Enhancing Android Accessibility for Users with Hand Tremor by Reducing Fine Pointing and Steady Tapping

Yu Zhong[1,2], **Astrid Weber**[2], **Casey Burkhardt**[2], **Phil Weaver**[2], **Jeffrey P. Bigham**[1,3]

Computer Science, ROC HCI[1]	Google Research[2]	HCI and LT Institutes[3]
University of Rochester	Mountain View, CA 94043 USA	Carnegie Mellon University
Rochester, NY 14627 USA	{aweber, caseryburkhardt,	Pittsburgh, PA 15213 USA
zyu@cs.rochester.edu	pweaver}@google.com	jbigham@cmu.edu

ABSTRACT

Smartphones and tablets with touchscreen have demonstrated potential to support the needs of individuals with motor impairments such as hand tremor. However, those users still face major challenges with conventional touchscreen gestures. These challenges are mostly caused by the fine precision requirement to disambiguate between targets on small screens. To reduce the difficulty caused by hand tremor in combination with small touch targets on the screen, we developed an experimental system-wide assistive service called *Touch Guard*. It enables *enhanced area touch* and a series of complementary features. This service provides the *enhanced area touch* feature through two possible disambiguation modes: magnification and descriptive targets list. In a laboratory study with motor-impaired users, we compared both modes to conventional tapping and tested *Touch Guard* with real-world applications. Targets list based disambiguation was more successful, reducing the error rate by 65% compared to conventional tapping. In addition, several challenges and design implications were discovered when presenting new touchscreen interaction techniques to users with motor impairments. As the experimental product of an intern research project at Google, *Touch Guard* demonstrates broad potential for solving accessibility issues for people with hand tremor using their familiar mobile devices, instead of high-cost hardware.

Author Keywords

Accessibility; motor space; visual space; magnification; Android; disambiguation; fine pointing; steady tapping.

ACM Classification Keywords

H.5.2 [Information interfaces and presentation]: User Interfaces–*Input devices and strategies*.; K.4.2 [Computers and society]: Social issues–*Assistive technologies for persons with disabilities*.

INTRODUCTION

Users of mobile touchscreen devices with motor impairments such as low strength, unintentional tremor, poor coordination and easy fatigue can find conventional gestures challenging. The difficulties may be so serious that users refrain from using touch devices such as smart phones and tablets. Prior studies in both controlled and real-world settings have confirmed that even the simplest task, single tapping, can require significant effort and time for these users [6, 11]. These studies have shown that users with motor impairments tend to make more mistakes when selecting their touch targets than users without impairments. When performing tapping tasks on touchscreen, users with motor impairments are slower and have long dwell times [14]. Users with motor impairments also find multitouch gestures difficult and sometimes impossible [14, 33]. Despite those challenges, users with motor impairments still express enthusiasm about touchscreen devices' potential to empower them with more independence in daily life [2, 15, 33].

Single tapping as a gesture is the most used method to activate targets on touch screen devices. Similar to clicking with the mouse on desktop, smaller size and higher density of targets leads to more errors and longer acquisition time [10, 30]. For motor-impaired users, targets on small touchscreens of smartphones or tablets can be especially difficult to tap. Naftali *et al.* [21] found error corrections are expensive for motor-impaired users and often lead to frustration. Unfortunately, small and dense targets are very common on mobile touchscreen interfaces, for example, the *"more options"* menu button is only 7 mm in width on Android. Error rate on such small targets can be higher than 40% [10]. Even worse, small targets may be nested in a bigger button which can be easily activated by error. For example, contact buttons are nested in dial buttons in the Android speed dial interface. In order to improve the accessibility of touchscreen devices, it is imperative to make target acquisition easier and less error prone for users with motor impairments.

Little work has been done to address the challenges motor-impaired users face everyday when using their touchscreen mobile devices, or to study what users with motor impairments value when presented with new interaction techniques. Early solutions on styli-style devices such as PDAs utilized physical edges or raised bezel to guide users' input on the screen [9, 36]. Those physical features are no longer available on mainstream touchscreen devices. Although other options exist on iOS and Android to solve some of the accessi-

bility problems for motor-impaired users, they often fail their purposes due to high demands of dexterity control to activate [33]. We incorporated valuable insights and design implications derived from prior studies to address usability problems of Android for motor-impaired users on system level.

In this paper, we introduce *Touch Guard*, a service that runs in the background on Android and alters the way touch events are handled to reduce the precision of the user. *Touch Guard* can be used on any standard Android phone and works with any application installed. It supports target activation for motor-impaired users with *enhanced area touch* techniques. *Enhanced area touch* magnifies the motor and visual space with the goal of reducing the need for precise pointing and steady tapping, as well as lessening the effects of small and dense targets on disambiguation difficulty. As shown in Figure 1, *enhanced area touch* enlarges the single touch point into a circle with customizable radius centered at the finger on the screen. Any target whose bounds intersect with the circle can be activated. When multiple targets intersect with the circle, *Touch Guard* initiates an additional step to disambiguate the user's intention. We present and evaluate two disambiguation techniques: one with screen magnification at the touch point, the other with a text list of target descriptions. In addition to *enhanced area touch*, *Touch Guard* has several complementary features to support interaction with Android touch devices for users with motor impairments. These features were suggested by prior studies [2, 21, 33].

In a controlled lab study with eight participants with hand tremor, we compared the two disambiguation modes to conventional tapping and monitored the usage of *Touch Guard* with existing mainstream Android apps. Although both disambiguation modes were slower than tapping because of unaccounted correction cost and additional steps introduced. With the list based disambiguation, *enhanced area touch* significantly reduced errors made when selecting targets on a touchscreen by 65%. Moreover, during the studies users found *Touch Guard* empowering and capable of preventing high cost errors such as accidental phone calls in the dialer.

The main contribution of this paper is the introduction and evaluation of *Touch Guard*, an assistive technology service that demonstrates the effectiveness of using enlarged motor and visual space to reduce errors for users with motor impairments on touchscreen devices. The design of *Touch Guard* includes the first general-purpose service that improves accessibility for users with motor impairments and supports the interaction with any application running on a touchscreen system. We also contribute our discoveries when presenting a new touchscreen interaction technique to users with motor impairments and confirm accessibility challenges documented in previous studies (*e.g.* [21]). Our findings show that *Touch Guard* provides solid benefits to users with motor impairments, empowering them to utilize Android touchscreen devices in daily life.

RELATED WORK
56% of American adults now own a smartphone [31]. 61.9% of them have Android devices, and 32.5% own Apple products, according to the latest data from Kantar Worldpanel

ComTech [18]. Most modern smartphone models now have a touchscreen which is physically easier for people with motor impairments to use [33]. Despite the potential of widely penetrated touchscreen devices to empower users with motor impairments, there are still accessibility issues that make the interaction with touchscreen challenging for those users.

In both laboratory and real-world contextual studies, it has been confirmed that users with motor impairments make more mistakes when tapping small targets than non-disabled [21, 33]. The reason is that considerable movements within touches were found, while a tap is only recognized when a touch event stream starts and ends on the target. Multi-touch gestures such as sliding is also difficult for their participants [21], especially when limited to a small area. Guerreiro *et al.* [10] observed that some directions are more problematic than others when performing directional gestures. Other gestures that involve a time limit or multiple fingers are even harder, sometimes impossible for users with motor impairments to complete. [33]. In a large scale study of user submitted videos, Anthony *et al.* [2] found users with motor impairments often have to come up with their own interaction styles and adaptations to make their touchscreen devices physically accessible. For example, many users use homemade physical guides and barriers to aid pointing or support the device. In many cases, those users interact with the touchscreen not by using their fingers, but with an alternative such as their foot, nose or a prosthesis. Those user-generated work-around techniques confirm the need for further work on touchscreen accessibility for users with motor impairments.

Some design implications or proposed techniques are also derived from prior projects. Nicolau *et al.* [22] suggested an inclusive interface is possible for users with and without motor impairments by employing tapping as the main interaction technique. By studying the performance of 15 participants with quadriplegia on touchscreens, Guerreiro *et al.* [10] concluded the minimum target size should be 12 mm. Multiple studies [2, 21, 33] have proposed to provide alternatives for multitouch gestures, and a filter of unintentional touches. Trewin *et al.* also suggested the importance to reduce visual efforts, e.g. larger fonts, when designing a new interaction technique as many users with motor impairments also have visual impairments [33]. Naftali and Findlater [21] highlighted the importance of configurablility and contextual adaptation. More proposed features include an undo functionality which prevents expensive corrections of errors [33] and using sliding to replace tapping [34]. However, Guerreiro *et al.* [10] compared tapping with crossing, exiting, etc. and found they did not have significant effects on error rate.

When examining the current state of touchscreen devices accessibility, the situation is far from optimal. Apple has deployed Zoom, Assistive Touch and Speak Selection features on iOS, but all of them are hard to activate and to operate because of a mismatch between the required abilities to operate those features, and the actual abilities of the participants [33]. Android has a screen magnification feature which is slightly easier than zooming on iOS but it is mainly designed for users with a visual impairment. On both iOS and Android, system
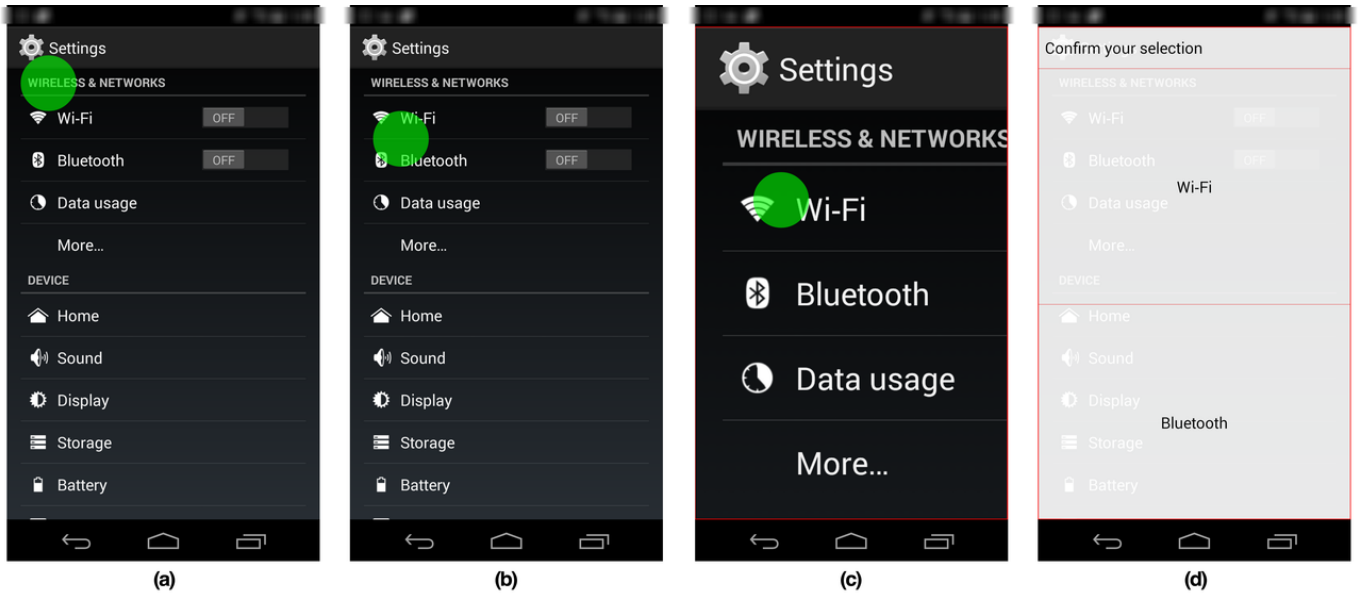
Figure 1. *Enhanced area touch* **working on existing Android application interface: (a) Touch area over single target. (b) Touch area intersecting with multiple targets. (c) Disambiguation in a magnified area of interest. (d) Disambiguation in a full screen list of captured target descriptions.**

default and in-browser magnification supports are designed for and tailored towards needs of users with visual impairments. For users with severe motor impairments, scanning-based interaction [7] is available on iOS but it is very inefficient for users with gross or fine motor control. Early research efforts proposed effective interaction techniques on non direct touch devices. HandyGlyph allows users to enter text more easily with a stylus [4]. Barrier Pointing utilizes the edge of the screen to improve target acquisition accuracy for users with motor impairments [9]. Similarly EdgeWrite uses hard edges to aid gestures on a stylus-based text entry interface [36]. But the described solutions are not applicable on mainstream touchscreen devices as those physical features have been deprecated. There has been some efforts to model users' motor control capacities in order to provide adaptive touchscreen interfaces [13, 16, 19], but they are limited to specific applications. Zooming, also known as magnification, has been used to make extremely small screens such as those of smart watches more useable. Especially text entry efficiency was improved that way [24], but the devices themselves are not designed to support users with tremor.

There have been many novel solutions to the challenges people with hand tremor face on more traditional computer setups. Although a mouse cursor is different from direct touch, some underlying ideas are the same and can be borrowed. For example, Area Cursors reduces the need for fine pointing when using a mouse by magnifying the visual and motor space. This has been proved to be beneficial for elderly [37] and motor impaired [8] people. Steady Clicks reduces errors by freezing the cursor at button down location, but does not improve target acquisition time [32].

For able-bodied users, various touchscreen interaction techniques have been developed to facilitate precise pointing of small targets. From early solutions proposed on less sensi-

tive resistive touchscreen such as Zoom-Pointing [1], Take-Off [29], and Rubbing [23], Pointing Lenses [27] for stylus input, to more recent work on use of finger orientation [35], fingerprints [12] and back-of-device [3]. Although many of techniques are not suitable for users with motor impairments because of high demands of physical abilities or dependence on additional hardware features, Take-Off inspired us to use similar strategy to reduce needs for steady tapping.

TOUCH GUARD
Touch Guard is an Android accessibility service designed for use with any applications and other accessibility services (e.g. *JustSpeak* [38]) on Android 4.0 and above. Once installed, users can enable and grant necessary permissions for *Touch Guard* within Android Accessibility Settings (one-time operation). Activation and settings of *Touch Guard* will then always be easy to access through the notification page.

In this section, we first discuss designs of the *Touch Guard*, then describe the implementation of *enhanced area touch* and its two disambiguation modes. Finally we introduce other features that complement the functionality and usability.

System Description
Instead of altering arrangements of the application interface, *Touch Guard* renders a transparent full-screen overlay on top of the foreground application. The overlay intercepts all touch events which are then processed in order to execute commands on the interface accordingly. From the user's perspective, he is still seeing and using the same Android they would without *Touch Guard*. Once deactivated, *Touch Guard* hides the overlay and Android responds to touch in its default ways. Since *Touch Guard* is powered by touch events, it is targeted at users who can physically access a touchscreen. As discussed before, given the high variation of motor impairments, configurability is crucial. *Touch Guard* allows the user

3

to toggle or adjust each feature so that it can best fit individual needs. The default settings are informed by our studies and the system design also reflects refinements made after careful reviews of design implications proposed by prior studies.

Touch Guard is *target aware*, meaning the service knows the location and size of all interactive on-screen targets so that it can be deployed in the wild and supports users with hand tremor within any applications installed on their Android touchscreen devices. Target awareness was made possible on Android when Google introduced its Accessibility APIs for Android [26]. Given permissions from the user, accessibility services can access view hierarchy and screen update events from Android system. *Touch Guard* retains a cache of on-screen targets and filter out those non-interactive elements. The cache is refreshed whenever there is an update on the screen. Commands such as target activation are also delegated through Accessibility APIs.

Enhanced Area Touch

By creating the *enhanced area touch* feature in *Touch Guard*, we aimed to reduce the needs for precise pointing, especially when selecting small and dense targets which are very commonly seen in Android applications. We followed the same logic of area cursors [8] to enlarge both motor and visual space so that users with hand tremor need to spend less visual and physical efforts to activate targets.

Enhanced area touch design converts a single tapping task into two steps: selection and disambiguation. Although the underlying structure is the same with area cursors, there are some necessary differences in the interaction flow to accommodate the unique characteristics of the touchscreen. First, a real-time lense-style magnification is not feasible. Because when using direct touch, the user's active finger on the screen blocks targets underneath. Second, we did not inherit the targets proximity based dynamic re-sizing technique because the distance between targets on a touchscreen is mostly minimal to save screen space. In fact, targets are often nested on touchscreen devices, meaning it is a common practice to put smaller targets inside bigger parent views which are also interactive. Since we do not have a real-time visual magnifier to enlarge the distance between targets, resizing of touch area degenerates and introduces unnecessary learning overhead.

As shown in Figure 1 *(a)*, when the user places a finger on the touchscreen, a green circle is drawn underneath the finger tip to illustrate the enlarged touch area. The size of the touch area can be adjusted in *Touch Guard* the settings (defaulted to 96 pixels on a 5 inch screen, ranging from 32 to 244 pixels). When the touch area intersects with a single target, even if the direct touch point is outside, the target will be activated when the finger leaves the screen. More often, the touch area intersects multiple targets. In this case, disambiguation of the user's intention at finger lifting time is necessary. *Touch Guard* offers two modes of disambiguation.

Magnification

The first is magnification as shown in Figure 1 *(c)*. When entering magnification disambiguation phase, *Touch Guard* magnifies the active area of current screen, excluding the status bar and system soft buttons. The magnification factor can be adjusted from 1 to 3 (defaulted to 1.5). When the magnification factor is set to 1, *Touch Guard* activates the intersected target with shortest distance to the center of touch area instead of magnifying the view. The magnified view is centered near the touch point to show the area of interest. An offset to the touch point is introduced to make sure targets close to edges of the screen will not be cut off by edges or hidden when magnified. The user then performs the same interaction in the magnified screen to confirm selection. In the case that the touch area still captures multiple targets in the magnified screen, *Touch Guard* activates the target which is closest to the touch point. We did not implement recursive magnification because it can potentially confuse users and slows down interactions as suggested in the pilot studies. To aid visual mapping, the appearance of the magnified screen is animated from touch point. A red border is also drawn to inform the user that *Touch Guard* is currently in magnification disambiguation phase. Some mobile web browsers have a similar in-app feature to support desktop sites but those browsing magnifiers are not designed for users with motor impairments and do not magnify motor space. The pinching gesture is not supported in magnified view as it has been proven difficult for those users, [2, 33] neither is dragging allowed because it is not compatible with the *click-on-lift* feature described below.

Targets list mode

The second disambiguation mode is called targets list mode which shows a list of descriptions of potential targets as shown in Figure 1 *(d)*. When entering this targets list disambiguation mode, *Touch Guard* extracts the title of targets, or descriptive text supplied for icons to form a full screen list and asks the user for confirmation. To aid visual mapping of targets, the background of the list is made partially transparent so the user can see through the list and inspect on-screen targets underneath. To ensure a large motor space in the list, *Touch Guard* enforces a minimum row size of the list which is 1/4 of the screen height if the device is currently held vertically or 1/3 of the screen width if it is held horizontally. The list is scrollable so even each page has at most 4 targets, it can supports indefinite number of targets. The placement of targets is based on an ascending order of distances between targets and the center of touch area. The target closest to the center is placed at the top of our list.

In order to allow the user to recover from errors, *Touch Guard* intercepts the Android hardware back button events in the disambiguation phase of either mode. When the user hits the back button in the disambiguation phase, they can exit incorrectly activated disambiguation and return to the previous screen. Also if the user accidentally touches somewhere, the targets list will be shown instead of initiating unwanted commands within the foreground application so that the user can easily recover the previous application states.

Complementary Features

In addition to *enhanced area touch*, we have provided several features in *Touch Guard* in order to address challenges users with tremor face when interacting with Android.

Click-on-lift option is offered to reduce needs for steady tapping. Steady tapping is only recognized if the user puts down and lifts a finger within the same target and did not move out in between. When this feature is enabled, the user can put one finger down anywhere on the screen and only needs to lift the finger within the target to activate it. It also helps the user recover from slipping while tapping as the user can move the finger back and still be able to acquire the target. This feature is carefully designed to work with *enhanced area touch*. When *click-on-lift* is enabled, only the touch area around the finger lifting point is recorded. On the other hand, when it is disabled, both touch areas around the initial contact point and finger lifting point are recorded. Only targets that intersect with both areas are considered candidates for activation.

Touch Guard also enables an *unintentional movements filter* by default to reduce errors caused by hand tremor. This filter targets at high speed movements or movements with sharp turning angle as those patterns indicate the user's finger has slipped. Therefore those movements should be ruled out when *Touch Guard* determine action to perform. *TouchGuard* records every touch event and detects real time speed of the active finger on the screen based on the distance between its current and previous location. It also calculates the angle between consecutive finger movements. When either speed or angle exceeds a certain threshold, *Touch Guard* deems the current touch event to be caused by hand tremor and ignores it. When *click-on-lift* is enabled, this filter prevents incorrect activation of targets when the user's finger slips and leaves the screen within the bounds of unwanted targets. It can also assist steady tapping if *click-on-lift* is disabled by ignoring movements outside the target, without this filter, the user has to lift the finger and start over after the finger slips out of the target.

Scrolling is another important interaction on touchscreen and often performed with swiping gesture. However, it has been evaluated by prior studies and proven challenging for many users with motor impairments [2, 21, 33]. Specifically, many of them have problems performing swiping in certain direction, or constraining the swiping path within the bounds of targets. *Touch Guard* supports swiping either horizontally or vertically anywhere on the screen to scroll a target forward or backward by one page, even if the orientation of swiping is different from the scroll bar's, or outside the scrollable element. This solution is feasible because in most touchscreen interfaces scrollable elements are bi-directional instead of quad-directional. Also because on most touchscreen interfaces only one element is scrollable, full screen support doesn't introduce risk of errors. Finally, the distance and velocity threshold of a swiping gesture are more relaxed than Android system default.

USABILITY STUDY

The main goal of this study was to evaluate the extent to which our *enhance area touch* design improves the performance and reduces errors among users with tremor compared to conventional tapping, particularly in regard to small and dense targets. Another objective was to observe the usage of

Touch Guard with real-world applications in order to estimate its value in a more realistic context.

Participants

For the purpose of this study we recruited eight participants with motor impairments that affected their hands causing them to struggle with hitting touch targets precisely. Four of our recruited participants had Parkinson's disease, two had Multiple Sclerosis and two had essential tremor that was caused by accidents. All users were 51+ years of age. Half of the users were existing Android users. The remaining ones used either iPhones or feature phones. Three out of eight users were female, five male. None of the participants were using any assistive technology to operate their phones. All participants used their fingers to interact with the touchscreen. Participants were each reimbursed with a shopping voucher of $125.

Apparatus

The experiment application to collect quantitative data and the prototype of *Touch Guard* were developed with standard Android SDK 19. The study sessions were run with a Nexus 5 phone (4.95 inches screen, 1920×1080 resolution) running Android 4.4. The experiment application recorded touch events with millisecond timestamps. Other accessibility services and options were turned off. Participants were given the option to adjust settings of *Touch Guard* at the beginning of the target acquisition experiment. 5 participants proceeded with default touch area size while the other 3 used larger touch area. All participants chose to complete the experiment with default magnification factor and all complementary features enabled.

Procedure

Sessions lasted 90 minutes for each participant. The protocol for each session was structured into 4 main parts.

- **Introduction:** we gave the participant a brief introduction interview in order to better understand the challenges each individual user was facing and to collect demographic data.

- **Tasks:** the participant was asked to complete a series of tasks in essential Android applications as shown in Table 1 using *Touch Guard*. We did not give instructions before the tasks in order to observe how *Touch Guard* was learnt. Guidance was only given when the participant asked for it. Disambiguation mode was switched once (random order) and each was covered by half of the tasks.

- **Experiment:** each feature of *Touch Guard* was explained to the participant and settings are adjusted by the participant based on their own preferences. Then, an experiment in controlled settings was conducted to compare three touch types with target activation trials: two disambiguation modes of *enhanced area touch* and conventional tapping. We began each condition with a ∼2 minute preparation phase to allow the participants to familiarize themselves with the experiment application. The preparation phase ended once the participant had completed at least 4 unassisted trials and confirmed in a dialog. A test of block of 25 trials was then presented and participants were asked

Order	App	Task
1	*Touch Guard*	Activation
2	Android Settings	Open exploration
3	Phone	Calling 1st speed dial
4	Hangouts	Send a SMS
5	Hangouts	Read a SMS
6	Youtube	Search for a band
7	Maps	Search for direction
	Switching disambiguation mode	
8	Android System	Read a notification
9	Android System	Turn Wi-Fi On/off
10	Camera	Take a photo
11	Google Plus	Share a photo
12	Play Store	Install a game
13	Game installed	Open exploration
14	Launcher	Count apps installed

Table 1. Tasks participants were asked to complete, using *Touch Guard*.

to complete the trials as quickly and accurately as possible. For each trial, the participant had to correctly activate a green target among a set of gray or white targets. Trials took 15-25 minutes for each participant, depending on individual motor abilities.

- **Discussion:** a questionnaire was issued once all trials were completed to collect Likert scale data. Finally, we asked open ended questions such as *"If you can make any changes to Touch Guard, what would you do?"* to gather feedback.

Experiment Design
The controlled experiment was a 3×3 within-subjects factorial design. Factors and levels were:

- **Touch Type:** (1) Magnification based area touch, (2) list based area touch and (3) conventional tapping.

- **Interface Type:** We evaluated *enhanced area touch* on three types of Android touchscreen interfaces. As shown in Figure 2 (a), the first is a standard icon grid interface which is commonly used, for instance, on home screen. The second is an interface (Figure 2 (b)) in which small targets are nested into larger parent targets. This type of interface is also widely used in Android applications such as the speed dial page of default phone app. Figure 2 (c) shows the interface containing clustered targets with size smaller than suggested minimum target size (7mm) of Android applications. We were most interested in the performance of *enhanced area touch* on the second and third interface.

Measures
Acquisition time was measured as mean trial duration, calculated as the elapsed time from the end of the previous trial to activation of the correct target. Error rate was defined as the percentage of trials in which at least one activation of incorrect target occurred. Finally, Likert scale rating was collected with a modified System Usability Scale (SUS) model [5].
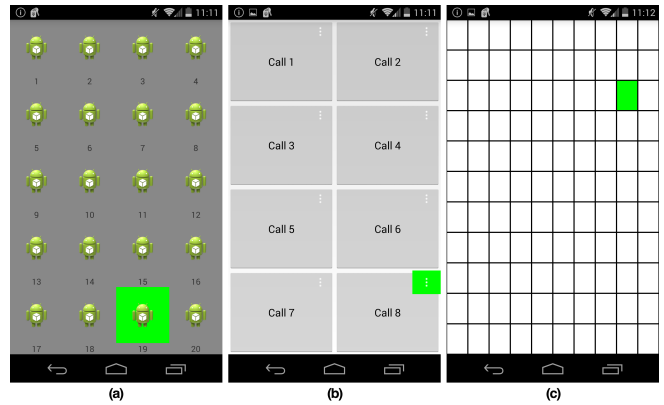

Figure 2. Interfaces on which *enhanced area touch* were evaluated.

Participants were asked to complete a questionnaire which evaluates *Touch Guard* as a whole application, and two disambiguation modes respectively at the end of sessions. Responses were categorized and coded to measure perceived usefulness, perceived ease of use, and user acceptance.

Analysis
Target acquisition time data were logarithmically transformed to correct for violations of normality as common practice. We analyzed the time data using mixed-effects ANOVA: touch and interface type were modeled as fixed effects while participant was modeled as a random effect because the levels of this factor were drawn randomly from a larger population. For such analyses, wider confidence intervals are used so it is not easier to detect significance despite larger denominator degrees of freedom retained [17, 28] . We used non-parametric Friedman tests and Wilcoxon pairwise comparisons on error rate data and Likert scale data. All post-hoc pairwise comparisons used a Bonferroni adjustment.

RESULTS
Only main and interaction effects with touch type are reported to save space. Pairwise comparisons were performed only when there were significant main effects.

Acquisition Time
A significant main effect of touch type on target acquisition time was found ($F_{2,672} = 81.71$, $p < .01$). Average target acquisition times were 1.7 seconds for tapping, 4.7 seconds for magnification based area touch, and 3.7 seconds for list based area touch (see Figure 3). No interaction effects were significant. We were not surprised to see *enhanced area touch* increased trial time. There are two reasons for the prolonged interaction time. The first is that because the trial app involve no cost for correction, participants developed a strategy of repeated tapping in the area of the target until hitting it. That strategy is not applicable to enhanced area touch since interface is updated in disambiguation phase. Second, when entering disambiguation mode, the target location is often shifted by a significant distance. Pairwise comparison showed that magnification is slower than tapping ($p < .01$) and the list mode ($p < .01$). This finding contradicts our initial intuition

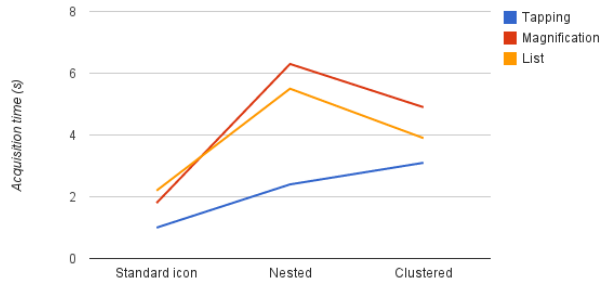since we thought magnification is more intuitive and offers easier visual mapping of targets.

Figure 3. Mean acquisition time across touch types and trial interfaces.

Error Rate

Average error rates were 7.0% for list based area touch, 25.8% for magnification based area touch and 20.2% for tapping as shown in Figure 4. Non-parametric tests were used on this data, with Bonferroni adjustments for significance testing. We report only significant results, and unadjusted p-value for consistency.
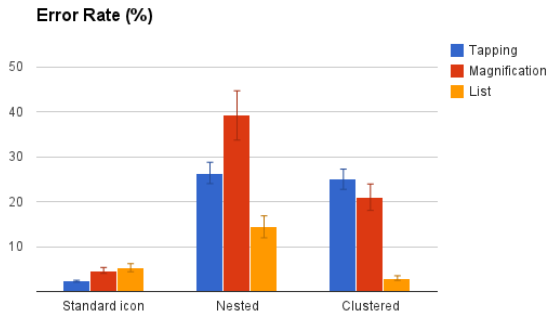

Figure 4. Mean error rates across touch types and interfaces show lower rates for list based area touch, especially on small and dense targets.

Only list based area touch mode reduced errors comparing to tapping. Overall, touch type had a significant effect on error rate ($\chi^2(2) = 56.09$, $p < .001$). Pairwise comparison showed participants made fewer errors when using list based disambiguation mode than tapping ($Z = -6.66$, $p < .001$), but did not have significantly different error rates when using magnification. When examining how the touch types compared on different interfaces, we found the differences in error rates were largely caused by small and dense targets. There was a significant effect of touch type on error rates when participant performed tasks in both nested ($\chi^2(2) = 20.03$, $p < .001$) and clustered ($\chi^2(2) = 21.0$, $p < .001$) interfaces, but not the standard icon interface in which targets have large size.

Subjective Measures

We asked participants to choose their preferred disambiguation mode, all users preferred the list based disambiguation mode over magnification. Perceived usefulness, perceived ease of use, and user acceptance of *Touch Guard* and each disambiguation mode are shown in Figure 5. We found different disambiguation modes significantly affected user acceptance ($Z = -2.384$, $p < .05$). Although differences in other two dimensions were not significant, the lower Likert scores of magnification conforms with our finding that it is less effective than the targets list mode. Overall, participants found *Touch Guard* useful (mean = 3.38) and acceptable (mean = 3.5), but not easy to learn as a new interaction technique (mean = 2.63). This might be related to the fact that our participants were all over 51, as it is more difficult for older individuals to adapt to changes of technology. [20].
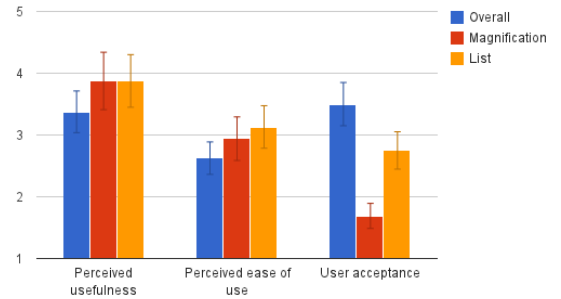

Figure 5. Perceived usefulness, perceived ease of use and user acceptance show that magnification is less preferred.

DISCUSSION

We were most interested in whether the *enhanced area touch* would help users with motor impairments select small, dense or nested targets, a situation in which conventional tapping is often too challenging for these users and consistently observed throughout our experiments. We are glad to see that, in comparison to tapping, with the targets list mode error rates were substantially reduced to under 10%. However, the failure of magnification is also of great value, especially its low acceptance score. It motivated us to dive deep into participants' responses and experiences and extract informative lessons.

Magnification vs. Targets List

As magnification seems more intuitive, has been applied in designs for able-bodied and visual impaired users on touchscreen, and proposed for users with motor impairments [21, 33], we found it surprising that the descriptive targets list mode outperformed magnification mode with both performance and users' preference. In retrospect, several reasons seem to trigger this result. Although magnification retains benefit of closer placing of targets and easier visual mapping, a factor smaller than 4 does not provide enough extra space when targets are small and dense to compensate for users' motor skill. However, when increasing the factor, the visible area of screen contents quickly decreases and this is especially problematic on small screens. During the study, the magnification mode often tended to cut off crucial parts of the regions of interest that were important for the users to understand which element they were about to activate, *i.e.* messages of a dialog. Another interesting discovery is that two

participants were convinced that magnification forced them to perform much more taps than the targets list mode while the numbers are comparable and both modes follow the same logic of labor division. It suggests the significant change of visual contents on the screen introduced by magnification requires more cognitive load to process. Several participants were also confused by magnification that they found themselves in a magnified view which looked familiar to them, but did not behave in a familiar way (*e.g.* inability to navigate within magnified screen). This problem might be solvable by placing the magnified view in a portion of the screen and highlighting the difference between background and magnified area. Two participants expressed that they do not need magnification as they *"do not have any vision problems" (P3, P8)*, indicating there is a conceptual link between visual impairments and any kind of screen magnification. Overall, magnification, although is an approach proven successful on desktop, might not be the solution on touchscreen, or at least has to be carefully designed to cope with characteristics of touchscreen devices.

Users appreciated the targets list mode for the following reasons. They liked the fact that it offers additional security before doing a crucial action like calling someone. This was mentioned several times, suggesting the importance of error reduction for motor impaired users as they have higher error rates and corrective operations are more expensive. They also felt that it was less noticeable (which is actually a crucial factor in designing for users with impairments as they often wish to be considered as *"normal" (P6)* as possible.) The better performance of error reduction was also noticed as the targets list mode was descried as *"clear and simple" (P6)* and made P5 *"more sure."* All complementary features were well received as *"tailored towards our needs and useful" (P1)*.

Design Implications

Participants, even those without any prior Android experience, were able to complete all tasks within real-world applications with the exception of playing a game. Since accessibility of games is not within our scope and the majority of games developed with graphic frameworks such as OpenGL do not properly implement the Android Accessibility API, we did not elaborate on this use case. The positive result displayed viability of supporting touchscreen users with motor impairments with low-cost software solutions. The following findings and design implications are based on our observations of the users interacting with the apps listed in Table 1 on the Android device.

- It is essential for developers to follow best practices for accessibility. We encountered several apps that used custom widgets and failed to provide meaningful content descriptions, which makes items in the targets list incomprehensible. In several cases, non-interactive widgets were falsely flagged and recognized as targets by *Touch Guard*, leading to confusion of participants. Those situations are also problematic for users with other impairments. For example, screen readers also rely on those descriptions to support blind users. In addition, supporting different orientations is important for users with motor impairments, as we frequently observed participants rotate their devices to ease fatigue.

- A steep learning curve is a major barrier for users with motor impairments, especially for those whose mental capability is also affected by disease or aging. We frequently observed that users unfamiliar with Android struggled with simple tasks or understanding certain features of *Touch Guard*. The higher perceived ease of use scores of targets list mode over magnification could also be partly attributed to the list being a standard Android UI component also used in other applications.

- When using text instead of graphic elements, original presentation of texts must be considered. It can be optimized by enriching the list layout with application context. For example, two participants thought that the targets list mode showed a summary of an email thread rather than separate grouped conversations inside Gmail. Two others initially believed this same mode was intended as an *"auto correction mode" (P5, P7)* that was about to predict what was going to be entered rather than confirming the number they had selected.

When gathering feedback, we spent more time discussing the value of *Touch Guard* with three of the participants who gave lower Likert scores, as their opinions would be more inspiring. From participants' point of view, the return on investment (ROI) was crucial. Those users who had enough physical control of their hands to access the back button were less likely to accept the additional *"tapping work"* introduced by *Touch Guard*. On the contrary, those users who found operating a touchscreen phone (including its back button) difficult or impossible showed more appreciation for *Touch Guard*. This discovery is related to the fact that *enhanced area touch* was slower than conventional tapping. They suggest there is a potential benefit in resizing touch area based on both target size and the user's individual pointing ability.

We found that various impairments that affect physical control of hands result in very unique needs that are difficult to be satisfy with a single service. Even within a group of users who shared a common condition, there was substantial variance in participants' ability to utilize *Touch Guard*. This leads to the conclusion that meaningful customization options are important for Touch Guard and similar services. Although we made configurabilty one of our priorities when designing *Touch Guard*, there is still room for further work, especially for scrolling support. Participants of our study struggled especially with those tasks that asked them to do things in a timely manner or with a specific level of precision. For example, even with our full-screen scrolling support with relaxed time and speed threshold, P8 still felt *"smaller movements"* were needed to scroll. Several users unintentionally touched their phone with two fingers at the same time. Although *Touch Guard* prevents accidental activation of incorrect targets by the second finger, it was not able to recognize swiping gestures in those situations. Moreover, as P6 did not have enough stability to keep his fingers away from the display during longer interactions, he scrolled back and forth, as

he could not lift his finger. Offering more customization options such as alternative gestures could be helpful. A more sophisticated approach would be automatic personalization with modeling of users' ability to perform different gestures. Although accurate profiling of motor-impaired users' abilities have been difficult to develop and deploy in the wild, especially with one-time assessment [16], a less intrusive learning model based on users' behaviors (*e.g.* measuring errors with times of immediate back button operations) could be feasible.

Finally, the fact that none of our participants used any assistive technology indicated a low awareness of accessibility products for users with motor impairments and an urgent need for more work in the space.

FUTURE WORK
Firstly, we plan to explore more sophisticated methods to apply magnification techniques in the same scope, such as context-aware lenses [25]. As discussed earlier, the key is to design a magnification mechanism which both conserves the important screen area and also sufficiently scales up visual space. It is also important to ensure any magnified view is presented with visual continuity while zooming in so the high cognitive load observed in the experiments can be avoided.

Experience with subjects in the targets list based disambiguation mode suggested several future works. First, graphic elements such as cropped target layout could be added to further ease visual mapping. Second, most likely target with shortest distance to touch point can be highlighted with a different color in order to speed up the process of confirmation. Finally, a smarter target placement algorithm that shortens distance between the target and corresponding item in the list can reduce the need for repositioning of fingers or sometimes even the whole hand to reach for targets on the other end of the display.

Participants also suggested several potential enhancements based on their own experience. One of them was to customize the scope of *Touch Guard*, only allowing it to function in certain situations where operations or corrections are relatively more difficult for him, such as when sending a text message or making a phone call (P1). Another participant (P6) proposed similar automatic toggling of *Touch Guard* based on the prediction of task importance, rather than forcing the user to *"do more tapping work"*. Moreover, to ease the effects of rapid fatigue, a "dead spot" on the screen could be useful, meaning an area where users could rest their fingers or even use it for stabilization purpose when using another finger.

In addition, *Touch Guard* will cater to an older demographic segment that largely consists of digital immigrants who have diseases that can also affect their mental capability. As such, we aim to further simplify the interface to increase ease of learning and use. One limitation of our study relates to participant selection. We had a relatively small group (N = 8), all participants were 50 years of age or older, and all had very limited experience with Android. Also, each used only their fingers to interact with the touchscreen. Although these participants could represent a difficult user group to design for, our findings may not generalize across gender, age, cul-

ture, interaction methods, and experience. In future work, we plan to explore more advanced interaction techniques, such as goal crossing, as well as quantitative evaluations of real-world applications that factor in more accurate costs for correction. After necessary iterations and enabling support for more gestures, e.g. disabmbiguation between multiple scrollable views, we plan to deploy and distribute *Touch Guard* in order to get feedback from real-world users with motor impairments and to improve their Android experience.

CONCLUSION
We have introduced *Touch Guard*, a background service designed to improve system-wide accessibility and usability of touchscreen devices running the Android operating system for users with motor impairments. The main feature, *enhanced area touch*, is designed to reduce needs for fine pointing and improve performance of motor impaired users. We focused on designing features that would allow users to select small and dense targets with less visual and physical efforts required by steady tapping, as it is an intensely difficult situation for users with motor impairments and yet commonly seen on small touchscreen devices. The design of *enhanced area touch* included two disambiguation modes: magnification and targets list. The targets list mode is more successful and significantly reduced errors when selecting small and dense targets, while magnification surprisingly failed to improve speed and accuracy. This experimental project shows the potential of using software based solutions to make readily available devices more usable for people with motor impairments, reducing the impact of cost barriers on touchscreen device adoption. Finally, we describe some of the less-documented challenges and design implications related to introducing a new touchscreen interaction technique for users with motor impairments.

REFERENCES
1. Albinsson, P. and Zhai, S. High precision touch screen interaction. CHI 2003.
2. Anthony, L. and Kim, Y. and Findlater, L. Analyzing user-generated youtube videos to understand touchscreen use by people with motor impairments. CHI 2013.
3. Baudisch, P. and Chu, G. Back-of-device interaction allows creating very small touch devices. CHI 2009.
4. Belatar, M. and Poirier, F. Text entry for mobile devices and users with severe motor impairments: handiglyph, a primitive shapes based onscreen keyboard. ASSETS '08.
5. Brooke, J. SUS-A quick and dirty usability scale. In *Usability evaluation in industry 189* page 194, 1996.
6. Duff, S. N. and Irwin, C. B. and Skye, J. L and Sesto, M. E. and Wiegmann, D. A. The effect of disability and approach on touch screen performance during a number entry task. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 2010.

7. Felzer, T., Nordmann, R. and Rinderknecht, S. Scanning-based human-computer interaction using intentional muscle contractions. In *Universal Access in Human-Computer Interaction. Intelligent and Ubiquitous Interaction Environments*, p. 509-518, 2009.

8. Findlater, L., Jansen, A., Shinohara, K., Dixon, M., Kamb, P., Rakita, J. and Wobbrock, J. O. Enhanced area cursors: reducing fine pointing demands for people with motor impairments. UIST 2010.

9. Froehlich, J. and Wobbrock, J. O and Kane, S. Barrier pointing: using physical edges to assist target acquisition on mobile device touch screens. ASSETS 2007.

10. Guerreiro, T. and Nicolau, H. and Jorge, J. and Gonçalves, D. Towards accessible touch interfaces. ASSETS 2010.

11. Guerreiro, T. João V. and Nicolau, H. and Jorge, J. and Gonçalves, D. Assessing mobile touch interfaces for tetraplegics. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, 2010.

12. Holz, C. and Baudisch, P. The generalized perceived input point model and how to double touch accuracy by extracting fingerprints. CHI 2010.

13. Hurst, A., Hudson, S. E. and Mankoff, J. Automatically identifying targets users interact with during real world tasks. IUI 2010.

14. Irwin, C. B and Sesto, M. E. Performance and touch characteristics of disabled and non-disabled participants during a reciprocal tapping task using touch screen technology. In *Applied ergonomics*, 43.6, pages 1038-1043, 2012.

15. Kane, S. K. and Jayant, C. and Wobbrock, J. O and Ladner, R. E. Freedom to roam: a study of mobile device adoption and accessibility for people with visual and motor disabilities. ASSETS 2009.

16. Kurschl, W. and Augstein, M. and Stitz, H. and Heumader, P. and Pointner, C. A User Modelling Wizard for People with Motor Impairments. In *Proceedings of International Conference on Advances in Mobile Computing & Multimedia*, 2013.

17. Littell, R. C. SAS for mixed models. *SAS institute*, 2006.

18. Galaxy S5 attracts some Apple customers in Europe. `http://goo.gl/56YKvw`. Accessed: 06/30/2014.

19. Montague, K. and Hanson, V. L. and Cobley, A. Designing for individuals: usable touch-screen interaction through shared user models. ASSETS 2012.

20. Morris, M. G and Venkatesh, V. Age differences in technology adoption decisions: Implications for a changing work force. *Personnel psychology*, 53.2, pages 375-403, 2000.

21. Naftali, M. and Findlater, L. Accessibility in context: understanding the truly mobile experience of smartphone users with motor impairments. ASSETS 2014.

22. Nicolau, H. and Guerreiro, T. and Jorge, J. and Gonçalves, D. Mobile touchscreen user interfaces: bridging the gap between motor-impaired and able-bodied users. In *Universal Access in the Information Society*, pages 1-11, 2012.

23. Olwal, A. and Feiner, S. and Heyman, S. Rubbing and tapping for precise and rapid selection on touch-screen displays. CHI 2008.

24. Oney, S., Harrison, C., Ogan, A. and Wiese, J. ZoomBoard: a diminutive QWERTY soft keyboard using iterative zooming for ultra-small devices. CHI '13.

25. Pindat, C., Pietriga, E., Chapuis, O. and Puech, C. Jellylens: Content-aware adaptive lenses. UIST 2012.

26. Raman, T. V. and Chen, C. L. and Tim, C. Leveraging Android accessibility APIs to create an accessible experience. Google I/O 2011.

27. Ramos, G. and Cockburn, A. and Balakrishnan, R. and Beaudouin-Lafon, M. Pointing lenses: facilitating stylus input through visual-and motor-space magnification. CHI 2007.

28. Schuster, C. and Von Eye, A. The relationship of ANOVA models with random effects and repeated measurement designs. In *Journal of Adolescent Research*, 16.2, pages 205-220, 2001.

29. Sears, A. and Shneiderman, B. High precision touchscreens: design strategies and comparisons with a mouse. In *International Journal of Man-Machine Studies*, 34.4, pages 593-613, 1991.

30. Sesto, M. E. and Irwin, C. B. and Chen, K. B. and Chourasia, A. O. and Wiegmann, D. A. Effect of touch screen button size and spacing on touch characteristics of users with and without disabilities. In *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 54.3, pages 425-436, 2012.

31. Smith, A. Smartphone ownership–2013 update. In *Pew Research Center: Washington DC*, 2013.

32. Trewin, S. and Keates, S. and Moffatt, K. Developing steady clicks:: a method of cursor assistance for people with motor impairments. ASSETS 2006.

33. Trewin, S., Swart, C. and Pettick, D. Physical accessibility of touchscreen smartphones. ASSETS '13.

34. Wacharamanotham, C. and Hurtmanns, J. and Mertens, A. and Kronenbuerger, M. and Schlick, C. and Borchers, J. Evaluating swabbing: a touchscreen input method for elderly users with tremor. CHI 2011.

35. Wang, F. and Cao, X. and Ren, X. and Irani, P. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. UIST 2009.

36. Wobbrock, J. O. and Myers, B. A. and Kembel, J. A. EdgeWrite: a stylus-based text entry method designed for high accuracy and stability of motion. UIST 2003.

37. Worden, A. and Walker, N. and Bharat, K. and Hudson, S. Making computers easier for older adults to use: area cursors and sticky icons. CHI 1997.

38. Zhong, Y. and Raman, T. V. and Burkhardt, C. and Biadsy, F. and Bigham, J. P. JustSpeak: enabling universal voice control on Android. W4A 2014.