Hover-Based Reachability Technique for Executing Single-Touch Gesture on Smartphone

Ryo Ikeda
University of Tsukuba
Japan
ikeda@iplab.cs.tsukuba.ac.jp

Kyohei Hakka
University of Tsukuba
Japan
hakka@iplab.cs.tsukuba.ac.jp

Buntarou Shizuki
University of Tsukuba
Japan
shizuki@cs.tsukuba.ac.jp

ABSTRACT

When operating a smartphone with only one hand, users typically use their thumb; however, there may be areas on the screen that the thumb cannot reach. Several reachability techniques have been developed to enable users to access such areas, but many of these techniques are designed to enable users to perform only a tap gesture, so other single-touch gestures such as long-tap, double-tap, swipe, and drag cannot be performed. To enable all single-touch gestures for unreachable areas, we designed a hover-based reachability technique that uses a cursor. In addition, we conducted a pilot study to compare user performance with our technique and that of existing techniques, One-Handed Mode (OM) and Event Forward Cursor (EC). The results showed that our technique was faster than EC, which uses the same cursor technique as ours, except for the double-tap gesture when the target was small. However, our technique was slower than OM, which is the technique that shrinks the entire screen. Lastly, we discuss possible improvements of our technique on the basis of the results.

CCS CONCEPTS

• **Human-centered computing → Human computer interaction (HCI)**; Interaction techniques; Gestural input;

KEYWORDS

reachability, touchscreens, mobile device, hover sensing

ACM Reference Format:
Ryo Ikeda, Kyohei Hakka, and Buntarou Shizuki. 2021. Hover-Based Reachability Technique for Executing Single-Touch Gesture on Smartphone. In *Asian CHI Symposium 2021 (Asian CHI Symposium 2021), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3429360.3468171

1 INTRODUCTION

Smartphone users often hold their devices in one hand and operate it using their thumb (i.e., one-handed operation) because of the users' preference or state (e.g., carrying a bag or holding an umbrella with the other hand). In one-handed operation, users need to change their grip on the smartphone frequently because they cannot comfortably reach all parts of the screen otherwise [3, 17]. However, frequent grip changes increase the device motion and thus make the device grip insecure [8]. Moreover, the insecure device grip may lead to the user accidentally dropping the smartphone.

Many techniques have been proposed to enable users to reach all parts of the screen with only one hand (e.g., [2, 14, 23, 24]). Most of these techniques are designed to enable users to perform only a tap gesture to unreachable targets. In other words, users cannot perform other single-touch gestures, e.g., long-tap, double-tap, swipe, and drag, which are performed using only one finger. These single-touch gestures are frequently used when operating smartphones, and they are also used in the unreachable areas. For example, users can long-tap the home icon or the text to display the context menu, double-tap the screen to rewind or fast forward a video, swipe down from the top of the screen to view notifications, or drag the caret to select text. Thus, it is important to design a technique that enables users to perform all single-touch gestures on all parts of the screen without having to change the grip.

In this paper, we present a hover-based cursor technique (Fig. 1) to enable users to perform all single-touch gestures on unreachable areas without having to change their grip. Our technique uses a smartphone with a hover-sensing touchscreen, which can sense the finger hovering above the screen (hover area) and track the x and y coordinates of the finger in the area. Users first move their thumb out of the hover area (Fig. 1a) and then back into the hover area within 300 ms (Fig. 1b) to activate the cursor. Then users move their thumb in the hover area to move the cursor to the unreachable target (Fig. 1c). To transfer a single-touch gesture to the cursor position, users perform a standard single-touch gesture; in the case of a tap, for example, users would tap the screen with their thumb (Fig. 1d) and then lift their thumb away. After performing a single-touch gesture, users can continuously operate with the cursor by keeping their thumb in the hover area (Fig. 1e). By moving their thumb out of the hover area, users can deactivate the cursor and resume standard touch operation (Fig. 1f).

We also conducted a pilot study to investigate the preliminary user performance with our technique. In this study, we compared our technique with two existing techniques that enable users to perform all single-touch gestures, One-Handed Mode (OM) and Event Forward Cursor (EC).

2 RELATED WORK

We designed a hover-based reachability technique. In this section, we discuss related work on reachability and hover input techniques for smartphones.

Asian CHI Symposium 2021 , May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8203-8/21/05…$15.00
https://doi.org/10.1145/3429360.3468171

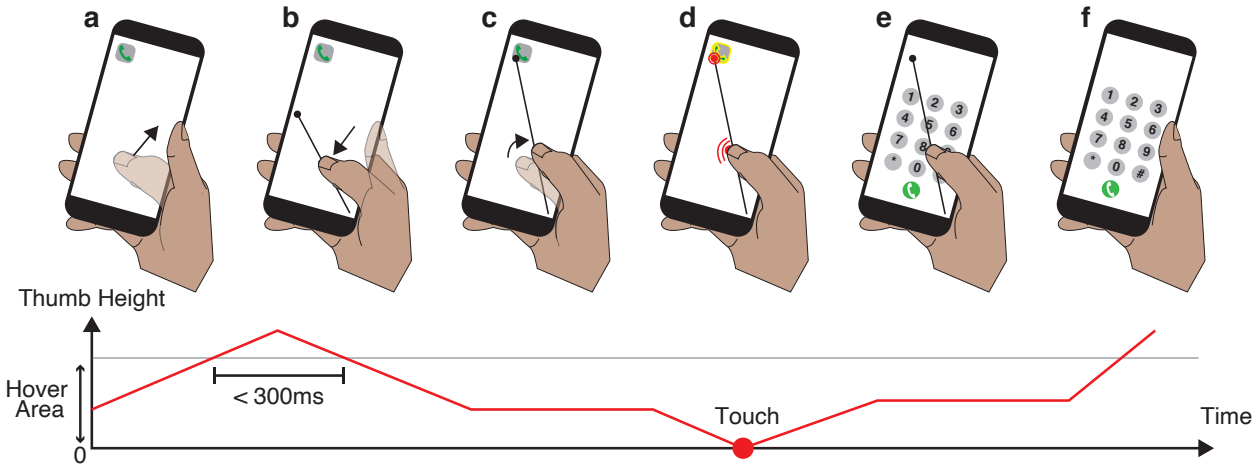

Figure 1: Procedure of our technique. In our technique, users first move their thumb out of the hover area from the hover area (a) and then move their thumb into the hover area again within 300 ms (b) to activate the cursor. Then, users move their thumb in the hover area to move the cursor to the unreachable target (c). To transfer a single-touch gesture to the cursor position, users perform a single-touch gesture as usual; in the case of a tap gesture, for example, users perform a tap on the screen, that is, touch the screen with their thumb (d) and then release their thumb from the screen. After performing a single-touch gesture, users can continuously operate with the cursor by keeping their thumb in the hover area (e). By moving their thumb out of the hover area, users can end the cursor operation and return to the normal touch operation (f).

2.1 Reachability Technique

Many reachability techniques have been developed to enable users to access unreachable areas. Chang et al. [5] classified these techniques into three categories: screen transform, proxy region, and cursor techniques. We also use these classifications to discuss the relationship between our technique and existing reachability techniques.

2.1.1 Screen Transform Technique. The screen transform technique enables users to operate on unreachable areas by moving or shrinking the entire screen such that it is near the thumb. This function is installed on some smartphones by default. On Apple iPhones, the screen can be moved down halfway when users double-tap the home button or swipe the bottom edge of the screen down [2]. In Samsung's One-Handed Mode [24], the entire screen shrinks and is moved close to the thumb when users triple-tap the home button or swipe their finger from the corner of the screen.

Researchers of human-computer interaction have also developed several screen transform techniques. TiltReduction [5] shrinks the entire screen when users tilt the smartphone. Sliding Screen [14] moves the screen in the opposite direction of the finger movements when users swipe from the bezel or in the same direction as the finger movements when users touch with the large contact area (large touch). TiltSlide [5] moves the screen in the direction in which the smartphone is tilted. MovingScreen [26] moves the screen in the opposite direction of the user's swipe from the bezel; users can adjust the screen moving speed by adjusting how far they scroll the screen edge before swiping from the bezel. Le et al. [16] developed techniques that enable users to move the entire screen by sliding their index finger on the touchpad on the back of the device.

In these techniques, part of the UI is hidden so context information is hidden, or it is difficult to select targets which have been shrunk.

2.1.2 Proxy Region Technique. The proxy region technique uses an alternative area to enable users to select an unreachable target.

ThumbSpace [13] shrinks the entire screen and displays it as a pop-up, the size and position of which can be specified by dragging. However, in this technique, the entire screen is shrunk, and the aspect ratio of the screen also changes. TapTap [23] magnifies the part of the screen around the users' tap location as a pop-up. Because users need to tap near the area they want to magnify, they cannot use this technique if the unreachable area is large. Hasan et al. [12] proposed a technique that uses in-air space between the screen and the thumb as a proxy region. While they used the hover area for shortcut gestures, we use the hover area to move the cursor. Löchtefeld et al. [19] added a touchpad to the back of the device to enable users to select a target located at the top of the screen with their index finger. This technique enables access to the unreachable area at the top of the screen; however, the bottom of the screen still cannot be reached.

2.1.3 Cursor Technique. The cursor technique enables users to select an unreachable target by using the cursor instead of directly touching the target.

TiltCursor [5] uses an accelerated cursor triggered by dragging the screen while the device is tilted. BezelCursor [18] and ExtendedThumb [15] also use an accelerated cursor; BezelCursor [18] is triggered by swiping from the bezel, and ExtendedThumb [15] is triggered by double-tapping. Extendible Cursor [14] is activated

when users perform a large touch or swipe from the bezel; when activated by large touch, the cursor moves in the opposite direction as the finger movement, and when activated by swipe, the cursor moves in the same direction. MagStick [23] is activated by touching the screen; the cursor moves in the opposite direction of the finger movement. In ForceRay [7], when users apply force to the screen, a virtual ray extends to the opposite edge of the screen in the direction of the finger and a cursor that moves along it is displayed. The harder the user presses, the farther away from the thumb the cursor moves. Voelker et al. [27] developed techniques that move the cursor using the users' head orientation. The cursors are activated by applying force to the screen, and then users move the cursor by their head orientation and dragging their finger.

In these cursor techniques, users can select unreachable targets by releasing the finger from the screen to transfer a tap gesture to the cursor position. Thus, users cannot perform single-touch gestures other than tap. With our technique, on the other hand, users can perform all single-touch gestures.

Hakka et al. [10] developed two techniques (Force Cursor and Event Forward Cursor) to enable users to perform all single-touch gestures on the unreachable areas. In Force Cursor, after moving the cursor by dragging, the touching state is assigned by increasing force to the screen, and the state where the thumb is released from the screen is assigned by decreasing the force. The accuracy of Force Cursor was reportedly low due to the difficulty of controlling the touch pressure to execute a gesture. In our technique, gestures are performed in the same way as standard touch operation. In Event Forward Cursor, after moving the cursor by dragging, users release their thumb to activate the forward mode. During the forward mode, the cursor is fixed and touch gestures performed at any position on the screen are transferred to the cursor position. The cursor disappears after the user executes the gesture during the forward mode, so if the user operates on an unreachable target, the user needs to activate and move the cursor again. In our technique, the user can continuously perform a single-touch gesture on an unreachable target by keeping the thumb in the hover area after executing the gesture.

2.2 Hover Input

Hover input has been investigated to expand the input vocabulary of smartphones.

HoverZoom Keyboard [22] is a software keyboard that enlarges keys under the hovering finger using a fisheye view to facilitate the selection of small keys. Aiyoshizawa and Komuro [1] also developed a software keyboard that enlarges the keys under the hovering finger using a different enlargement mechanism than that of HoverZoom Keyboard. Chen et al. [6] explored the combination of touch and in-air input. AirFlip [11] is an in-air gesture in which the user double-crosses the side boundary surface of the hover area, while AirFlip-Undo [25] is a technique for quickly undoing text input by AirFlip. Hover Cursor [20] displays the cursor at an offset position from the hovering finger to facilitate the selection of small targets.

In our technique, we used the hover input to enable users to perform all single-touch gestures on unreachable areas.

3 DESIGN OF OUR TECHNIQUE

Various cursor triggers have been used in cursor techniques (e.g., swipe from the bezel [10, 14, 18], large touch [14], and force touch [7, 27]). These triggers provide seamless operation for techniques where users move the cursor by touching the screen with a finger. However, these triggers could not provide seamless operation for our technique, in which the cursor is moved using a hovering finger. Moreover, the touch gestures used as cursor triggers in previous studies were assigned to various operations on smartphones. To address these issues, our technique uses an in-air gesture as a trigger for activating the cursor. The in-air gesture is a modified AirFlip [11], which utilizes double-crossing the side boundary surface of the hover area. AirFlip is the gesture in which users move their thumb from the outside of the hover area into the area and then back out. We designed a gesture which is the reverse of AirFlip such that the user double-crosses the top boundary surface of the hover area. To activate the cursor with our technique, users move their thumb from inside to outside the hover area and then back into the hover area within 300 ms (Fig. 2a). This threshold is defined such that the cursor will not be activated unintentionally.

The displayed position of the cursor is absolutely determined by the position of the thumb; the cursor is displayed on the extension of the base position and hover or touch position (Fig. 2b). The base position is located in the lower right corner of the screen for right-handed users and in the lower left corner for left-handed users. This position was designed to be the position of the thumb's carpometacarpal, which is the joint used to rotate the thumb. The cursor is displayed at the position where the vector from the base position to the hover or touch position is multiplied by α (extension rate).

In our technique, when the cursor is activated, all touch events that occur on the screen are transferred to the cursor position. Therefore, to transfer a single-touch gesture to the cursor position, after moving the cursor to the desired position, users perform the gesture on the screen as usual. For example, in the case of a tap, users touch the screen and then immediately lift their finger away.

4 PILOT STUDY

We conducted a pilot study with the first author (23 years old, male, right-handed) to investigate the preliminary performance of our technique using a Samsung Galaxy Note 3 ($151.2 \times 79.2 \times 8.3$ mm, 5.7 inches) smartphone, which has a screen that can detect a hovering finger up to approximately 20 mm above the screen. We compared our technique (Ours) with Direct Touch input (DT, i.e., without a reachability technique) as a baseline and two existing techniques, One-Handed Mode (OM) and Event Forward Cursor (EC) [10]. We chose these two techniques because OM is the fastest technique among those that can execute all single-touch gestures, and EC can execute all single-touch gestures accurately regardless of gesture, according to Hakka et al. [10].

4.1 Technique

One-Handed Mode (OM), which mimics Samsung's reachability technique [24], shrinks the entire screen to $\frac{2}{3}$ and moves it to the lower right corner of the screen. Because Android SDK cannot detect when the home button is double-tapped, OM is activated

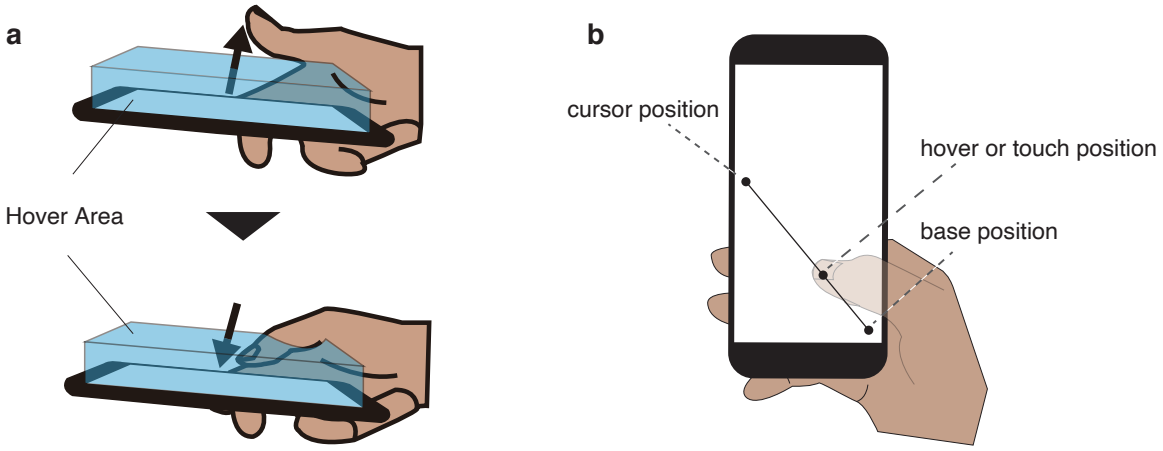

Figure 2: Cursor trigger (a) and cursor position (b) in our technique. To activate the cursor, users move their thumb from the inside to the outside of the hover area, and then move the thumb into the hover area (a). The cursor is on a straight line passing through the base position and the hover or touch position, and the position is calculated by multiplying the distance from the base position to the hover or touch position by α (extension rate) (b).

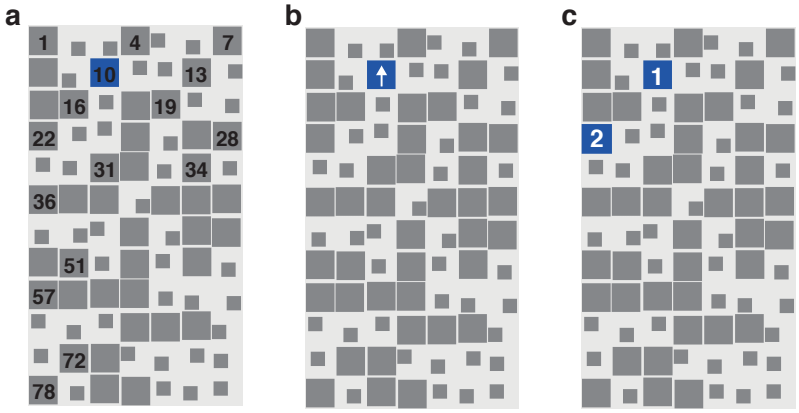

Figure 3: Screen configuration in the pilot study. (a) tap and double-tap task, (b) swipe task, and (c) drag task. 16 targets (numbered in (a) but not in the actual trial) and distractors are placed in an invisible 7×12 grid.

by double-tapping the back button next to the home button. In addition, users can return the screen to its original size by double-tapping the back button while the screen is shrunk. In the pilot study, the participant was able to continuously perform gestures on the target after the screen was shrunk.

Event Forward Cursor (EC) is the cursor technique activated by swiping from the bezel. After the cursor is activated, its movement is three times as long as that of the dragging finger. Users move the cursor to the target by dragging their thumb and then releasing it to activate the forward mode. During the forward mode, the cursor position is fixed and the touch gesture performed by the user at any position on the screen is transferred to the cursor position. When users finish transferring any one of the touch gestures, the forward mode ends and the cursor disappears from the screen.

For the implementation of our technique, since the hover area that the device we used can detect hovering finger is a little inside the screen, we shifted the base position 6.6 mm inward vertically and horizontally from the lower right corner of the screen, so that the device can detect the hovering finger successfully when the participant moves the thumb to the side or bottom edge of the screen. We used the x and y coordinates of the hovering finger acquired through the Android API as the hover position. In the

pilot study, we used an extension rate of 3.0. In addition, to reduce cursor jitter, the hover location obtained from the device was filtered using the 1€ Filter [4].

4.2 Task and Target

We placed 16 targets in an invisible 7×12 grid and added distractors to other cells where targets were not placed (Fig. 3a). Targets were centered in the cells and distractors were placed at random positions in the cell to avoid a regular-looking arrangement as [13]. There are two target sizes, small (4.8 mm × 4.8 mm) and large (9.6 mm × 9.6 mm), based on the experiment of [7]. These represent the size of a button and an application icon of typical iOS widgets, respectively.

There are four tasks: tap task (Tap), double-tap task (DTap), swipe task (Swipe), and drag task (Drag). In Tap and DTap, the participant performs a tap or a double-tap on the target shown in blue (Fig. 3a). In Swipe, a white arrow is displayed on the target shown in blue (Fig. 3b), and the participant performs swipe on the target in direction of the arrow; the direction is randomly chosen from one of four directions (up, down, right, and left). In Drag, the two targets are shown in blue and labeled "1" (T1) and "2" (T2) (Fig. 3c), and the participant drags T1 to T2. T2 is randomly selected from targets other than T1. In each task, if the correct gesture is performed on the target, the trial is recognized as a success; if a different gesture is performed, or if a gesture is performed anywhere except for on the highlighted target, the trial is recognized as a failure. If successful, the next randomly selected target will be displayed, and in case of a failure, the same target will be displayed again.

We recorded 4 techniques × 4 tasks × 16 targets × 2 sizes × 2 repetitions = 1,024 trials.

4.3 Result

We show the results of *Time* and *Success* by *TECHNIQUE* × *TASK* on large (Fig. 4) and small (Fig. 5) targets.

4.3.1 Time. We conducted a three-way repeated-measures ANOVA on Time; the independent variables are techniques, tasks, and target sizes. Technique had a significant main effect on *Time* ($F_{3,992} = 66.440$, $p < .01$). Tukey HSD post hoc pairwise comparisons were all significant (p <.01) except between DT and OM. For large targets, OM was the fastest, followed by DT, Ours, and EC for all *TASK* (Fig. 4 left). On the other hand, for small targets, while DT was the fastest for DTap and Drag followed by OM, for Tap and Swipe, OM was the fastest followed by DT (Fig. 5 left). The result that techniques of directly touching the target are faster than cursor techniques regardless of the performed gesture is the same as the result shown by Hakka et al. [10]. In Ours and EC, which are cursor techniques, Ours was fast except for DTap when the target was small. Therefore, Ours may perform gestures quickly as a cursor technique that can perform single-touch gestures.

4.3.2 Success. For large targets, all techniques had a success rate of 90% or higher regardless of the performed gesture except for DTap of OM (Fig. 4 right). On the other hand, for small targets, many *TECHNIQUE* × *TASK* had low success rates (Fig. 5 right). For example, the success rate for Tap of OM was 76%, for DTap of OM and DTap of Ours was 74% and 68% respectively, and for Drag of OM was 64%. In Ours, while the success rate of Tap was the highest, the success rate of DTap and Swipe was the lowest.

4.3.3 Summary. The results show that our technique could be used to perform single-touch gestures faster than EC, which enables users to perform all single-touch gestures using the cursor. However, for gestures other than Tap, our technique had a lower success rate than that of other techniques. The success rate was particularly low (68%) in the case of DTap for small targets. For Tap, our technique was faster and more accurate than EC. Nevertheless, it is necessary to improve the performance of gestures other than Tap.

5 DISCUSSION

In the pilot study, the performance of DTap using our technique was low, especially for small targets. For double-tap, the target tapped on the first and the second time must be the same. However, in our technique, the amount of cursor movement is larger than the amount of finger movement. As a result, in the pilot study, after tapping the target the first time, the cursor moved unintentionally, and then the second tap was outside of the target, resulting in frequent failures. This problem may be solved by reducing the amount of cursor movement for a certain period of time after the first tap. For example, in the Android default setting, a double-tap is recognized when users perform a second tap within 300 ms after the first tap. Therefore, it may be possible to improve the accuracy of a double-tap by reducing the amount of cursor movement during the 300 ms after the first tap is executed.

In our technique, many failures occurred in the upper right area of the screen; out of the 33 failures, 25 (75.8%) were in the upper right area of the screen (targets labeled 7, 13, 19, 28, and 34 in Fig. 3a). When the thumb moves to the right, the thumb rotates to the right, so the user has to touch the screen with the side of the thumb. This may have been the cause of the drop in accuracy.

6 LIMITATION AND FUTURE WORK

The pilot study only had one participant, one of the authors who was familiar with the developed technique. Therefore, it is unclear whether the same results would be obtained from other users. Additional user studies need to be conducted with more varied participants (e.g., wider range of age, experience with hover operation).

Our technique is currently limited to devices with a hover-sensing touchscreen, so our technique cannot be implemented on many devices. However, techniques have been developed which track the fingers moving around the screen with a mirror [28] or fisheye lens [21] attached to the front-facing camera of the device. By incorporating these techniques which detect hovering of the finger using a camera, we can potentially expand our technique to more devices. In future work, we will try to implement our technique using a camera.

In the current implementation of our technique, users need to specify which hand they are using to operate the smartphone beforehand. However, the users may change their operating hand depending on the situation. Therefore, to eliminate the trouble of specifying which hand to use, we aim to implement device rotation [9] to recognize the user's grip automatically.

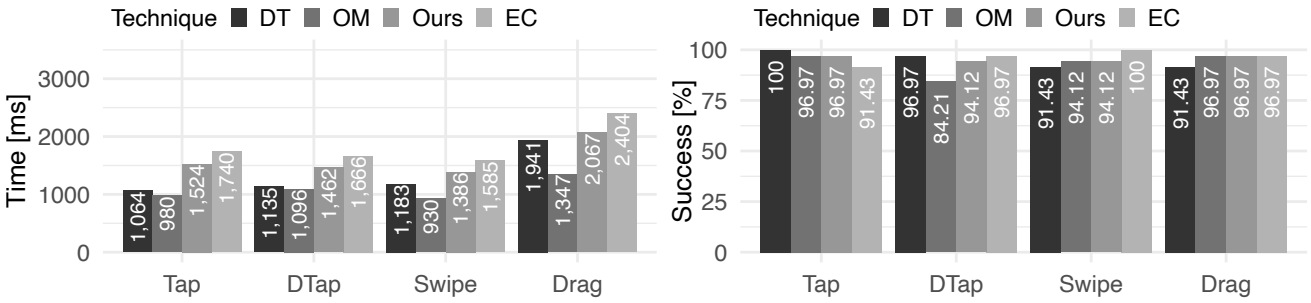

Figure 4: *Time* **(left) and** *Success* **(right) by** *TECHNIQUE* × *TASK* **on large targets.**

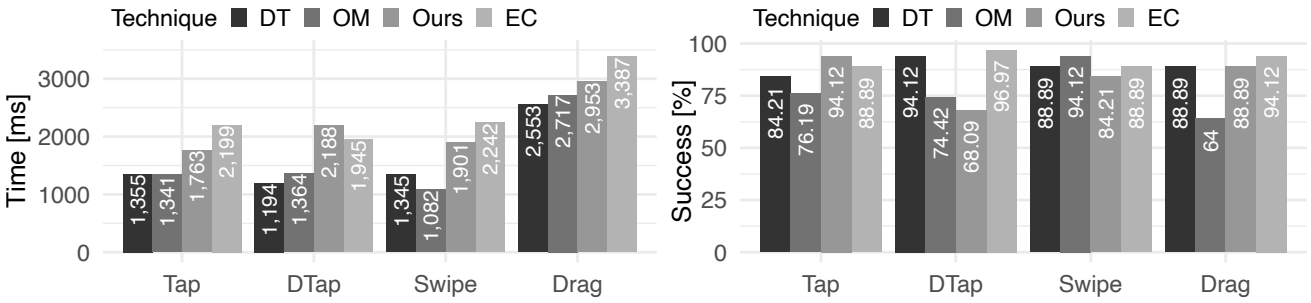

Figure 5: *Time* **(left) and** *Success* **(right) by** *TECHNIQUE* × *TASK* **on small targets.**

7 CONCLUSION

We designed a hover-based reachability technique that uses a cursor to enable users to perform all single-touch gestures on unreachable areas of a smartphone screen. Subsequently, we conducted a pilot study to investigate the user performance with our technique and compared it with existing techniques, One-Handed Mode (OM) and Event Forward Cursor (EC). The results showed that our technique was faster than EC, which uses the cursor technique same as ours, except for double-tap when the target was small. However, our technique was slower than OM, which is the technique that shrinks the entire screen. In future work, we will improve the proposed technique on the basis of the results of the pilot study and conduct additional user studies with a wider range of participants.

REFERENCES

[1] Toshiaki Aiyoshizawa and Takashi Komuro. 2017. Comparative Study on Text Entry Methods for Mobile Devices with a Hover Function. In *Proceedings of the 16th International Conference on Mobile and Ubiquitous Multimedia* (Stuttgart, Germany) *(MUM '17)*. Association for Computing Machinery, New York, NY, USA, 355–361. https://doi.org/10.1145/3152832.3156614

[2] Apple. 2020. Adjust touch settings on iPhone - Apple Support. https://support.apple.com/guide/iphone/touch-iph77bcdd132/14.0/ios/14.0. [Online; accessed January 20, 2021].

[3] Joanna Bergstrom-Lehtovirta and Antti Oulasvirta. 2014. Modeling the Functional Area of the Thumb on Mobile Touchscreen Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 1991–2000. https://doi.org/10.1145/2556288.2557354

[4] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1 € Filter: A Simple Speed-Based Low-Pass Filter for Noisy Input in Interactive Systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI '12)*. Association for Computing Machinery, New York, NY, USA, 2527–2530. https://doi.org/10.1145/2207676.2208639

[5] Youli Chang, Sehi L'Yi, Kyle Koh, and Jinwook Seo. 2015. 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* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1499–1508. https://doi.org/10.1145/2702123.2702425

[6] Xiang 'Anthony' Chen, Julia Schwarz, Chris Harrison, Jennifer Mankoff, and Scott E. Hudson. 2014. Air+Touch: Interweaving Touch & in-Air Gestures. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 519–525. https://doi.org/10.1145/2642918.2647392

[7] Christian Corsten, Marcel Lahaye, Jan Borchers, and Simon Voelker. 2019. ForceRay: Extending Thumb Reach via Force Input Stabilizes Device Grip for Mobile Touch Input. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3290605.3300442

[8] Rachel Eardley, Anne Roudaut, Steve Gill, and Stephen J. Thompson. 2017. Understanding Grip Shifts: How Form Factors Impact Hand Movements on Mobile Phones. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 4680–4691. https://doi.org/10.1145/3025453.3025835

[9] Mayank Goel, Jacob Wobbrock, and Shwetak Patel. 2012. 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* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 545–554. https://doi.org/10.1145/2380116.2380184

[10] Kyohei Hakka, Toshiya Isomoto, and Buntarou Shizuki. 2020. Investigation of One-Handed Interaction Techniques for Single-Touch Gesture on Large Smartphones. In *Proceedings of the 25th IPSJ Symposium on Interaction* (Tokyo, Japan) *(INTERACTION 2020)*. Information Processing Society of Japan, 48–57. http://www.interaction-ipsj.org/proceedings/2020/data/bib/INT20006.html. (In Japanese).

[11] Hiroyuki Hakoda, Takuro Kuribara, Keigo Shima, Buntarou Shizuki, and Jiro Tanaka. 2015. AirFlip: A Double Crossing In-Air Gesture Using Boundary Surfaces

of Hover Zone for Mobile Devices. In *Proceedings of the 17th international conference on Human-Computer Interaction* (Los Angeles, CA, USA) *(HCII '15)*. Springer International Publishing, Cham, Switzerland, 44–53. https://doi.org/10.1007/978-3-319-20916-6_5

[12] Khalad Hasan, Junhyeok Kim, David Ahlström, and Pourang Irani. 2016. Thumbs-Up: 3D Spatial Thumb-Reachable Space for One-Handed Thumb Interaction on Smartphones. In *Proceedings of the 2016 Symposium on Spatial User Interaction* (Tokyo, Japan) *(SUI '16)*. Association for Computing Machinery, New York, NY, USA, 103–106. https://doi.org/10.1145/2983310.2985755

[13] Amy K. Karlson and Benjamin B. Bederson. 2007. ThumbSpace: Generalized One-Handed Input for Touchscreen-Based Mobile Devices. In *Proceedings of the 11th IFIP TC 13 International Conference on Human-Computer Interaction* (Rio de Janeiro, Brazil) *(INTERACT '07)*. Springer-Verlag, Berlin, Heidelberg, 324–338. https://doi.org/10.1007/978-3-540-74796-3_30

[14] Sunjun Kim, Jihyun Yu, and Geehyuk Lee. 2012. Interaction Techniques for Unreachable Objects on the Touchscreen. In *Proceedings of the 24th Australian Computer-Human Interaction Conference* (Melbourne, Australia) *(OzCHI '12)*. Association for Computing Machinery, New York, NY, USA, 295–298. https://doi.org/10.1145/2414536.2414585

[15] Jianwei Lai and Dongsong Zhang. 2015. ExtendedThumb: A Target Acquisition Approach for One-Handed Interaction With Touch-Screen Mobile Phones. *IEEE Transactions on Human-Machine Systems* 45, 3 (June 2015), 362–370. https://doi.org/10.1109/THMS.2014.2377205

[16] Huy Viet Le, Patrick Bader, Thomas Kosch, and Niels Henze. 2016. Investigating Screen Shifting Techniques to Improve One-Handed Smartphone Usage. In *Proceedings of the 9th Nordic Conference on Human-Computer Interaction* (Gothenburg, Sweden) *(NordiCHI '16)*. Association for Computing Machinery, New York, NY, USA, Article 27, 10 pages. https://doi.org/10.1145/2971485.2971562

[17] Huy Viet Le, Sven Mayer, Patrick Bader, and Niels Henze. 2018. Fingers' Range and Comfortable Area for One-Handed Smartphone Interaction Beyond the Touchscreen. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3173574.3173605

[18] Wing Ho Andy Li, Hongbo Fu, and Kening Zhu. 2016. BezelCursor: Bezel-Initiated Cursor for One-Handed Target Acquisition on Mobile Touch Screens. *International Journal of Mobile Human Computer Interaction (IJMHCI)* 8, 1 (Jan. 2016), 1–22. https://doi.org/10.4018/IJMHCI.2016010101

[19] Markus Löchtefeld, Christoph Hirtz, and Sven Gehring. 2013. Evaluation of Hybrid Front- and Back-of-Device Interaction on Mobile Devices. In *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia* (Luleå, Sweden) *(MUM '13)*. Association for Computing Machinery, New York, NY, USA, Article 17, 4 pages. https://doi.org/10.1145/2541831.2541865

[20] Anna Ostberg and Nada Matic. 2015. Hover Cursor: Improving Touchscreen Acquisition of Small Targets With Hover-Enabled Pre-Selection. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI EA '15)*. Association for Computing Machinery, New York, NY, USA, 1723–1728. https://doi.org/10.1145/2702613.2732903

[21] Keunwoo Park, Sunbum Kim, Youngwoo Yoon, Tae-Kyun Kim, and Geehyuk Lee. 2020. DeepFisheye: Near-Surface Multi-Finger Tracking Technology Using Fisheye Camera. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '20)*. Association for Computing Machinery, New York, NY, USA, 1132–1146. https://doi.org/10.1145/3379337.3415818

[22] Frederic Pollmann, Dirk Wenig, and Rainer Malaka. 2014. HoverZoom: Making on-Screen Keyboards More Accessible. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI EA '14)*. Association for Computing Machinery, New York, NY, USA, 1261–1266. https://doi.org/10.1145/2559206.2581173

[23] Anne Roudaut, Stéphane Huot, and Eric Lecolinet. 2008. TapTap and MagStick: Improving One-Handed Target Acquisition on Small Touch-Screens. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (Napoli, Italy) *(AVI '08)*. Association for Computing Machinery, New York, NY, USA, 146–153. https://doi.org/10.1145/1385569.1385594

[24] Samsung. 2019. Galaxy Smartphone - One handed mode | Samsung Australia. https://www.samsung.com/au/getstarted/advanced/one-handed-mode/. [Online; accessed January 19, 2021].

[25] Keigo Shima, Ryosuke Takada, Kazusa Onishi, Takuya Adachi, Buntarou Shizuki, and Jiro Tanaka. 2015. AirFlip-Undo: Quick Undo Using a Double Crossing In-Air Gesture in Hover Zone. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software and Technology* (Daegu, Kyungpook, Republic of Korea) *(UIST '15 Adjunct)*. Association for Computing Machinery, New York, NY, USA, 97–98. https://doi.org/10.1145/2815585.2815737

[26] Hsin-Ruey Tsai, Da-Yuan Huang, Chen-Hsin Hsieh, Lee-Ting Huang, and Yi-Ping Hung. 2016. MovingScreen: Selecting Hard-to-Reach Targets with Automatic Comfort Zone Calibration on Mobile Devices. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct* (Florence, Italy) *(MobileHCI '16)*. Association for Computing Machinery, New York, NY, USA, 651–658. https://doi.org/10.1145/2957265.2961835

[27] Simon Voelker, Sebastian Hueber, Christian Corsten, and Christian Remy. 2020. HeadReach: Using Head Tracking to Increase Reachability on Mobile Touch Devices. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376868

[28] Chun Yu, Xiaoying Wei, Shubh Vachher, Yue Qin, Chen Liang, Yueting Weng, Yizheng Gu, and Yuanchun Shi. 2019. HandSee: Enabling Full Hand Interaction on Smartphone with Front Camera-Based Stereo Vision. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300935