Dynamic UI Adaptations for One-Handed Use of Large Mobile Touchscreen Devices

Daniel Buschek(✉), Maximilian Hackenschmied, and Florian Alt

LMU Munich, Munich, Germany
{daniel.buschek,florian.alt}@ifi.lmu.de, m.hackenschmied@googlemail.com

Abstract. We present and evaluate dynamic adaptations for mobile touch GUIs. They mitigate reachability problems that users face when operating large smartphones or "phablets" with a single hand. In particular, we enhance common touch GUI elements with three simple animated location and orientation changes (*Roll*, *Bend*, *Move*). Users can trigger them to move GUI elements within comfortable reach. A lab study ($N = 35$) with two devices (4.95 in, 5.9 in) shows that these adaptations improve reachability on the larger device. They also reduce device movements required to reach the targets. Participants perceived adapted UIs as faster, less exhausting and more comfortable to use than the baselines. Feedback and video analyses also indicate that participants retained a safer grip on the device through our adaptations. We conclude with design implications for (adaptive) touch GUIs on large devices.

Keywords: UI adaptation · Reachability · One-handed use · Thumb · Touch · Mobile device

1 Introduction

Mobile devices today come equipped with increasingly larger touchscreens. Large screens are attractive for displaying photos, videos and websites. They also promise to ease input: Following Fitts' Law [7], large GUI elements (e.g. buttons, text links) are easier targets than smaller ones.

However, supporting and using a device with the same hand limits touch interaction to the use of the thumb. Thus, interaction with large screens suffers from the thumb's limited reach, both with respect to reachable distance and comfortably bendable angles [1]. Certain screen regions and thus GUI elements cannot be reached comfortably or even not at all.

Several UI adaptations have been proposed to address these problems. A popular approach implemented on devices on the market today shrinks and/or moves the displayed content to cover only that part of the screen which is easy to reach (e.g. "Reachability" on the iPhone[1], Samsung's "One-handed operation"

[1] http://appleinsider.com/articles/14/09/09/how-apple-made-the-iphone-6-and-iphone-6-plus-one-handed-use.

R. Bernhaupt et al. (Eds.): INTERACT 2017, Part III, LNCS 10515, pp. 184–201, 2017.
DOI: 10.1007/978-3-319-67687-6_13

feature[2]). Other concepts change layouts (e.g. for keyboards[3]) or introduce new widgets (e.g. radial menus[4]).

Unfortunately, resizing the displayed content and input areas mitigates the benefits of a larger screen. Special widgets allow to keep the full screen area, but they must be introduced to the user, overlay existing content, or are difficult to integrate well into existing layouts (e.g. radial menu vs common box-layout of apps and websites).

To address these challenges and improve one-handed use, we investigate dynamic adaptations of common GUI elements (Fig. 1). We contribute: (1) four dynamic UI adaptations for three common main elements in mobile touch GUIs, (2) evaluated in a user study with 35 participants in the lab, (3) resulting in insights into one-handed use and relevant design implications for large mobile touchscreen devices.

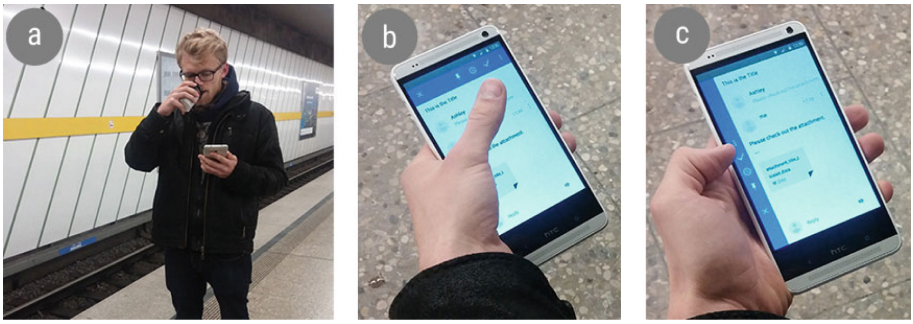

Fig. 1. In everyday life, (a) we often operate mobile devices with one hand. (b) On large devices, some screen regions cannot be reached comfortably. Here, the user struggles to touch the blue action bar at the top. (c) Rotating the bar to the left screen side renders it easily accessible. This may be triggered, for example, by "flicking", i.e. tilting the device to the left.

2 Related Work

The main problem with one-handed touch interaction on large devices is the limited range and flexibility of the human thumb [1]. Large screens may easily exceed the thumb's reachable area, yet one-handed use is often required in many every-day situations, for example when holding onto a rail or carrying other objects [15,23,24]. Users may cope by tilting the device to bring the far screen corner closer to the stretched thumb [5,22]. Besides the industry solutions

[2] http://www.androidcentral.com/how-set-your-galaxy-s5-better-one-handed-use.
[3] https://support.swiftkey.com/hc/en-us/articles/201457382-How-do-I-change-my-keyboard-layout-with-SwiftKey-Keyboard-for-Android-.
[4] https://play.google.com/store/apps/details?id=jun.ace.piecontrol.

mentioned in the introduction, HCI research has proposed various approaches to improve one-handed mobile touch interaction:

TouchShield [12] allows users to bring up a new control widget with their thumb in the screen centre to facilitate a more stable grip. *ThumbSpace* [16, 19] introduced an easily reachable small proxy area to map touches to their corresponding location on the whole screen. Kim et al. [20] enabled users to (1) pan the screen content to move far targets towards the thumb, or to (2) summon a cursor that multiplies the thumb's movements to reach further. They used edge-swiping and "fat" touches to trigger these modes. In contrast, Chang et al. [5] triggered similar methods once the device's tilt indicated reaching for a distant target. While these methods can improve reachability and/or grip stability, many introduce indirect input [5,12,16,19,20], or require extra panning actions [5,20], which can slow down target selection.

Roudaut et al. [25] proposed a two-tap selection method – first triggering magnification of an area of interest, then selecting within the magnified display. They further proposed a magnetic target selection "stick" of sizeable length. Magnification might also move some targets within reach, and the "stick" can extend the thumb, but their goal was to improve target selection on *small* screens and thus the concepts were not designed and evaluated for improving reachability on large devices.

Other related work designed task-specific widgets for one-handed use, such as a radial contact list [13], an interface for video browsing [14], or an app-launcher [18]. While these designs can mitigate problems with reachability and precision, they are limited to their specific use-cases and thus in general not applicable across different tasks and applications.

Involving the fingers of the grasping hand on the back of the device [26–28] can also address reachability issues. However, this requires additional touch sensors on the back. Similarly, concepts envisioning a bendable device [8] cannot be realised with current off-the-shelf hardware.

Recent work has predicted front screen touches *before* the thumb hits the screen, based on grip changes during the reaching movement, registered with (1) back-of-device touch sensors [21] or with (2) device motion sensors [22]. While such methods could also be used to predict touch locations that users could actually never reach, it seems unlikely that users would even try to stretch towards obviously unreachable targets in order to enable such predictions in the first place.

Another line of research adapted underlying touch areas of on-screen keyboards to typing behaviour and hand posture without visual changes [4,9,11,29]. However, the main concern of these projects was typing precision, not reachability. In contrast, Cheng et al. [6] visually adapted keyboard shape and location to the user's grasp on a tablet, also to facilitate reachability. Similarly, we also follow the idea of adapting the location and shape of GUI elements to better suit the user's hand posture. Instead of keyboards for two-handed typing on tablets, we address main navigation elements and action buttons for one-handed use of large phones or "phablets".

In summary, related work (1) examined the thumb's limited reach [1], (2) motivated the support of one-handed use [15,23,24], and (3) proposed solutions which were either applied to the whole interface [5,16,19,20] or introduced new UI elements [5,13,14,18,20,25] and hardware [8,26–28]. In contrast, we investigate how one-handed use of mobile touch devices can be improved by adapting existing main GUI elements. In particular, we are interested in simple and easily understandable changes to UI layout, location and orientation.

3 Concept Development

Our design goal is to improve reachability of existing GUI elements – in contrast to related work that often invented new (task-specific) widgets. We decided to explore adaptations of the main elements of Google's Material Design[5] as an example of a popular modern design language for mobile touch GUIs. The following subsections describe our concept development process.

3.1 Brainstorming Session

We conducted a brainstorming session with colleagues and students from an HCI lab to generate ideas. No default GUI was given; rather, the task was to freely come up with ideas for adapting common mobile touch GUI elements to improve reachability in one-handed use. Clustering the ideas revealed design dimensions for adaptations: changing layouts globally (i.e. in GUI) and locally (i.e. within menu); changing alignment, orientation, shape, item arrangement; floating elements; and adding new elements.

3.2 Paper Prototyping

The generated ideas were captured as simple GUI sketches on paper, and shown to colleagues and students to gain early feedback. Ideas with moving elements were liked best overall. However, this step also revealed that, for further feedback, we needed to go beyond paper sketches.

3.3 App Prototyping

Hence, we moved from paper to phone. Integrating feedback from the discussions, we created click-through prototypes to be able to demonstrate the refined ideas on an actual device. We used the prototyping software *Sketch*[6] and *POP*[7]. We created 38 interactive mockup screens (like the ones in Fig. 2), showing the concepts embedded into a fake email client to give them meaningful application context.

[5] https://www.google.com/design/spec/material-design/, *last accessed 23rd Jan. 2017.*
[6] https://www.sketchapp.com/, *last accessed 22nd Jan. 2017.*
[7] https://popapp.in/, *last accessed 22nd Jan. 2017.*

3.4 Pre-study

We reduced the number of ideas to a feasible amount for a small study, based on another round of feedback from colleagues and guided by the previously identified design dimensions. We kept the six concepts shown in Fig. 2.

We employed them in a small user study to gather qualitative feedback on our ideas. We recruited four participants (three female, mean age 26, all right-handed). They were compensated with a €5 gift card for an online shop.

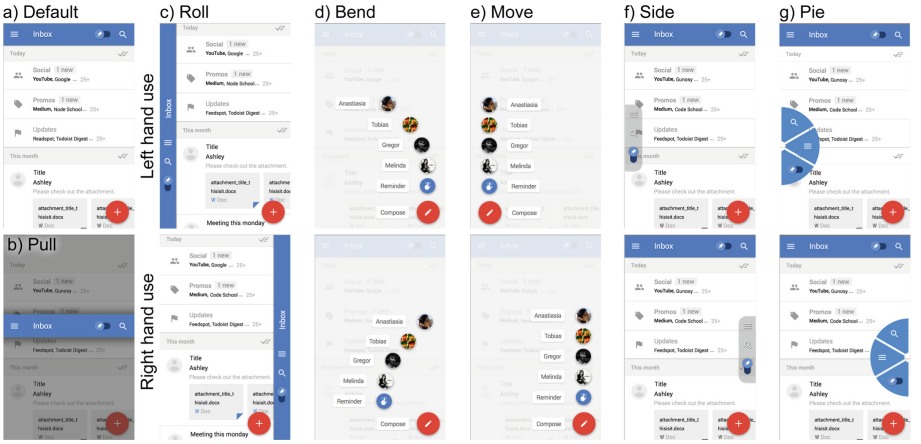

Fig. 2. Mockups showing (a) the unadapted UI, and the six adaptations evaluated in the pre-study: (b) *Pull:* users can pull down the action bar, it moves back up after using one of its buttons; (c) *Roll:* the action bar rotates around the screen corners into a vertical layout close to the holding hand; (d) *Bend:* the menu items bend to match the thumb's reachable angles; (e) *Move:* the button/menu is moved to the side of the holding hand; (f) *Side* and (g) *Pie:* redundant action bars can be swiped in from the screen edge. The top row shows the left-hand versions, the bottom row the ones for the right hand (exception: (b) *Pull* is pulled down for both hands). *Roll*, *Bend* and *Move* were selected for the final study.

Each participant tested all six concepts in random order. Participants were asked to fulfil simple tasks: opening an email, composing a new mail, deleting one. They were encouraged to think aloud during the tasks. After completing each task, they rated the current concept on a five-point Likert scale regarding eight items on aspects such as reachability, ease-of-use, understandability and distraction. In addition, they shared their thoughts verbally.

3.5 Selection of Concepts

The study revealed problems for: (a) *Pull* - pulling down the action bar merely reduces vertical distance to targets. Horizontally distant targets are still problematic. (b) *Side* - feedback revealed the importance of clear distinction between

content and (adaptive) controls, which was problematic due to the transparency of *Side*. (c) *Pie* - Some participants were confused about redundant GUI elements. This was the case for both *Side* and *Pie*, which duplicate the action bar without hiding it. Hence, we decided against duplication. All other concepts received promising ratings and feedback and were thus selected for the main study: *Rolling Action Bar (Roll)*, *Moving Action Button/Menu (Move)*, and *Bending Action Menu (Bend)*

3.6 Final Concepts: Dynamic Adaptive UI

In summary, we propose four adaptations and an example trigger action.

Rolling Action Bar (Roll). An *Action Bar* is located at the top of most Android GUIs. It features buttons for navigation or main functionality in the current view. Triggering our adaptation rotates the bar around the screen corner (animated), changing it from its default location at the top to a left/right-aligned layout (see Fig. 2c). Triggering adaptation again moves it back to the default location at the top.

Moving Action Button/Menu (Move). The *Floating Action Button* is a single button "floating" on top of the view, often in the bottom right corner (see Fig. 2a). Our adaptation makes it movable: When triggering the adaptation, the button moves over to the left side of the screen (see Fig. 2e). This makes it easier to reach in left-hand use. Triggering the adaptation again moves it back to the right.

There is also a menu-version of this button, *Floating Action Menu*. Touching this menu button opens a floating menu (see Fig. 2e). As with the button, adaptation moves it to the other screen side, triggering again moves it back.

Bending Action Menu (Bend). Instead of moving the *Floating Action Menu*, this concept adapts the arrangement of its menu items. While the normal version displays menu items in a straight line, triggering our adaptation moves them into a curve to better fit the thumb's reachable area (see Fig. 2d). Repeated adaptation moves them back into a straight line.

Triggering Adaptations. Adaptations could be triggered explicitly or automatically. For our study, we implemented a simple explicit trigger: tilting the device in a short wrist turn (Fig. 3b). We decided not to use a touch gesture for the study to keep the trigger clearly distinguishable from the interactions, so that people could easily report feedback on adaptations and trigger separately. Tilting may also go well with the idea of movable elements – users can "flick" the UI elements to new locations. However, this trigger is an example, not part of our contribution.

4 User Study

To evaluate our adaptations, we conducted a repeated measures lab study. The independent variables were *device* (Nexus 5, HTC One Max), *hand* (left, right), and *concept* (four adaptive versions, plus the baseline versions of the three UI elements). We measured *completion time*, *device orientation*, and *reachability*, as well as user opinions on five-point Likert items.

4.1 Participants

We recruited 35 participants, mostly students, via a university mailing list and social media. 17 were female, 6 were left-handed. The average age was 24.7 years (range: 19–47). They received a €10 gift card for an online shop as compensation.

4.2 Apparatus

We used two devices to cover a range of interesting (i.e. "large") screen sizes, namely an LG Nexus 5 (4.95 in screen, $137.9 \times 69.2 \times 8.6$ mm), and an HTC One Max (5.9 in screen, $164.5 \times 82.5 \times 10.3$ mm). Our study app (Fig. 3c and d) showed the tested GUI elements. Menu elements had labelled buttons (e.g. "C1" to "C6", see Fig. 3c). The floating action menus/buttons were located near the bottom right screen corner (see Fig. 3d). A video camera captured the study, focusing on device and hands. Figure 4 shows a few selected scenes.

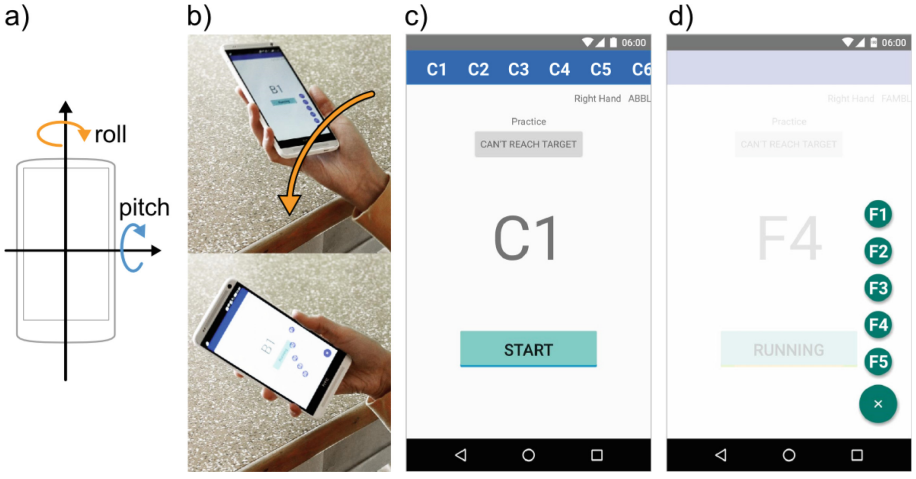

Fig. 3. This figure shows (a) roll and pitch measured relative to the device, (b) the trigger gesture, and example screens from the study app, (c) an *Action Bar* trial, and d) a running *Floating Action Menu* trial.

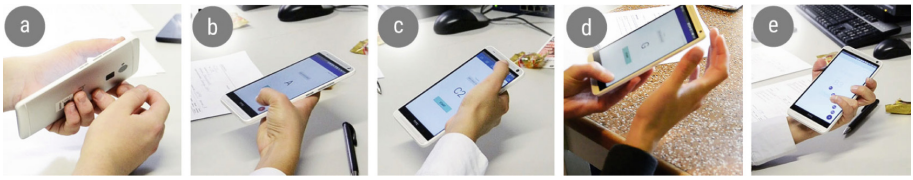

Fig. 4. Coping strategies: (a) tilting the device, (b) and (c) shifting the grip depending on the target location, (d) unstable grips, bringing the second hand close to "catch" the device if necessary, and (e) further strategies, such as using other fingers, reaching around the device.

4.3 Procedure

Each participant used both the left and right hand with both devices. For each device and hand, each participant completed seven tasks, namely using the four adaptations, plus the non-adaptive baseline versions of the three GUI elements. The order was varied using a Latin Square design. We explain the procedure by describing its three hierarchical components: tasks, trial sequences, and trials.

Tasks. Each task covered one GUI element and adaptation concept. We first explained the tested element and its adaptation, if available. A training trial sequence allowed participants to familiarise themselves with the task. They then completed six trial sequences.

Trial Sequences. Each trial sequence consisted of subsequently hitting five targets (for the *Floating Action Menu/Button*) or six targets (for the *Action Bar*). For the adaptive versions, adaptation had to be triggered at the start of the trial sequence; the element then stayed in its adapted state until the start of the next trial sequence. This procedure allowed us to analyse how many subsequent interactions were needed to compensate for trigger overhead (i.e. time required to trigger adaptation vs time saved by using the adapted element).

Trials. To start each trial within a sequence, participants had to touch a *"Start"* button with their thumb. The trial was over once they either had hit their target or a *"Can't reach"* button, to indicate that it was not possible for them to hit the target.

Two GUI elements featured multiple targets – the six buttons on the *Action Bar*, and the five menu options of the *Floating Action Menu*. In these cases, the app displayed the target label for the current trial in the screen centre (see Fig. 3c and d).

Subjective Ratings and Feedback. After each task, participants filled in a short questionnaire with five-point Likert items on perceived speed, comfort

of reach, safe grip, and exhaustion. At the very end, qualitative feedback was gathered in a short semi-structured interview. Here, we also asked for opinions on explicitly triggered adaptations (i.e. manual trigger, as in the study) and implicit ones (i.e. automatic trigger, e.g. with a hand posture recognition system).

4.4 Limitations

Our lab study provides control and direct observation. However, future work should also evaluate acceptance in a field study over a longer period of time. Moreover, we chose a non-touch trigger (Fig. 3b) to keep it clearly separate from main interactions. Feedback suggests that turning a larger device in this way is not ideal for everyday use. Other triggers can be investigated, for example a touch swipe from the edge, a large touch [20], long touch, or implicit triggers. Finally, moving elements is only one way of adapting touch GUIs; our study does not include a comparison to other approaches, which we leave for future work.

5 Results

Results are reported per adaptation in comparison to the corresponding non-adaptive baseline element. For each element, we report on task completion time, device orientation, and reachability. Significance, tested where applicable, is reported at the 0.05 level. Further aspects of our evaluation and report are described as follows:

We first focus on data from the larger device (HTC One Max), and then provide a summarised comparison to the results obtained on the other device (Nexus 5). To evaluate the adaptations, we first analyse the data without the trigger. We then report on the influences of the trigger in a separate section. The elements' default versions are their right hand versions for all but the action bar (*Roll*) and the bending menu (*Bend*). Hence, if not stated otherwise, we report on the interesting *adaptation to the left hand*. We removed outliers (e.g. a participant interrupting the study in the middle of a trial).

Table 1 gives an overview, explained in detail in the following subsections.

5.1 Action Button – Fixed vs Move

We compared the fixed *Action Button* against the adaptive version (*Move*).

Time: Using the adapted version was significantly faster than the baseline ($t(28) = 3.11, p < .01$). Note df = 28 in this test, since three people decided to never use adaptation, and three others could not reach the baseline with their left hand at all.

Device Orientation: We found significantly less roll (Fig. 3a) with *Move* than in the baseline ($t(28) = 3.82, p < .01$).

Table 1. Results overview. The table shows the observed time and device roll when hitting the given targets on the HTC One Max, for both unadapted and adapted versions (adapted to left hand). Note that menu and action bar had multiple targets (i.e. menu items), hence we give ranges of values covering the target-specific values for these elements. We also measured pitch, but omit it here, since no significant differences were found for any element.

Element	Concept	Time (s)		Orientation (deg roll)	
		mean	sd	mean	sd
Action button	Unadapted	0.63	0.65	−11.70	16.28
Action button	Move	0.40	0.12	−0.71	4.10
Action menu	Unadapted	1.77–2.10	0.88–1.35	−23.87–−17.67	18.81–19.71
Action menu	Move	1.30–1.43	0.21–0.35	−5.97–2.92	3.94–5.85
Action menu	Bend	1.85–2.03	0.85–1.58	−13.87–−12.61	11.06–12.51
Action bar	Unadapted	0.72–2.35	0.20–2.25	−43.86–−19.75	14.67–18.29
Action bar	Roll	0.58–0.77	0.18–0.29	−10.94–−1.93	5.31–7.41

Reachability: Three people could not reach the baseline button with their left thumb. Observations and video analyses showed that many people required considerable effort and grip changes. *Move* could easily be reached by everyone.

5.2 Action Menu – Fixed vs Move or Bend

We compared the baseline *Action Menu* with two menu adaptations, moving (*Move*) and bending (*Bend*). All three menus featured five menu buttons as targets (Fig. 3d).

Time: Using the adapted version was only significantly faster than the baseline for moving the menu ($t(28) = 4.19, p < .001$), but not for bending it.

Device Orientation: We found significantly less roll for both adaptations than with the baseline (*Move:* $t(28) = 4.93, p < .001$; *Bend:* $t(29) = 3.72, p < .001$).

Reachability: Five people could not open the baseline menu with their left hand. The closed bended menu at the same location could not be reached by four people. This difference is explained by the coping strategies that some people invented but only employed in some cases. No reachability problems occurred for the movable menu.

5.3 Action Bar – Fixed vs Roll

We compared the static baseline *Action Bar* with the adaptive version (*Roll*).

Time: Using the adapted version (*Roll*) was significantly faster than the baseline for both the left hand ($t(34) = 6.12, p < .001$) and right hand ($t(34) = 7.36, p < .001$). We also found positive correlations between time taken and the reaching distance, that is the distance from the target to the screen edge of the holding hand (left hand: r = .445; right hand: r = .314). Standard deviation of time also correlated positively with reaching distance (left hand: r = .925; right hand: r = .788), indicating less controlled movements for further reaching.

Device Orientation: The adapted version had significantly less device roll than the baseline for both the left hand ($t(34) = 16.28, p < .001$) and right hand ($t(34) = 17.23, p < .001$).

Reachability: Many people struggled to reach the outer targets of the baseline action bar: Nine could not reach "6" and three could not reach "5" with their left hand (labels see Fig. 3c). With the right hand, seven could not reach "1" and two could not reach "2". *Roll* enabled everyone to reach all targets.

5.4 Trigger Overhead

Adaptations require time for trigger, animation, and users' reorientation. Repeated use of adapted elements might compensate for this. For our trigger we observed the following results:

The *Bending Action Menu* could not compensate for the trigger, taking up to 3.03 s longer than the baseline after the fifth target. The *Moving Action Button* almost reached zero, taking up to 0.43 s longer after the fifth target. The *Moving Action Menu* compensated for the trigger after three targets, leading to a mean advantage of 1.07 s after five targets. The *Rolling Action Bar* compensated for this after four interactions, leading to a mean advantage of 1.23 s after six targets. These values relate to our main adaptation scenario (default to left hand). We observed similar yet less pronounced results for right-hand adaptations.

5.5 Comparison of Screen Sizes

We have focused on the results from the larger device so far (HTC One Max, 5.9 in screen), since such large devices are the main target of our adaptations. To further evaluate benefits and limitations, we now compare the results to those obtained on the smaller device (LG Nexus 5, 4.95 in):

We found almost no effect of screen size on time with the adapted elements. However, the baseline elements strongly profited by the smaller screen. In consequence, the speed improvements achieved by using adapted elements almost disappeared on the smaller device. The smaller device required slightly less tilting than the larger one. Adapted elements still caused less tilting than non-adaptive ones. Reachability greatly improved on the smaller device. However, one person could still not reach the outermost target of the unadapted baseline *Action Bar*.

5.6 Reachability Coping Strategies

Our observations and video recordings revealed several coping strategies to deal with far targets: A main strategy was tilting the device, sometimes up to 90 degrees (Fig. 4a). Another common strategy was to move the hand along the edge of the device (Fig. 4b and c). This requires users to loosen their grip on the device. As a rough estimate from live/video observations, on average they moved approximately 2 cm along the edge to reach targets at the very top (on the HTC One Max). Some even moved the hand to the device's bottom, to enable their thumbs to better cover the full screen width. This was difficult and people often brought their second hand closer, ready to catch the device in case of a slip (Fig. 4d). Two tried reaching around the device with their fingers (Fig. 4e). This required considerable efforts and was reported as tiresome. These participants also stated that they probably would have involved their second hand instead in real-life use.

5.7 Subjective User Ratings

After each task, people rated the used element with four Likert items. All adaptive elements were perceived as faster, more comfortable to use, less exhausting to reach, and more grip safe, compared to their baselines for use with the left hand on the larger device. These differences were significant for all concepts apart from *Bend* (Bonf.-corrected Wilcoxon signed rank tests, all $p < .05$). For the right hand, *Roll* was also preferred over the baseline in all questions, since it had a right-hand adaptation as well (Fig. 2). Ratings for "It is exhausting to reach this element" had a moderate positive linear relationship with roll ($r = .511$). We also found weak to moderate negative relationships between roll and both comfort ($r = -.497$) and safe grip ($r = -.425$). On the smaller device, people did not feel at disadvantage with static GUIs, yet adapted UIs still had an advantage for comfort.

6 Discussion and Implications

6.1 Can Dynamic UI Adaptations Improve Reachability?

On the larger device, 25% of participants could not reach the (unadapted) *Action Bar*, 15% the *Floating Action Menu*, and 9% the *Floating Action Button*. Others struggled to reach some targets. In general, the thumb's functional area depends on the grip [1], which we did not keep fixed to avoid unnatural use. Our adaptations moved elements into the centre (*Bend* when opened), or closer to the thumb (*Move*, *Roll*), thus making them generally easier to reach.

Crucially, our adaptations enabled all participants to reach all targets. The only exception was the bended menu (*Bend*), which did not move the menu button itself and thus could not improve reachability for an unopened menu. Apart from this exception, our adaptations greatly improved reachability since they brought all targets into reach for everyone in our study.

Feedback, thinking-aloud, and Likert ratings further revealed that the majority of participants also found the adapted GUI elements less exhaustive and more comfortable to use.

Reachability was much less of an issue on a 4.95 in. device compared to the 5.9 in. one. In conclusion, we see the main applications of our adaptations for reachability improvements on the very large end of smartphones, as well as on "phablets".

6.2 Can Dynamic UI Adaptations Improve Speed?

The results show that our adaptations improve speed under certain circumstances. Firstly, adapted elements improved speed on the large device, yet this effect was marginal on the smaller one. This indicates that our adaptations are mainly beneficial on devices beyond the five inch mark. Secondly, the trigger plays an important role. Two of four adaptations could compensate for the $\approx$1.74 s overhead caused by our example trigger. Moving a single button or changing the shape of a menu could not compensate for the trigger, but relocating whole menus saved time after three to six interactions. We argue that such numbers of interactions without switching hand posture occur in many use-cases involving an action bar or menu, such as a browsing session.

Once adapted, all but one element (*Bend*) significantly shortened interaction time on the larger device. The exception for *Bend* is explained by the fact that the bended menu must still first be opened by reaching the menu button, as in the unadapted version.

Participants also subjectively rated these elements significantly faster than the baselines. Even moving a single important button could be worthwhile with a fast or automatic trigger (e.g. by inferring hand postures from preceding touch behaviour [2,10,29]). Finally, all adaptations were subjectively perceived as faster than the baselines where they mattered most, namely on the larger device when used with the left hand.

6.3 Can Dynamic UI Adaptations Facilitate a Safer Grip?

Device roll was significantly lower for elements adapted to the left hand, compared to the baselines. Pitch showed a similar yet non-significant tendency. The action bar's adaptation (*Roll*) also significantly reduced roll for the right hand. Less variance in targeting times suggests that *Roll* also resulted in more control, especially for the non-dominant hand. Since greater device tilting and movements may increase the risk of letting it slip, these results suggest that adaptations can facilitate a safer grip.

This conclusion is supported by participants' own perceptions: Regarding their grip on the device, they perceived the adapted elements as significantly safer than the baselines. This was revealed by the Likert ratings and was in line with further verbal feedback. Recorded sensor values and users' Likert ratings were also moderately correlated ($r = .4$ to $.5$): Less device tilt was associated with higher ratings on comfort and grip, and lower ratings on exhaustion.

6.4 Do Users Accept Dynamic UI Adaptations?

All adaptations received more positive ratings than the baselines. Most people elected the *Bending Action Menu* as their favourite adaptive UI element. Interestingly, this was the only adaptation that could not improve speed, even without the trigger overhead. However, people's comments suggest that they liked the feeling of "ergonomic optimisation". This was probably more visibly conveyed in the bended arrangement of the menu items than in location/rotation changes. Moreover, the bended layout matches the more comfortable movement directions, in which the thumb describes an arc instead of bending and extending (see [17]).

In conclusion, all GUI adaptations were well accepted by our participants. When asked about explicit/implicit triggers after the study, some participants said that they liked the explicit influence over the UI, yet the majority stated that they would prefer automatic adaptation, if it worked reliably.

6.5 Design Implications

We expect several observations, insights, and lessons learned from this project to be useful for designing adaptive touch GUIs for large devices beyond this work:

Reconsider Default Locations: Our baselines showed that users of larger devices can face serious reachability problems, even for UIs following a modern touch GUI design language. On devices beyond five inches and without dynamic adaptations, navigation and action bars would be easier to reach at the bottom of the screen, and action buttons can improve reachability for both hands by being centred horizontally.

Separate Controls from Content: Our pre-study revealed that it is difficult for users to understand and follow adaptive UI elements that are less clearly separated from the main content (e.g. due to transparency or overlays).

Avoid Introducing Redundancy: In our concept discussions and pre-study, we found that duplicating UI elements at easier reachable locations caused confusion due to the redundancy. Participants had to (1) understand that functionality had been duplicated and (2) mentally draw a connection between the new element and the normal one.

Make Explicit Adaptations "Sticky": We suggest that if adaptations are triggered by the user, they should not revert themselves after a single interaction. First, based on our speed and trigger-overhead analyses, keeping elements in their adapted state for subsequent actions helps to compensate for the time required by the trigger. Second, participants who would prefer an explicit trigger outside of the study indicated that they liked the feeling of control and changing the UI on their own command.

Adapt View Groups: Based on our analyses of trigger-overhead against saved time, we suggest to relocate multiple buttons at once (e.g. as in our moving menu and rotating action bar). This improves convenience and helps to compensate for trigger overhead, since users are then more likely to already find future actions within reach.

An "Ergonomic Look" Can Facilitate User Acceptance: Although bending the menu did not improve the quantitative measures, it looked ergonomically fitting for the thumb and the majority of participants highlighted this adaptation as their favourite.

6.6 Integration of Adaptations into Existing GUIs

Our non-adapted elements are simply the familiar elements from Material Design, not new ones. This makes it easy to integrate them into existing apps - simply replace the old version with an adaptive one; the visuals stay the same. On the other hand, this may make it hard for users to discover the new adaptive functionality. Ideally, the presented adaptations would be integrated into existing mobile interfaces by making them available on an operating system level. Thus, for example, all action bars would be adaptive ones. In contrast, integration into single apps might confuse users, since the GUI's behaviour then becomes less consistent across applications.

These issues can be addressed by explaining new adaptations to the user, for example with an overlay upon first use, similar to the commonly used app-introduction screens. More generally, new adaptiveness in any app or on OS level should be revealed and explained to avoid unexpected GUI behaviour from the users' point of view.

Finally, from a practical perspective, developers can realise adaptations like the ones presented here by making use of generalised frameworks that facilitate implementing adaptive GUIs (e.g. see [3]).

7 Conclusion and Future Work

While large mobile devices are attractive for displaying multimedia content, they also introduce reachability problems when users need to operate them with a single hand as required in many every-day situations. Some screen regions cannot be reached comfortably or not at all.

We have proposed dynamic adaptations of basic touch GUI elements to mitigate reachability problems for one-handed use of large mobile devices. A lab study ($N = 35$) showed that adapted elements improve reachability on a large device, and can reduce interaction time and device tilting. Moreover, using adaptations resulted in perceived higher comfort, less exhaustion, and a safer grip. We further derived lessons learned and implications for future design of adaptive mobile touch GUIs for large devices.

To take full advantage of dynamic GUI adaptations in practice, the next step is to investigate better triggers, including automatic ones. For example, adaptations could be triggered based on automatically inferred hand postures (e.g. [2,3,10,29]).

References

1. Bergstrom-Lehtovirta, J., Oulasvirta, A.: Modeling the functional area of the thumb on mobile touchscreen surfaces. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2014, pp. 1991–2000. ACM, New York (2014). http://doi.acm.org/10.1145/2556288.2557354
2. Buschek, D., Alt, F.: TouchML: a machine learning toolkit for modelling spatial touch targeting behaviour. In: Proceedings of the 20th International Conference on Intelligent User Interfaces, IUI 2015, pp. 110–114. ACM, New York (2015). http://doi.acm.org/10.1145/2678025.2701381
3. Buschek, D., Alt, F.: ProbUI: generalising touch target representations to enable declarative gesture definition for probabilistic GUIs. In: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems, CHI 2017, pp. 4640–4653. ACM, New York (2017). http://doi.acm.org/10.1145/3025453.3025502
4. Buschek, D., Schoenleben, O., Oulasvirta, A.: Improving accuracy in back-of-device multitouch typing: a clustering-based approach to keyboard updating. In: Proceedings of the 19th International Conference on Intelligent User Interfaces, IUI 2014, pp. 57–66. ACM, New York (2014). http://doi.acm.org/10.1145/2557500.2557501
5. Chang, Y., L'Yi, S., Koh, K., Seo, J.: Understanding users' touch behavior on large mobile touch-screens and assisted targeting by tilting gesture. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, pp. 1499–1508. ACM, New York (2015). http://doi.acm.org/10.1145/2702123.2702425
6. Cheng, L.P., Liang, H.S., Wu, C.Y., Chen, M.Y.: iGrasp: grasp-based adaptive keyboard for mobile devices. In: CHI 2013 Extended Abstracts on Human Factors in Computing Systems, CHI EA 2013, pp. 2791–2792. ACM, New York (2013). http://doi.acm.org/10.1145/2468356.2479514
7. Fitts, P.M.: The information capacity of the human motor system in controlling the amplitude of movement. J. Exp. Psychol. **47**(6), 381–391 (1954)
8. Girouard, A., Lo, J., Riyadh, M., Daliri, F., Eady, A.K., Pasquero, J.: One-handed bend interactions with deformable smartphones. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, pp. 1509–1518. ACM, New York (2015). http://doi.acm.org/10.1145/2702123.2702513
9. Goel, M., Jansen, A., Mandel, T., Patel, S.N., Wobbrock, J.O.: Contexttype: using hand posture information to improve mobile touch screen text entry. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2013, pp. 2795–2798. ACM, New York (2013). http://doi.acm.org/10.1145/2470654.2481386
10. Goel, M., Wobbrock, J., Patel, S.: Gripsense: using built-in sensors to detect hand posture and pressure on commodity mobile phones. In: Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology, UIST 2012, pp. 545–554. ACM, New York (2012) http://doi.acm.org/10.1145/2380116.2380184

11. Gunawardana, A., Paek, T., Meek, C.: Usability guided key-target resizing for soft keyboards. In: Proceedings of the 15th International Conference on Intelligent User Interfaces, IUI 2010, pp. 111–118. ACM, New York (2010) http://doi.acm.org/10.1145/1719970.1719986
12. Hong, J., Lee, G.: Touchshield: a virtual control for stable grip of a smartphone using the thumb. In: CHI 2013 Extended Abstracts on Human Factors in Computing Systems, CHI EA 2013, pp. 1305–1310. ACM, New York (2013) http://doi.acm.org/10.1145/2468356.2468589
13. Huot, S., Lecolinet, E.: Spiralist: a compact visualization technique for one-handed interaction with large lists on mobile devices. In: Proceedings of the 4th Nordic Conference on Human-computer Interaction: Changing Roles, NordiCHI 2006, pp. 445–448. ACM, New York (2006). http://doi.acm.org/10.1145/1182475.1182533
14. Hürst, W., Merkle, P.: One-handed mobile video browsing. In: Proceedings of the 1st International Conference on Designing Interactive User Experiences for TV and Video, UXTV 2008, pp. 169–178. ACM, New York (2008). http://doi.acm.org/10.1145/1453805.1453839
15. Karlson, A.K., Bederson, B.B.: Studies in one-handed mobile design: habit, desire and agility. In: Proceedings of the 4th ERCIM Workshop on User Interfaces for All, UI4ALL 1998. Technical report (2006)
16. Karlson, A.K., Bederson, B.B.: One-handed touchscreen input for legacy applications. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2008, pp. 1399–1408. ACM, New York (2008). http://doi.acm.org/10.1145/1357054.1357274
17. Karlson, A.K., Bederson, B.B., Contreras-Vidal, J.L.: Understanding single-handed mobile device interaction. In: Handbook of Research on User Interface Design and Evaluation for Mobile Technology, pp. 86–101 (2007)
18. Karlson, A.K., Bederson, B.B., SanGiovanni, J.: Applens and launchtile: two designs for one-handed thumb use on small devices. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2005, pp. 201–210. ACM, New York (2005). http://doi.acm.org/10.1145/1054972.1055001
19. Karlson, A.K., Bederson, B.B.: ThumbSpace: generalized one-handed input for touchscreen-based mobile devices. In: Baranauskas, C., Palanque, P., Abascal, J., Barbosa, S.D.J. (eds.) INTERACT 2007. LNCS, vol. 4662, pp. 324–338. Springer, Heidelberg (2007). doi:10.1007/978-3-540-74796-3_30
20. Kim, S., Yu, J., Lee, G.: Interaction techniques for unreachable objects on the touchscreen. In: Proceedings of the 24th Australian Computer-Human Interaction Conference, OzCHI 2012, pp. 295–298. ACM, New York (2012). http://doi.acm.org/10.1145/2414536.2414585
21. Mohd Noor, M.F., Ramsay, A., Hughes, S., Rogers, S., Williamson, J., Murray-Smith, R.: 28 frames later: predicting screen touches from back-of-device grip changes. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2014, pp. 2005–2008. ACM, New York (2014) http://doi.acm.org/10.1145/2556288.2557148
22. Negulescu, M., McGrenere, J.: Grip change as an information side channel for mobile touch interaction. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, CHI 2015, pp. 1519–1522. ACM, New York (2015). http://doi.acm.org/10.1145/2702123.2702185
23. Ng, A., Brewster, S.A., Williamson, J.H.: Investigating the effects of encumbrance on one- and two- handed interactions with mobile devices. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2014, pp. 1981–1990. ACM, New York (2014). http://doi.acm.org/10.1145/2556288.2557312

24. Oulasvirta, A., Bergstrom-Lehtovirta, J.: Ease of juggling: studying the effects of manual multitasking. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2011, pp. 3103–3112. ACM, New York (2011). http://doi.acm.org/10.1145/1978942.1979402
25. Roudaut, A., Huot, S., Lecolinet, E.: Taptap and magstick: improving one-handed target acquisition on small touch-screens. In: Proceedings of the Working Conference on Advanced Visual Interfaces, AVI 2008, pp. 146–153. New York (2008). http://doi.acm.org/10.1145/1385569.1385594
26. Wigdor, D., Forlines, C., Baudisch, P., Barnwell, J., Shen, C.: Lucid touch: a see-through mobile device. In: Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology, UIST 2007, pp. 269–278. ACM, New York (2007). http://doi.acm.org/10.1145/1294211.1294259
27. Wolf, K., McGee-Lennon, M., Brewster, S.: A study of on-device gestures. In: Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services Companion, MobileHCI 2012, pp. 11–16. ACM, New York (2012). http://doi.acm.org/10.1145/2371664.2371669
28. Yang, X.D., Mak, E., Irani, P., Bischof, W.F.: Dual-surface input: augmenting one-handed interaction with coordinated front and behind-the-screen input. In: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services, MobileHCI 2009, pp. 5:1–5:10. ACM, New York (2009). http://doi.acm.org/10.1145/1613858.1613865
29. Yin, Y., Ouyang, T.Y., Partridge, K., Zhai, S.: Making touchscreen keyboards adaptive to keys, hand postures, and individuals: a hierarchical spatial backoff model approach. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2013, pp. 2775–2784. ACM, New York (2013). http://doi.acm.org/10.1145/2470654.2481384