# Floating Hierarchical Menus for Swipe-Based Navigation on Touchscreen Mobile Devices

Alen Salkanovic, Ivan Štajduhar, and Sandi Ljubic(✉)

University of Rijeka, Faculty of Engineering, Vukovarska 58, 51000 Rijeka, Croatia
{alen.salkanovic,ivan.stajduhar,sandi.ljubic}@riteh.hr

**Abstract.** In this paper, we present two menu implementations that allow swipe-based navigation through deep hierarchical menu configurations. Instead of utilizing repetitive tap-based selections, the proposed interaction relies on continuous finger movement across different submenus. The menus are implemented as a service; hence they can easily be attached to the target mobile application and visualized as a semi-transparent floating widget on top of it. Similar to the marking menu concept, the provided designs also enable a smooth transition from novice to expert user, as swipe gestures used for menu item selections can be memorized and subsequently executed faster. Both menus initially act as a floating action button, allowing the user to change its location by dragging it to the preferred place on the screen. Visualization of the menu starts in this pivotal position, according to the utilized design: *Tile* menu or *Pie* menu. The *Tile* menu uses a linear scheme and dynamically occupies more screen real-estate when a submenu is triggered. On the other hand, the *Pie* menu is displayed as a circular widget without extra containers and uses touch-dwelling for submenu invocation. Implementations of the proposed menu designs are evaluated and comparatively analyzed by conducting a controlled experiment involving 30 participants. We present the results of this empirical research, specifically focusing on menu navigation efficiency in two different contexts of use, the related interaction workload, and usability attributes.

**Keywords:** Hierarchical menus · Swipe-based navigation · Touchscreen gestures

## 1 Introduction

Hierarchical menus are widely used in software applications written for contemporary desktop operating systems. Different menu structures are commonly visualized as linear widgets (vertically and horizontally expandable menus), which can utilize different designs. Namely, the most popular types of multi-level menus are dropdown menus, flyout menus, dropline menus, accordion menus, and split menus. However, the corresponding implementations are usually optimized for mouse and/or keyboard interaction, hence applying the same concepts in the touchscreen mobile domain can sometimes be quite questionable.

It is an undeniable fact that present-day mobile devices are powerful enough to run relatively complex software applications. Mobile versions of the popular desktop

M. Kurosu (Ed.): HCII 2020, LNCS 12182, pp. 509–522, 2020.
https://doi.org/10.1007/978-3-030-49062-1_34

applications, such as Microsoft Word or Photoshop, are already used by numerous smartphone and tablet users. However, as mobile touchscreens are substantially smaller when compared to desktop screens, implementing multi-level menus for mobile applications remains a challenge in the HCI field.

Menus in contemporary mobile applications come in different shapes and forms and follow a trade-off between screen real-estate occupation and navigation efficiency. Touchscreen mobile devices raise particular problems in menu interaction, such as delay-based context menu activation, lack of shortcuts, occlusion, and insufficient accuracy [1]. The small screen size cannot easily accommodate many subcategories; hence the most common designs include accordions (a hamburger menu), sequential menus (e.g. navigation drawer), section menus (tabs), and floating action buttons (FABs) [2]. While all mentioned solutions have pros and cons, they are all tied to the underlying application and predefined activation points, usually located on the top/bottom of the screen. Typically, menu items from a single submenu are only presented on the screen, leaving the application content predominantly hidden.

In this paper, we investigate alternative menu designs that combine some of the previously introduced concepts (e.g. radial-shaped menus, marking menus) and touchscreen interaction modalities (e.g. swipe gestures instead of tapping sequence). Our design considerations also involve menu semi-transparent visualization and customization of the menu position on the screen. Navigation efficiency of the proposed menu solutions is empirically analyzed in two contexts of use: single-handed and cradling (the case wherein one hand is holding the device, while the other one is performing swipe gestures).

## 2 Related Work

Pie menu, also known as a radial menu, is a type of menu where item selection depends on the movement direction of the pointing device. The menu options are placed along the circumference of a circle at equal radial distances from the center. The original concept is attributed to a system called PIXIE [3]. A number of empirical comparisons between pie and linear menus, applied to desktop systems, has been made. For example, one of the early studies claimed pie menus to be about 15% faster than linear menus, with a significantly lower error rate [4].

Marking menu can be described as a specific type of pie menu wherein items are selected using a straight stroke gesture. In general, marking menu concept enables an easy transition from novice to expert user, according to the two different modes it imposes [5]. In the novice mode, the menu is visible to the users while making the selections, and it disappears from the screen once a menu item is selected. However, in the expert mode, the menu is completely hidden from the user. Therefore, previously memorized stroke gestures can be made without the need for the menu to appear. Besides the possibility of learning gestures for commonly used menu options, marking menu thus supports better screen utilization from the application content standpoint.

Both the pie menu and the marking menu concept were initially proposed for the desktop interaction; nevertheless, they have been successfully applied and investigated in touchscreen mobile domain as well.

*Wavelet Menu* [6] supports exploration and navigation in multimedia data hierarchies (e.g. photos and music) on mobile devices. The menu uses previsualization, a special feature which allows user to see the submenu corresponding to a certain menu option before it is actually activated. Additionally, this radial menu incorporates two different layout solutions. In a situation when a list of menu options is very long, the circular layout is combined with the linear one. This way it is easier to display and interact with submenus containing a large number of different options.

The *Swiss Army Menu* (SAM) [7] is another solution which utilizes radial design. Its main advantage is hierarchy navigation based on small thumb movements. A pointer controlled by the finger is used to select certain menu item in order to avoid the occlusion problem. Four different types of menu items are allowed in this design, which correspond to typical usage of buttons, checkboxes, sliders, and scrollbars. Similar to the *Wavelet Menu*, SAM also implements a preview feature which improves submenu navigation efficiency. Although it is possible to activate the menu from any location on the screen, it always appears at the same predefined position.

*Two-Handed Marking Menus* [8] is a solution specifically designed for multi-touch mobile devices and two-handed interaction. It allows user to perform two gestures, one with each hand, at the same time, in order to maximize parallelism in hand motions and provide faster menu selections. The ordered variant of the same solution, in which users alternates his/her gestures between two hands, is also implemented.

Semi-circular layouts, like *ArchMenu* and *ThumbMenu* presented in [9], offer a menu design in which all menu items are easily accessible with the thumb. However, due to the utilized layout shape, which increases in size with every new triggered submenu, the number of items a menu can display is significantly constrained.

More recent work refers to *M3 Gesture Menu* [10], a re-conceptualized version of traditional marking menu, which utilizes a persistent screen space and contains menu items arranged in a grid instead of a circular layout. Swipe gestures are used in order to select menu items, and when a certain option is activated, the same space is being occupied by its submenu items. Gesture shapes required for item selections are predefined and depend on related item locations inside the menu.

Radial menu layouts applied to small touchscreen interfaces can occasionally cause interaction burden. Namely, swipe gestures near the screen borders can be difficult to accomplish. However, hierarchical multi-level menus for mobile touchscreens can also be designed by making use of linear schemes. *Leaf Menu* [11] is a type of marking menu implemented with such design, wherein linearly organized items can be selected using swipe gestures. To select the target item from the specific submenu, the user simply needs to lift his/her finger off the screen. *Leaf Menu* supports precise finger interaction, mitigates the occlusion problem, and can be used close to the screen borders. Nevertheless, because all menu items are displayed one below the other, the screen space utilization can represent a problem, depending on the different hierarchy depth. For this purpose, a feature called mirror effect is used in order to compensate for the lack of available screen space.

It is important to mention that both linear and radial menu designs are prone to certain limitations [5]. The number of menu items on each (submenu) level is one of the limiting factors. As the number of menu options increases, less space is available for the corresponding UI widgets. This way targets are becoming smaller and harder to select. Additionally, increasing the menu depth (i.e. total number of submenus) also increases response time. Finally, interacting with complex menu configurations (with higher breadths and depths) usually involves the higher number of incorrectly selected menu items.

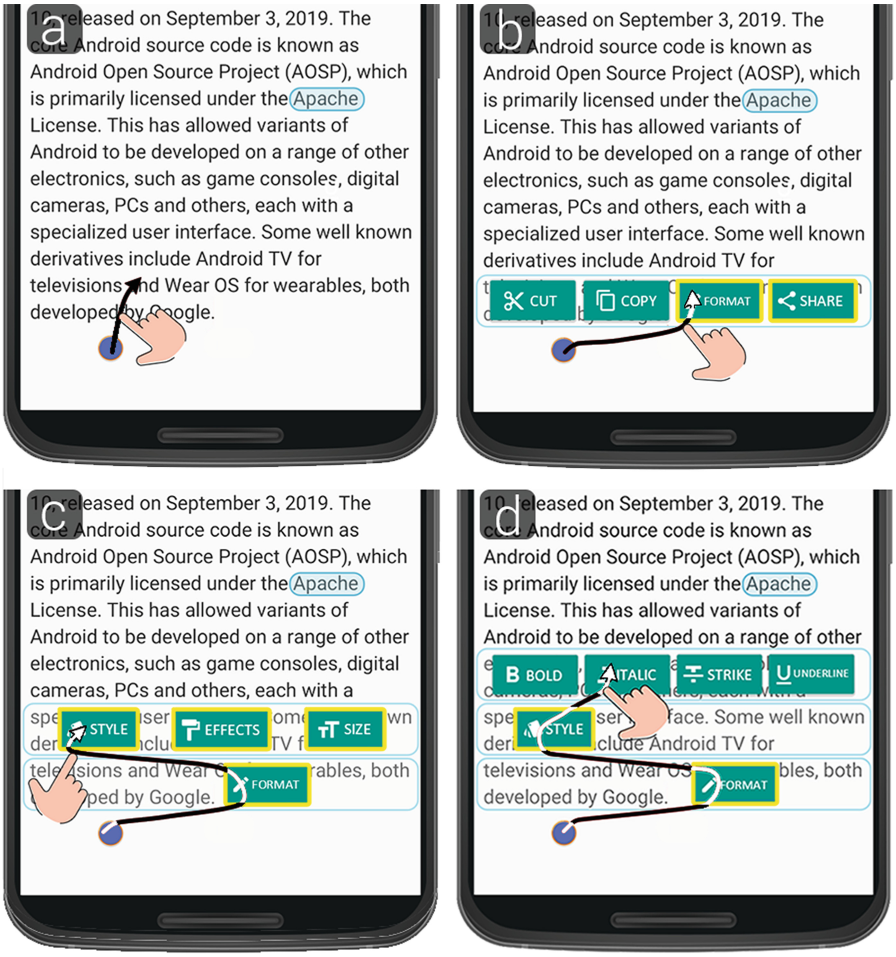
## 3 Floating Hierarchical Menus – *Tile* Menu and *Pie* Menu

Following the motivating factors drawn from the related work, as well as some new ideas about possible enhancements in menu navigation on touchscreen mobile devices, we introduce two different solutions – the *Tile* menu and the *Pie* menu – that allow swipe-based navigation through deep hierarchical menu configurations.

The proposed *Tile* menu design and the accompanying interaction are illustrated in Fig. 1. The top-level menu with associated options is activated using simple swipe movement from the FAB location in an upward direction. Dragging the finger on top of any (expandable) option activates the corresponding submenu above the top-level menu. Thus, the navigation across the menu hierarchy is achieved using a continuous swipe gesture, with submenu levels being stacked on top of each other. All sub-menus are created dynamically, only when needed, according to the user's swipe trajectory. Target option can be selected simply by lifting the finger off the screen, once the swipe gesture has reached (or passed through) the particular menu item. The proposed design allows memorizing swipe gestures for commonly used menu options (similar to the marking menu concept [5, 10, 11]).

To make underlying application content visible as much as possible, we had to replace the usual navigation model in which the menu widget covers up the whole screen or large portion of the screen real-estate. Instead, the *Tile* menu utilizes a semi-transparent visualization and additionally hides all items which are not relevant to the current state of the swipe gesture. As shown in Fig. 1, only FORMAT and STYLE items remain visible once the user swipes to the third level in the menu hierarchy. Hence, the described approach provides several interaction benefits: (i) saving valuable space on small touchscreens, (ii) allowing hierarchical navigation within the same application activity, (iii) item selection by making use of a single swipe gesture, and (iv) retaining the visibility of application content while handling the menu at the same time. It must be noted here that swiping in a downward direction (and stacking submenus in a top-down manner) was not considered as a possible design choice, due to the well-known occlusion issue.

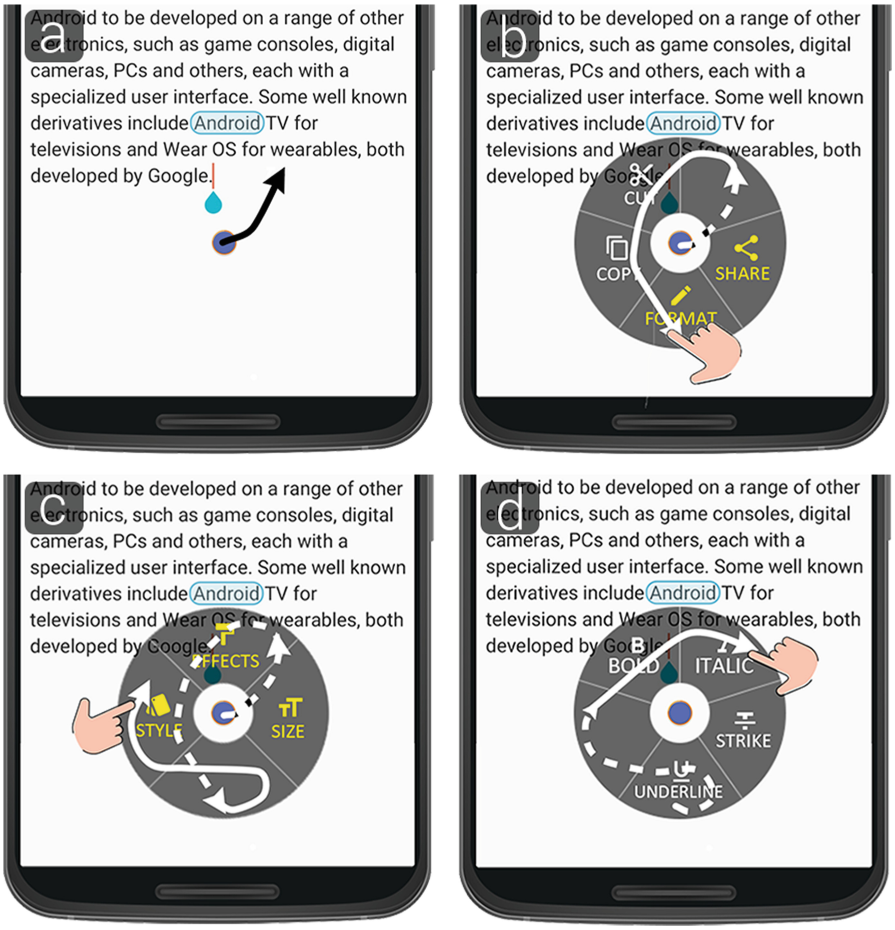
When the menu is minimized and visualized as FAB only, it allows two states of operation. Namely, the menu activation button can be either active or locked. In the locked state, swipe gestures for hierarchical navigation are enabled, and interaction with the *Tile* menu proceeds as described above. However, if FAB enters the active state, menu navigation is then disabled, and FAB itself becomes draggable instead. Consequently, the active state can be utilized for changing the menu position on the screen, according to the users' current needs and individual preferences. Altering the menu pivotal position can become convenient when the context of use is changed (for example, switching between single-handed and two-handed usage). Toggling the FAB state, between active and locked, is enabled via simple tap-and-hold gesture.



10 released on September 3, 2019. The core Android source code is known as Android Open Source Project (AOSP), which is primarily licensed under the Apache License. This has allowed variants of Android to be developed on a range of other electronics, such as game consoles, digital cameras, PCs and others, each with a specialized user interface. Some well known derivatives include Android TV for televisions and Wear OS for wearables, both developed by Google.

**Fig. 1.** *Tile* menu is, initially, minimized and draggable, and can be activated when the user swipes up from its locked state (a). Top-level menu is activated: dragging the finger above FORMAT option will activate new submenu and will hide all other items within the current level (b). Submenu for FORMAT option is placed above the previous container. Item border (yellow) indicates that there is an additional submenu available for a particular option (c). Lifting the finger off the screen activates the last item (ITALIC) along the swipe trajectory (d).

As for alternative design which also utilizes floating principle and swipe-based navigation, we implemented the *Pie* menu – a circular (pie-like) widget that already attracted a number of research efforts in HCI field. In our case, finding the target item within this menu relies on continuous finger movement. The main difference from the *Tile* menu, as well as from other similar radial-based menu solutions [6, 7, 9], lies in the submenu visualization. Namely, instead of dynamically creating extra containers for new submenus, existing elements inside the circular layout of the *Pie* menu are accordingly replaced with new items. The corresponding interaction concept is illustrated in Fig. 2.


Android to be developed on a range of other electronics, such as game consoles, digital cameras, PCs and others, each with a specialized user interface. Some well known derivatives include Android TV for televisions and Wear OS for wearables, both developed by Google.

Android to be developed on a range of other electronics, such as game consoles, digital cameras, PCs and others, each with a specialized user interface. Some well known derivatives include Android TV for televisions and Wear OS for wearables, both developed by Google.

Android to be developed on a range of other electronics, such as game consoles, digital cameras, PCs and others, each with a specialized user interface. Some well known derivatives include Android TV for televisions and Wear OS for wearables, both developed by Google.

Android to be developed on a range of other electronics, such as game consoles, digital cameras, PCs and others, each with a specialized user interface. Some well known derivatives include Android TV for televisions and Wear OS for wearables, both developed by Google.

**Fig. 2.** The *Pie* menu is, initially, minimized and draggable, and can be activated when the user swipes in any direction from its locked state (a). User can invoke a new submenu by hovering (touch-dwelling) over the certain item. Highlighted items (yellow) indicate that there are additional submenus available (b). When expandable option is selected (FORMAT), all items from the same hierarchy level are replaced with new submenu (c). Lifting the finger off the screen activates the last selected option along the swipe trajectory (ITALIC) (d). (Color figure online)

As with the *Tile* menu, the *Pie* menu also allows changing its pivotal position by dragging the FAB to the preferred location on the screen. Placing the finger on FAB in a locked state, and subsequently starting a swipe gesture in any direction will create a circular container around the starting point, with all items from the top-level menu. The container itself cannot exceed the screen boundaries, and cannot be furthermore expanded. If the finger is dragged to the certain (expandable) item, and retained in the same position for a given time (i.e. dwell time), all current menu options are replaced with the new ones from the corresponding submenu. This navigation pattern can be repeated as long as there are available options within the hierarchical menu configuration. The final selection is invoked by ending the swipe gesture once the finger is dragged across the target menu item. When this happens, the menu is automatically minimized to its FAB form.

When compared to the *Tile* menu design, the *Pie* menu generally utilizes less screen space and usually requires less swiping. On the other hand, it is more susceptible to the occlusion problem and allows only the current submenu to be visualized at a given moment.

## 4 Empirical Evaluation

Implementations of the menu solutions so far described are evaluated and comparatively analyzed by conducting a controlled experiment. In this section, we present the details about the experiment design and discuss the obtained results.

### 4.1 Participants, Apparatus, and the Procedure

Thirty participants were involved in empirical research (23 males, 7 females), their age ranging from 19 to 41 with an average of 23,6 years (SD = 3,63). In the introductory questionnaire users reported their personal smartphone models (87% were owners of an Android device), their dominant hand (only one was left-handed), and their preferred hands posture while holding a smartphone (depicted in Fig. 3).

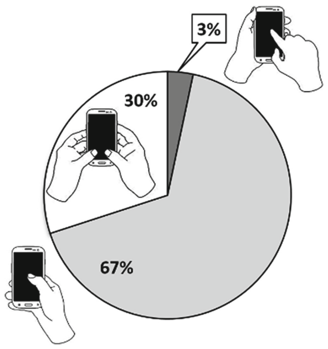


**Fig. 3.** The majority of the recruited participants generally prefer one-thumb single-handed interaction with a smartphone device.

Before the actual experiment, users were involved in the short practice session in order to get familiar with the two different menu designs. While testing the provided solutions, participants were asked to complete twenty different menu navigation tasks in total, ten using the *Tile* menu and the other ten using the *Pie* menu. At this stage, no data was collected whatsoever.

In the actual experiment, we considered two independent variables: menu design (*Tile* vs *Pie*) and interaction style. When addressing the interaction style, we are referring to the way of handling the mobile device. Specifically, we were interested if particular hand posture (single-handed vs cradling) also makes a difference in executing menu navigation tasks. The term cradling [12] corresponds to the use case wherein one hand is holding the device, while the other – usually the dominant one – performs the touch interaction.

For each experiment condition A–D (see Table 1), i.e. combination of the menu design and the interaction style, users were instructed to select certain items from different menu configurations. In total, users were asked to complete 200 menu navigation tasks, of which 50 unique ones had to be repeated within each experiment condition. Experiment tasks were generated according to the available menu configurations with various hierarchy structures. Namely, we defined five menu configurations with different breadths and depths, equal to the ones that can be found in popular text editors, photo editors and integrated development environments. This way we wanted to present users with a rather familiar menu navigation tasks, instead of using fictional mock-ups. The tasks involved both shorter and longer navigation routes, with the menu depth limit being set to six submenu levels. The single task was displayed on the smartphone screen as the required navigation path (e.g. *Format* → *Style* → *Italic*).

**Table 1.** Experiment conditions involved in the empirical evaluation.

| Experiment condition | Menu design/implementation | Interaction style/device handling |
|---|---|---|
| A | *Tile* menu | Single-handed (one-thumb) |
| B | *Tile* menu | Cradling (forefinger) |
| C | *Pie* menu | Single-handed (one-thumb) |
| D | *Pie* menu | Cradling (forefinger) |

Both implementations of the proposed hierarchical menus were tested on a Samsung Galaxy S5 smartphone (SM-G900F) running Android Marshmallow OS. This device is 142 × 72.5 × 8.1 mm large and weighs 145 g. Two different devices of the same model were available for the experiment, so two participants could execute tasks simultaneously.

Two Android applications were developed in order to run *Tile* and *Pie* menu services, as well as to log all relevant interaction events. Both applications are using the same SQLite database which stores information about the menu navigation tasks that have to be accomplished during the experiment. Furthermore, both applications can access configuration files (stored in the internal memory of a mobile device) which contain a description of the menu structure. From these configuration files, specific menu navigation tasks can be defined either manually or automatically. Hence, we provided support for tweaking the experiment settings in an easy way. For example, introducing a new menu hierarchy and new tasks in the experiment requires preparing the corresponding configuration file and utilizing the available task generator.

The time taken to complete the required menu navigation task is considered to be the interval between a first touch inside the FAB (in its locked state) and a finger lift-off event which ends the associated swipe gesture. This task execution time is measured by the application itself, by making use of a built-in monotonic clock which is tolerant to power saving modes and is anyway the recommended basis for general- purpose interval timing on Android devices. All network-based services on the smartphones were turned off during the experiment. In case when target selection was not successfully achieved, the corresponding task was not repeated, and error details were logged along the task execution time.

A repeated measures (i.e. within-subjects) experiment design was utilized. The order of experiment conditions was properly counterbalanced using balanced Latin squares [13], to compensate for possible learning effects. Additionally, the order of the menu navigation tasks in a given sequence was randomized as well. Breaks were allowed when switching between different experiment conditions. Users were also allowed to change menu pivotal position, but only at the beginning of the new task sequence. Replacing the menu position was programmatically constrained in a way that ensures that all menu items stay inside the visible screen area.

After the participants completed all the tasks using both menus and both interaction styles, they were asked to complete a post-study questionnaire based on the rating part of the NASA-TLX (Raw-TLX format). Individual opinions about perceived workload had to be estimated on a 20-point Likert scales for five factors: mental demand, physical demand, frustration, performance, and effort. Subsequently, subjective opinions about usability attributes of the two proposed menu designs were collected. Participants had to rank the menus' usability features, according to their personal preference, using 7-point Likert scales. In the end, general design issues for both the *Tile* menu and the *Pie* menu were assessed. The complete qualitative part of the evaluation was constructed in a way to address the perceived differences between the two menu designs, hence interaction style aspects were not specifically addressed within the related questionnaires.

## 4.2 Results and Discussion

Participants accomplished 6000 menu item selections in total. After averaging data across four experiment conditions, altogether 120 menu navigation performance records were obtained: 30 participants $\times$ 2 menu designs $\times$ 2 interaction styles. Figure 4 depicts the task execution times for two different menu designs, achieved both single-handedly and via cradling.

To analyze the obtained data, a two-way repeated measures ANOVA was used, with *Design* (*Tile*, *Pie*) and *Style* (single-handed, cradling) being the within-subjects factors. The analysis revealed a significant effect of *Design* on menu navigation time ($F_{1,29}$ = 18.783, $p < .001$). The effect of *Style* (i.e. the way of holding a mobile device) was also found statistically significant ($F_{1,29}$ = 8.132, $p < .05$). Finally, the effect of *Design* * *Style* interaction was not found statistically significant.

As for the pairwise comparisons, the differences in task execution times between the proposed menu designs, as well as between interaction styles, are reported as follows:

- *Tile* menu vs. *Pie* menu: (5,002 $\pm$ 0.12 s) vs. (5,443 $\pm$ 0.08 s), $p < .001$
- Single handed vs cradling: (5,334 $\pm$ 0.10 s) vs. (5,110 $\pm$ 0.09 s), $p < .001$

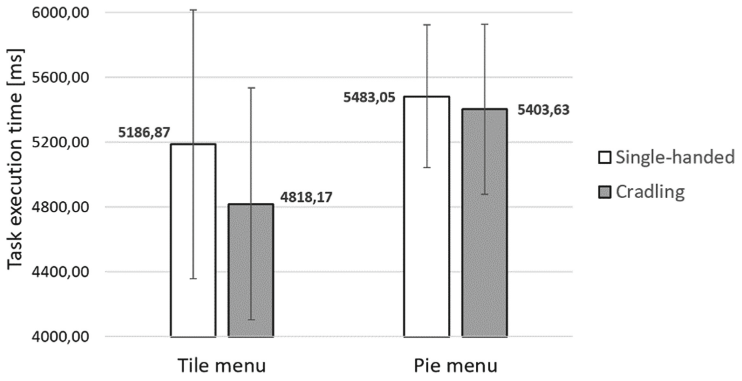


**Fig. 4.** Menu navigation times (mean values and standard deviations) obtained within four experimental conditions.

Thus, the *Tile* menu design showed to be more efficient, as swipe-based navigation tasks were accomplished significantly faster than with the *Pie* menu. However, interacting with the menus single-handedly showed to be significantly less efficient when compared to cradling posture. It can be seen that difference between interaction styles is more prominent when *Tile* menu design is utilized. This can be explained as the result of thumb movement constraints that are inherently higher when deep submenus have to be reached within the *Tile* hierarchy. Specifically, the thumb needs to be stretched more for reaching the submenus at the top part of the screen, which is not the case when forefinger is used or if the *Pie* menu design is applied.

Errors made in menu navigation tasks are reported from the descriptive statistics standpoint only. As shown in Table 2, error rates were rather low in all experiment conditions. Log records revealed that most of the errors were made on the last submenu level, when non-target item was unintentionally selected by ending the swipe gesture on the wrong place.

**Table 2.** Error rates obtained within four experimental conditions.

| | Single-handed | Cradling |
|---|---|---|
| *Tile* menu | 5,13% | 3,66% |
| *Pie* menu | 3,93% | 3,89% |

As mentioned before, questionnaire based on Raw-TLX format was used in order to obtain comparative ratings of perceive workload between *Tile* and *Pie* menu designs. The respective outcomes are shown in Fig. 5.

The Wilcoxon signed-rank tests were used to assess obtained TLX-based scores for each considered factor. The *Pie* menu seems to be generally preferred among participants; however significant difference was found for one factor only. Namely, it was confirmed that *Tile* menu interaction implies significantly higher physical demand when compared

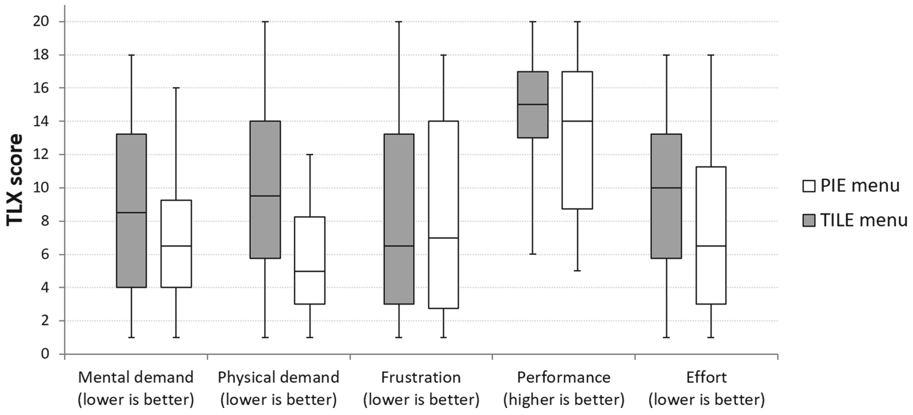


**Fig. 5.** Users' opinions on perceived workload for the proposed menu designs. For each factor, the corresponding box plots show minimum and maximum, 25 percentile (Q1), 75 percentile (Q3), and median value (M).

to *Pie* menu (Z = − 3,699, p < .001). This result validates the aforementioned discussion about particular movement constraints that are imposed by the *Tile* menu design.

Although frustration levels are evenly perceived, it was interesting to find out that sources of frustration are completely different for two menu solutions. According to the users' comments, some were annoyed by continuous UI change within the *Tile* layout, which combines submenu stacking and hiding certain menu items at the same time. Conversely, the *Pie* menu layout is more consistent, but many participants reported dwell time as too long, which made them somewhat irritated, especially towards the end of the experiment. It must be noted here that dwell time was set to 1 s without the possibility to change it during the experiment.

In the first part of the concluding survey, participants used 7-point Likert scales for rating two menu implementations against the ease of use, perceived learnability, and overall satisfaction. The obtained results are presented in Fig. 6. Wilcoxon signed-rank tests revealed no significant differences between the *Tile* menu and the *Pie* menu with regard to usability attributes in question. Nevertheless, score mean values indicate that participants perceive *Pie* menu design easier to use and marginally easier to learn, which automatically explains the outcome of satisfaction level comparison.

Finally, in the second part of the concluding survey, participants had a chance to evaluate design choices implemented within the proposed menus, as well as to assess the suitability of the menus for particular contexts of use. 7-point Likert scales were used for comparative rating once again. Figure 7 presents the corresponding outcomes.

As can be seen, users reported the *Pie* menu to be a more convenient solution than the *Tile* menu in terms of single-handed device usage. This difference was confirmed as statistically significant (Z = −2,324, p < .05) by making use of the Wilcoxon signed-rank test. Thus, the stated arguments on *Tile* menu interaction constraints are corroborated once again. No other statistically significant effects were found.

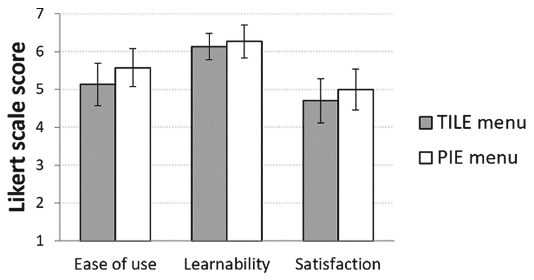


**Fig. 6.** Usability attributes of the proposed menu solutions. Mean values and confidence intervals are presented.

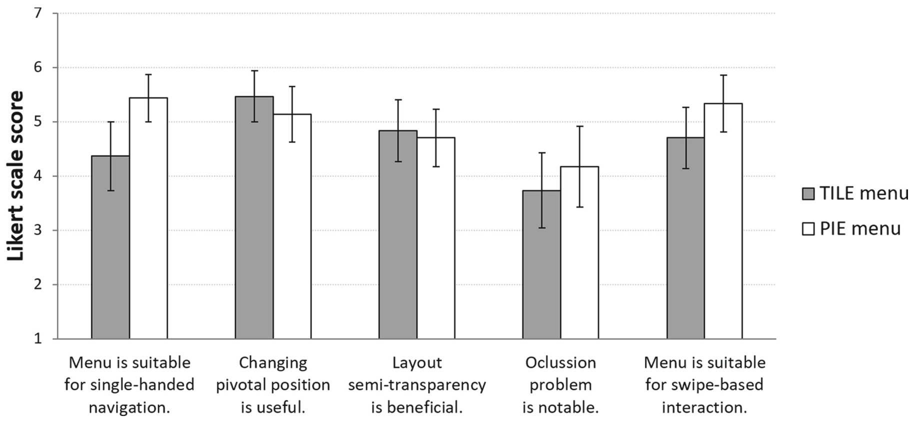


**Fig. 7.** Users' opinions about specific design choices and suitability of the menus for particular contexts of use. Mean values and confidence intervals are presented.

Obtained scores may furthermore indicate that the *Pie* menu is more suitable for swipe-based interaction as well, although the occlusion problem seems to be less prominent when a linear-based layout is used. According to the users' answers, the semi-transparency feature represents a reasonable design choice, because it benefits both menu designs evenly.

Lastly, the possibility of changing menu pivotal position is generally considered useful, with slightly more advantageous effects being perceived for the *Tile* menu. This is in accordance with the log records which revealed that menu repositioning was almost always utilized before starting the *Tile* testing sequence with different hands posture. In general, FAB was predominantly moved at the bottom of the screen when *Tile* menu was utilized, whereas bottom corners (mostly the right one) were commonly occupied with the *Pie* menu. When cradling posture was part of the particular experiment condition, some users tended to move menu layout closer to the center of the screen.

## 5 Conclusion and Future Work

Two different hierarchical menu designs for touchscreen mobile applications, that utilize swipe-based navigation, were introduced and comparatively evaluated in a study involving thirty participants. *Tile* menu implementation applies linear layout, and relies on dynamical submenu stacking according to the gesture being performed. Conversely, *Pie* menu utilizes consistent radial layout and additional touch-dwells for invoking submenus in a multi-level menu hierarchy. Both menus are implemented as an Android service, i.e. they are visualized as a semi-transparent floating widget on top of the target application which can furthermore be repositioned. Navigation across the menu hierarchy is achieved using a continuous swipe gesture, thus avoiding the need for recurrent tapping.

Empirical results revealed that menu navigation tasks are executed significantly faster using the *Tile* design. Single-handed menu interaction, wherein a thumb is used for making swipe gestures, showed to be significantly less efficient when compared to the cradling style in which non-dominant hand is providing device stability. Qualitative analysis, based on the questionnaire outcomes, showed that *Pie* menu is nevertheless more preferred among the participants. It was confirmed that interacting with the *Tile* menu implies significantly higher physical demand, which makes the *Pie* menu a more convenient solution (especially in the single-handed context of use). Other differences between *Tile* and *Pie* menu, in terms of interaction workload, usability attributes, and design-related aspects were not found as statistically significant, but are nevertheless discussed in detail.

In general, the proposed menu designs proved to be a feasible option for swipe-based navigation through deep hierarchical configurations. We believe that these solutions could provide more effective utilization of available screen space, and an easy transition from beginner to expert user.

Our future work plan includes providing further enhancements to existing menu designs. In order to additionally reduce occlusion effects in *Tile* menu interaction, a submenu activation should immediately follow when a specific (expandable) item is selected. In the current version, a new submenu is displayed after dragging the finger above the current-level layout. Due to dwell time being one of the limiting factors in navigation efficiency with the *Pie* menu, our current research efforts are focused on evaluating different dwell time values for such design. Reducing the time needed to automatically activate submenus could potentially provide faster navigation through the menu hierarchy. However, short dwell times could make *Pie* menu interaction more error-prone, because unintentional item selections are more probable in such case. Finally, a longitudinal study should be conducted in order to investigate the effects of memorizing swipe gestures for frequently selected menu items. This way, we could test the marking menu concept, which is inherently involved in our designs, and actually observe the expected transition from novice to expert user.

## References

1. Bailly, G., Lecolinet, E., Nigay, L.: Visual menu techniques. ACM Comput. Surv. **49**(4), 60:1–60:41 (2017)

2. Budiu, R.: Mobile Subnavigation. Nielsen Norman Group (2017). https://www.nngroup.com/articles/mobile-subnavigation/
3. Wiseman, N.E., Lemke, H.U., Hiles, J.O.: PIXIE: a new approach to graphical man-machine communication. In: Proceedings of the CAD Conf. Southampton, vol. 463. IEEE Conference Publication 51 (1969)
4. Callahan, J., Hopkins, D., Weiser, M., Shneiderman, B.: An empirical comparison of pie vs. linear menus. In: Proceedings SIGCHI Conference Human Factors in Computing Systems (CHI 1988), pp. 95–100. ACM Press, New York (1988)
5. Kurtenbach, G., Buxton, W.: The limits of expert performance using hierarchic marking menus. In: Proceedings Conference Human Factors in Computing Systems (CHI 1993), pp. 482–487. ACM Press, New York (1993)
6. Francone, J., Bailly, G., Nigay, L., Lecolinet, E.: Wavelet menus: a stacking metaphor for adapting marking menus to mobile devices. In: Proceedings International Conference Human-Computer Interaction with Mobile Devices and Services (MobileHCI 2009), pp. 49:1–49:4. ACM Press, New York (2009)
7. Bonnet, D., Appert, C.: SAM: the swiss army Menu. In: Proceedings Conference l'Interaction Homme-Machine (IHM 2011), pp. 5:1–5:4. ACM Press, New York (2011)
8. Kin, K., Hartmann, B., Agrawala, M.: Two-handed marking menus for multitouch devices. ACM Trans. Comput. Hum. Interact. **18**(3), 16:1–16:23 (2011)
9. Huot, S., Lecolinet, E.: ArchMenu et ThumbMenu: contrôler son dispositif mobile «sur le pouce». In: Proceedings Conference l'Interaction Homme-Machine (IHM 2007), pp. 107–110. ACM Press, New York (2007)
10. Zheng, J., Bi, X., Li, K., Li, Y., Zhai, S.: M3 gesture menu: design and experimental analyses of marking menus for touchscreen mobile interaction. In: Proceedings Conference Human Factors in Computing Systems (CHI 2018), pp. 249:1–249:14. ACM Press, New York (2018)
11. Roudaut, A., Bailly, G., Lecolinet, E., Nigay, L.: Leaf menus: linear menus with stroke shortcuts for small handheld devices. In: Gross, T., Gulliksen, J., Kotzé, P., Oestreicher, L., Palanque, P., Prates, R.O., Winckler, M. (eds.) INTERACT 2009. LNCS, vol. 5726, pp. 616–619. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03655-2_69
12. Hoober, S.: How do users really hold mobile devices? In: UXmatters. http://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php
13. MacKenzie, I.S.: Human-Computer Interaction: An Empirical Research Perspective. Morgan Kaufmann, San Francisco (2013)