PJL380 : Marking Menus on Android

Guilaume Galliano Bondurand
Télécom ParisTech
46, rue Barrault, Paris, France
guillaume.galliano@telecom-paristech.fr

Mathieu Mansanarez
Télécom ParisTech
46, rue Barrault, Paris, France
mathieu.mansanarez@telecom-paristech.fr

***Abstract*—Clearly, smartphones are a great achievement and also a nice improvement for our lifestyle. However it subsists some problems which are strongly linked to the user interface. Indeed, touchscreen phones are not suitable for exploring the different options of the context linear menus and sub-menus. That's why, in this paper, we present a new and revolutionizing approach to explore the hierarchy of menus in a simple and comprehensible way through the Marking Menus. The aim of this project is to design a marking menu widget which will be dedicated to Android devices. In the end, android developers will be allowed to use the library we provide in their application.**

I. Introduction

A. Description

Marking menus are a combination of pop-up radial menus and gesture recognition. This is a revised form of radial menu. Unlike linear menus where items are aligned sequentially, the marking menus' items are displayed in a circular fashion. The following figure illustrates the difference.

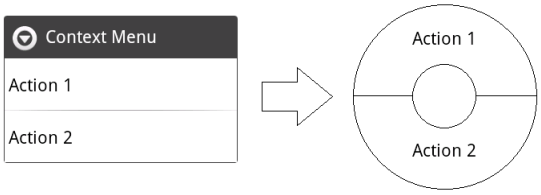

Figure 1. Context menu and Marking menu

When a slice is selected or "marked", the sub-menu pops up and this, indefinitely until the user reaches an leaf item. We have a continual visual feedback of item browsing. The first mode corresponds to the novice mode. In the end, by repetitively using the novice mode, the user would memorize the gesture shape and would be able to draw directly it without even feeling the need of popping up the menu.

Hand-shape interfaces allow a very efficient interaction mode. For the purpose of enhancing the human interface, marking menus have been created but, to date, no android version exists.

B. Usability

This kind of menu is faster than a plain linear menu. It has been demonstrated that it is 3 times faster when the learning process is over. The muscle memory being a form of procedural memory, the body will remember gestures through repetition. The selecting action is then no longer a cognitive one. To enhance the user experience marking menus follow the Fitts's law. If the distance and the size of an item are optimal, the time to reach it will be small. We are therefore working on a widget holding a great potential in terms of learning speed and efficiency.

II. User experience

We let dynamically and discretely the user choose between two modes. The choice will be realized depending on the amount of time the user presses the screen without moving. In order to completely understand the two modes developed in our project, we will see in the following parts screenshots of the menus/sub-menus browsing along with some explanations.

A. Novice Mode

For this mode, the user is discovering the menu and doesn't know any pattern. The menu is popping up every time. The central area is deprived of any component, corresponding to the size of a finger. Our marking menu is depicted in the following figure.

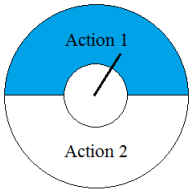

Figure 2. Novice mode - action 1 marked

B. Expert Mode

After a learning process, the user masters the novice mode and becomes an expert. He can now easily navigate in the menu by drawing the pattern leading to the item he needs to reach without displaying the menu. If the pattern doesn't lead to a leaf item, a message is displayed on the screen indicating that the user has to try again.

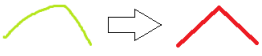

Figure 3. Expert mode - pattern recognition

III. Constraints and assumptions

Working in the mobile environment means you have to accept its constrains. The screen's size limits the user's motions. In the case of the marking menus it would at first makes us consider restricting the item number. Thus, we have decided to set up the number of items to 6. Beyond this value, it would be difficult to display the menu in novice mode and decrypt the scheme in expert mode. Plus, as long as we can make the menu pop up wherever we want, it would be a problem if we proceed actions at the edge of the screen. This have made us think about limiting as well the menu depth.

IV. Effective implementation

In order to provide something that could be integrated in the Android environment, we need to develop a widget in accordance with the standard official Android GUI procedure [1]. Doing this make us first think about the visual representation of our menu and the way we will draw it. It turns out that a simple arc would fit the purpose [2]. The distance to the finger where it will be drawn is deducted from the pythagorean theorem.

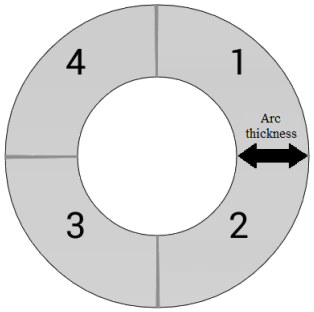

Figure 4. Drawing the widget

When popping up the menu in novice mode at the edge of the screen, we apply an offset so that the depth limitation is no longer a problem. Plus our widget is not drawn truncated.

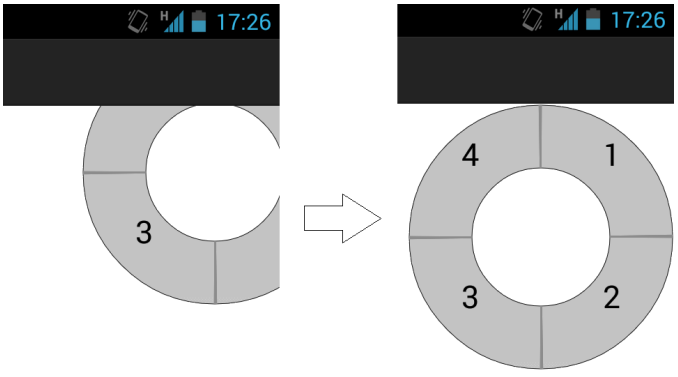

Figure 5. Drawing at the edge of the screen

As long as we would like to let the users choose their menu structure, we provide a simple way of doing it. By means of an item object containing other item objects, we create a bedrock for user's customization.

Regarding the expert mode, the main challenge in the scheme analysis is to determine how many inflection points there are. Because we have limited the number of items per level to 6, an inflection point should present an angle of at least 60°. In this way we are sure that a choice has been made by the user. But what if the choice was to go twice in the same direction? What are we supposed to do? We have chosen to display the menu from the novice mode to help the user know where he is and where he can go.

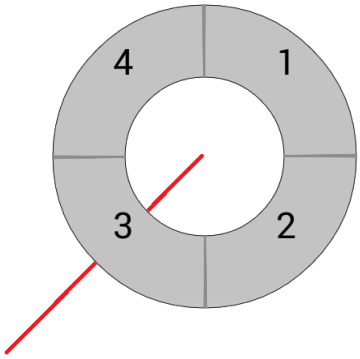

Figure 6. Trying to go twice in the same direction

References

[1] http://developer.android.com/guide/topics/ui/custom-components.html
[2] https://github.com/strider2023/Radial-Menu-Widget-Android