cao, X., Irani, P. (2009). Detecting and Leveraging Finger Orientation for Interaction with Direct-Touch Surfaces. *ACM UIST*. 23-32.
30. Wigdor, D., Balakrishnan, R. (2004). A comparison of consecutive and concurrent input text entry techniques for mobile phones. *ACM CHI*. 81-88.
31. Wigdor, D., Fletcher, J., and Morrison, G. (2009). Designing user interfaces for multi-touch and gesture devices. *ACM CHI*. 2755-2758.
32. Wilson, A. (2005). PlayAnywhere: A Compact Tabletop Computer Vision System. *ACM UIST*. 83-92.
33. Wobbrock, J.O., Morris, M.R. and Wilson, A.D. (2009) User-defined gestures for surface computing. *ACM CHI*. 1083-1092.
34. Wu, M., Balakrishnan, R. (2003). Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays. *ACM UIST*. 193-202.
35. Zhao, S., Balakrishnan, R. (2004). Simple vs. compound mark hierarchical marking menus. *ACM UIST*. 33-42.
36. Zhao, S., Agrawala, M., Hinckley, K. (2006). Zone and polygon menus: using relative position to increase the breadth of multi-stroke marking menus. *ACM CHI*. 1077-1086.