International Journal of Advanced Smart Convergence Vol.10 No.3 143-148(2021)
http://dx.doi.org/10.7236/IJASC.2021.10.3.143

IJASC 21-3-18

OverIT: An Interactive Overlay for Touchscreen-based UI Customization by Demonstration

Kyungyeon Lee[1], SeungA Chung[2], Uran Oh[3*]

[1]*Bachelor, Department of Computer Science and Engineering Ewha Womans University, Seoul, South Korea*
[2]*Master Student, Department of Computer Science and Engineering Ewha Womans University, Seoul, South Korea*
[3]*Assistant Professor, Department of Computer Science and Engineering Ewha Womans University, Seoul, South Korea*
Email: ruddus716@ewhain.net, ewhacsa@ewhain.net, uran.oh@ewha.ac.kr

Abstract

Smartphones have been widely used for various purposes and stay connected with people at all times. However, the use of such touchscreen devices can be physically restricted depending on users' context where only one hand is available to interact with the device. Even major smartphone manufacturers (e.g., Apple, Samsung) offer one handed mode, they still lack functions in the third-party applications, and the process is also complicated. We propose OverIT, a system that enables users to customize interfaces by adding new buttons on an interactive overlay which can be positioned anywhere on the touchscreen where each button serves the same functionality as an existing one. It is designed to support users to map a certain button event freely and easily to a newly created button by performing a demonstration of a button tap. We expect our system to improve the overall user experience of one-handed interaction with touchscreen devices.

Keywords: *One-handed mobile interaction, UI customization; programming-by-demonstration; touchscreens*

1. Introduction

Many people prefer using one hand to both hands when interacting with their mobile phones [1]. However, one-handed use of mobile phones is not always possible especially for people with small hands and/or with large devices, or people whose one hand is occupied for other tasks (*e.g.*, holding an umbrella). As such, major smartphone manufacturers such as Samsung [2] and Apple [3] offer one-handed mode. However, the mode setting is effectless for third party applications (apps) and the number of available functions is very limited. Jailbreaking phones is another option to customize their installed mobile apps, the process is complicated (*e.g.*, rooting) for most users.

Inspired by prior application-independent software-based interaction techniques for supporting one-handed uses of mobile phones [4-7] we propose OverIT that allows users to customize their user interface (UI) of an app by adding duplicates of the original buttons to any desired locations on the touchscreen as shown in Figure

Manuscript Received: August. 2, 2021 / Revised: August. 8, 2021 / Accepted: August. 10, 2021
*Corresponding Author: uran.oh@ewha.ac.kr
Tel: +82-2-3277-6896
Assistant Professor, Department of Computer Science and Engineering Ewha Womans University, Seoul, South Korea

1. An example use case scenario of OverIT is as follows:

Alice would like to take a selfie with her smartphone using a camera app. However, she cannot reach certain buttons such as the "LockExposure" button at the top left corner with her thumb. Thus, she wishes to adjust the location of the certain button to closer to the bottom-right corner of the screen to be able to tap the button with her thumb while holding the phone with her right hand.

As programmatically making this type of UI customization for end-users is difficult, prior studies have shown its effectiveness of adopting the programming by demonstration (PBD) approach for end-users on mobile and IoT platforms [8, 9]. SUGILITE [8], for instance, allows users to develop smartphone automation scripts by demonstrating a series of repeated user input performed on the interface of a third-party app. Their tool records the interaction as a script and plays it so that the set of UI input gets performed in the same order automatically as recorded. Similarly, the customization process of OverIT was supported via programming-by-demonstration where the type of demonstration is invoking a UI event (i.e., a button tap). The contribution of this work is the proposal of a UI customization system for improving the user experience of one-hand mobile interactions, which we have open-sourced for public use.

Figure 1. The "LockExposure" button of a camera app is not reachable while holding the phone with one hand (left). However, with OverIT, users can create a duplicated button which can be placed at any desired location (right).

2. The System: OverIT

While OverIT can be embedded in any smartphone apps, we used Open Camera [10], an open-source app whose number of downloads exceeds 10,000,000 on Google Play, to demonstrate our work.

2.1 The System Components

As shown in Figure 2, we added a touchscreen overlay into this app, and installed a PBD module into the internal database to record the mappings between functionalities and UI components (*i.e.*, buttons) of the app. As for the implementation, we used Java and XML and OverIT was tested on Galaxy S10 and Galaxy Note 10

running Android 10.0. Note that the system does not need root access and can be run on any android smartphones running Android 4.19 (API Level 16) or above. The code is open-sourced on GitHub [11]. The implementation details per system components are below.

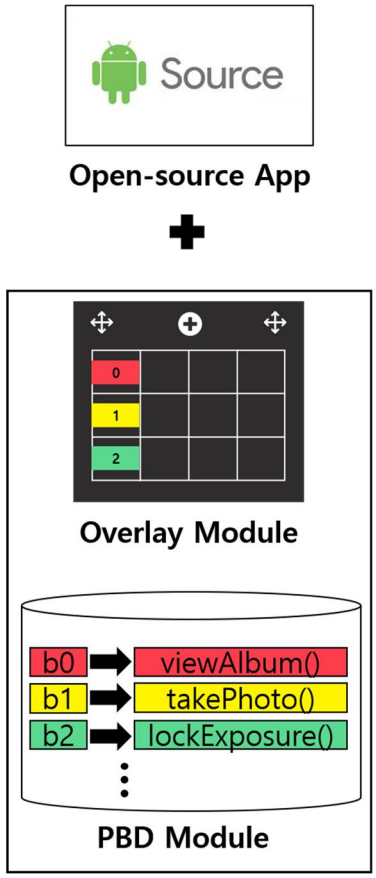


Figure 2. The system overview. OverIT adds a touchscreen overlay on the UI of an open-source app while maintaining a mapping between newly created buttons and their corresponding functions.

Overlay module. OverIT can easily be embedded into open-sourced Android or custom apps during the development by adding an interactive overlay interface to the target app (Figure 2). That is, the overlay is application-independent so that it is designed not to interfere with the original app's interface; it is transparent and its location on the screen can be easily adjusted by a drag-and-drop gesture. While we had 3-by-4 grid overlay for the demonstration of OverIT, the width and height of the layout can be set differently when embedding this module to the original app.

PBD module. The PBD module is designed to support UI customization by demonstrations. With this module, OverIT allows users to easily duplicate a button as well as its functionality by tapping an existing button on the original app. By leveraging the internal database [12], we built a key-value store where the key is the defined UI button, and value is the method that gets invoked when the corresponding button is tapped such as "b0"-"viewAlbum()" and "b1"-"takePhoto()" as shown in Figure 2. Once a key-value pair is recorded via demonstration, users can now tab the new button that they created on the overlay to experience the same effect

as the original one which is difficult to reach for one-handed use.

2.2 The Customization Process

To customize the UI of an app using OverIT, a user can follow 4 steps as shown in Figure 3 (watch the video figure as well for the reference):

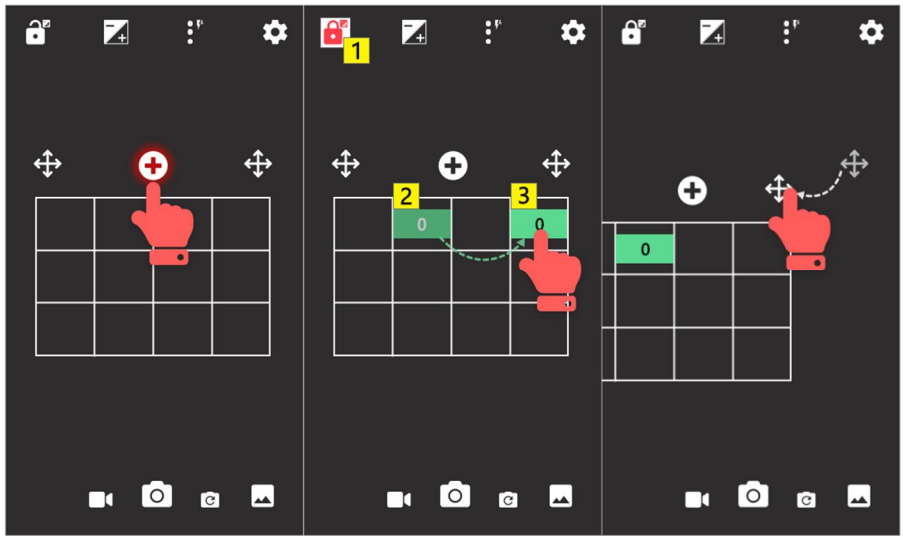
(1) *Initiation*: create a new button by tapping the plus button at the top center of the overlay (Figure 3a).
(2) *Demonstration*: map the functionality of an existing button to the newly created button by tapping the original button (Figure $3b_1$).
(3) *Arrangement*: re-locate the new button from the initial location (Figure $3b_2$) to any cell on the overlay (Figure $3b_3$) by dragging if needed.
(4) *Re-position*: position the overlay to desired location on the screen by dragging either the top left or the top right corner anchor buttons (Figure 3c).

Initiation. As shown in Figure 3a, the overlay has a plus button that creates a button on this overlay. However, No function is mapped to this newly created button at this point yet. Note that the number of buttons that users can create depends on the layout up the overlay. For example, with the current 3-by-4 layout of the overlay, users can create up to 12 new buttons in total.

Demonstration. To duplicate the functionality of a button in the original app and map it to the newly created button, a user has to demonstrate a button tap. For instance, suppose a user wishes to create a duplicate for "LockExposure" button in the upper left corner as shown in Figure $3b_1$. Then she can perform a button tab on the "LockExposure" button right after creating a new button on the grid (Figure $3b_2$). As soon as the demonstration is completed, the invoked method and the new button information get recorded immediately to the PBD module so that she can start tapping the duplicated button that functions exactly the same as the original button. If a user wishes to change the mapped functionality of a button, she can jut.

Arrangement. Once the new button and one of the functionalities of existing buttons have been mapped after the demonstration, a user can now change the location of the new button to a different location as shown in Figure 3b. To do so, a user should first press-and-hold the target button (Figure $3b_2$) that she wishes to arrange, drag it to one of the cell locations on the grid (Figure $3b_3$), and then release the button. If the button has been released at a location that is ambiguous or out of the grid layout, the arrangement will be canceled and the button will be returned to its original cell position.

Re-Position. The overlay of OverIT has two draggable buttons at the top of each side, which allows users to change the location of the overlay to a different location on the screen by performing drag-and-drop as shown in Figure 3c. Since the layer of the overlay and the original app is independent, users can still be able to interact with the existing UI components of the original app even if the overlay is on top of the components unless they are occluded with the newly created buttons on the grid.



(a) (b) (c)

Figure 3. The interface of OverIT for (a) adding a new button, (b) demonstrating a user input by tapping the target button for making a duplicate and arranging button locations, and (c) re-positioning the overlay.

4. Future Work and Conclusion

In this study, we presented OverIT, a system that allows users to customize the UI of an Android app by performing a simple demonstration. By implementing an interactive overlay, users could add and locate buttons that are mapped with the function of the original app only following the 4 steps (*Initiation, Demonstration, Arrangement, and Re-Position*). In the future, we are working on supporting more sophisticated manipulations in OverIT. For instance, we will enable users to edit buttons and their mapped functions as well as the size of the overlay. In addition, while the current version can only duplicate a tap gesture for buttons, we will allow a variety of interactions including dragging and long-click for various UI components such as sliders and drop-down buttons. Eventually, we plan to extend this work by deploying it as an app which can be used as a plug-in for any third-party applications and assess real-world usages and its effectiveness of our system.

Acknowledgement

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2021R1F1A1052786).

References

[1] Amy K Karlson, Benjamin B Bederson, and J Contreras-Vidal. 2006. Understanding single-handed mobile device interaction. Handbook of research on user interface design and evaluation for mobile technology 1 (2006), 86–101. DOI: https://doi.org/10.1007/978-3-540-74796-3_30

[2] Using One Handed Mode on my Samsung Phone, https://www.samsung.com/au/support/mobile-devices/using-one-handed-mode/

[3] iMore, https://www.imore.com/iphone-11-secret-gesture-button-shortcuts

[4] Karlson, A. K., & Bederson, B. B. (2008, April). One-handed touchscreen input for legacy applications. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 1399-1408). DOI: https://doi.org/10.1145/1357054.1357274

[5] Le, H. V., Bader, P., Kosch, T., & Henze, N. (2016, October). Investigating screen shifting techniques to improve one-handed smartphone usage. In Proceedings of the 9th Nordic Conference on Human-Computer Interaction (pp. 1-10). DOI: https://doi.org/10.1145/2971485.2971562

[6] Le, H. V., Mayer, S., Bader, P., & Henze, N. (2018, April). 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 (pp. 1-12). DOI: https://doi.org/10.1145/3173574.3173605

[7] Lee, S., Kyung, G., Lee, J., Moon, S. K., & Park, K. J. (2016). Grasp and index finger reach zone during one-handed smartphone rear interaction: effects of task type, phone width and hand length. Ergonomics, 59(11), 1462-1472. DOI: https://doi.org/10.1080/00140139.2016.1146346

[8] Toby Jia-Jun Li, Amos Azaria, and Brad A Myers.2017a. SUGILITE: creating multimodal smart phone automation by demonstration. In Proceedings of the 2017 CHI conference on human factors in computing systems. 6038–6049. DOI: https://doi.org/10.1145/3025453.3025483

[9] Toby Jia-Jun Li, Yuanchun Li, Fanglin Chen, and Brad A Myers. 2017b. Programming IoT devices by demonstration using mobile apps. In International Symposium on End User Development. Springer,3–17. DOI: https://doi.org/10.1007/978-3-319-58735-6_1

[10] OpenCamera, https://sourceforge.net/projects/opencamera/

[11] Kyungyeon Lee, https://github.com/kyungyeon-lee/OverIT

[12] Android Developers, https://developer.android.com/training/data-storage/shared-preferences