TentaCular - OctoPocus Reloaded

Mobile Information System Project (September 9, 2016)

Magdalena Keil
Bauhaus-Universität Weimar
magdalena.keil@uni-weimar.de

Jula McGibbon
Bauhaus-Universität Weimar
jula.mcgibbon@uni-weimar.de

1. ABSTRACT

Dealing with menus often can be an exhausting and frustrating work. There are a lot of solutions such as categorizing the functions and use pull-down menus that try to minimize the effort of the user. A guide for selecting functions is often hidden in the corner or at the top of the screen. The user first has to route to this position. OctoPocus[2] is a dynamic guide by Olivier Bau and Wendy E. Mackay that starts at the position of the cursor. In this project, we present TentaCular, our adaptation of OctoPocus. We did some slight changes that we will discuss in chapter 4.

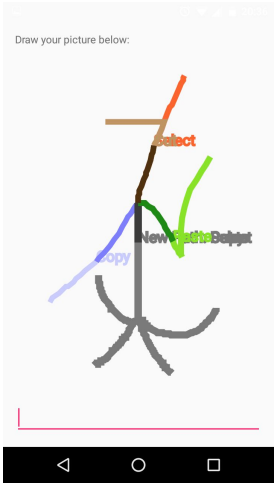

Figure 1: Screenshot of Tentacular. The fingertip's position is in the center of all paths. The paths adjusted to this position. Unfortunately the labels are overlaying. This is due to our design. The colored paths are the paths with the following functions: Select, Copy, Paste and Cut. The grey paths are for changing the colored ones. There is one grey path for each colored path.

2. INTRODUCTION

Gesture recognition or gesture execution is a highly discussed topic, since touch screens began to interfere in our digital interacting world. Every command is evoked by one finger tip. But standards are still missing; meaning, the user doesn't know which gesture will evoke which command, except for the well-known pinch zoom. There are different attempts to create an easy to use and fast mechanism with a range of possible execution commands. One of them are marking menus which we will also place our focus onto. The core concept includes the placement of the menu at the position of the cursor. Furthermore, the selection of a function is done by following the corresponding path that starts at the cursor's position.

3. RELATED WORK

Our project is based on the paper [2] by Olivier Bau & Wendy E. Mackay and embodies the idea, which we were trying to implement. The general idea was to implement a system, which is adapting user's abilities onto the system.

To do so, they introduced an expert and a novice mode. After a few gesture executions, the user might remember the gesture without the need of a menu, which is why it is not displayed anymore. Moreover, it is helpful that the user gets permanent feedback about what he is doing. For instance, when following a gesture path, the already drawn path is displayed as a thin and black path. Also other paths, which become more unlikely, are getting thinner, while the followed path remains in its size, cf. figure 2 below. Especially with gesture recognition, one of the biggest problems is, that the user needs to know which gestures are possible. Here OctoPocus helps the user out with a feedforward mechanism, whereas the user can see all possible paths when touching the screen, cf. figure 2. The prefix marks a small part of the whole path in a deep color starting at the finger position. The rest of the path has a translucent color. All of these points are efforts to prevent the user of frustration.

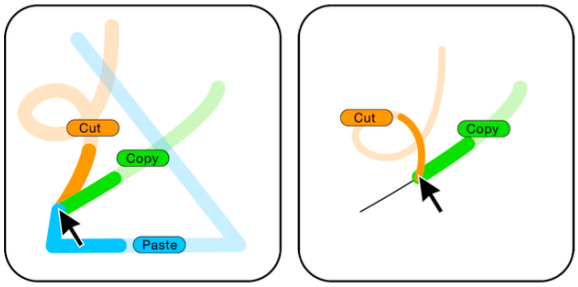

Figure 2: Left: Feedforward mechanism, Right: Feedback mechanism

4. CONTRIBUTION

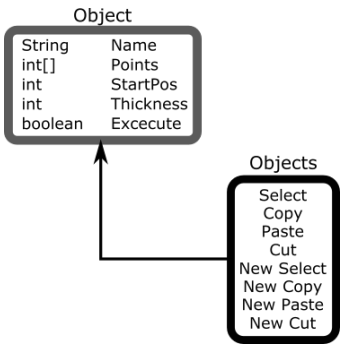

Figure 3: The system knows eight objects: the four text editing paths and the changing paths of the former. Each object consists of a name, the points of the path, the starting position (index of the position where the finger tip is nearest), the thickness of the path and the boolean if the path was drawn and the belonging function can now be executed.)

In the following we will describe our implementation of OctoPocus: "TentaCular".
TentaCular also allows a novice mode and an expert mode. The algorithms are based on the image in Figure 3. All paths are stored as objects. Our system consists of four paths that have different functions. Each path also has a new-path-function that will be explained later. Therefore, we have eight objects. Each object has the following attributes: the name of the path and its function, the points of the path, the starting position of where the path starts and the information whether the user selected this function. In the following we describe the two algorithms (novice and expert mode) in detail, the new path mode and the design.

4.1 Novice Mode

Based on OctoPocus, we created a system that displays a menu starting from the fingertip on the screen. The user can follow the different paths to execute the associated functions. When the finger is lifted up, the menu will disappear and either executes a specific function or does nothing.
The paths we use are given in global space. That means they are starting somewhere in the global coordinate system with a specific size. To display the paths at the right position, we have to transform them into the local space of the finger tip. Therefore we calculate the local position for each global coordinate as follows:

$$local_i = scale * (initPos_i + global_i - global_0)$$

- $scale$: The scale which the local coordinate should have. It is the same for all paths.
- $initPos_i$: The position of the inital finger tip
- $global_0$: The starting position of the path (global coordinates)
- $global_i$: The current position of the path (global coordinates)

Each path is displayed as a feedforward path, which is subdivided into the back part and the prefix part. TentaCular also draws a feedback path. In difference to OctoPocus, the feedback path is not following the exact finger tip position but the selected path. Therefore, every part of the path behind the finger turns from it's origin color into black.

When the finger is moving, the prefix of the paths adjust themselves. For each prefix, our system calculates the euclidean distance between the starting position of the prefix and the finger tip:

$$EuclideanDistance(i) = \sqrt{(x_i * x_i) + (y_i * y_i)},$$

where x_i, y_i is the x/y-difference from finger tip to the point on prefix at index i.

When following a specific path, less likely paths become thinner. We realized that by mapping the euclidean distance between the finger tip and the starting position of the prefix onto the thickness of the particular path. The farther the finger gets from the prefix the thinner becomes the path. If the finger is too far away, the path disappears. If the finger is in the range of the path, the path gets an maximum thickness, so that it does not vary when following the path. When the user has chosen to follow a path, he can decide to go back to the beginning anytime and all paths will be displayed again.

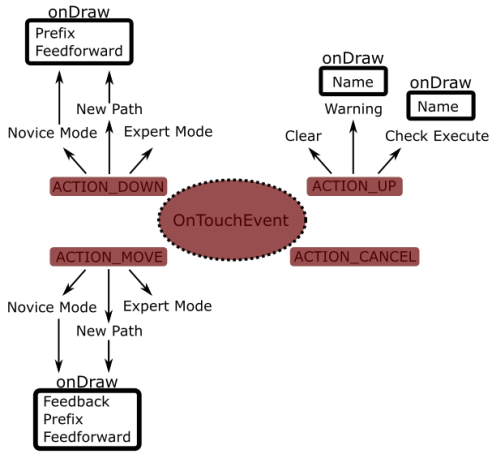

Figure 4: The app is based on the view listener of "OnTouchEvent". Depending on which action is happening, the different modes are chosen. For example, if the novice mode is chosen, the paths are displayed, when the finger is moving. If the New Path mode is chosen, all other paths are shown in grey whereas the new one is drawn in its color. If the finger was moved up, the chosen function is displayed or a warning, if the new path drawing was not done properly.

4.2 Expert Mode

The previous described algorithm is the novice mode. Similar to OctoPocus, we also have created an expert mode. The

expert mode does not include any feedback or feedforward. The paths are not displayed, as an expert user should know all paths.
For the path recognition in the expert mode, we use the 1-Dollar-Recognizer[1] because the expert does not see the paths and can not be as exact as a novice user. The 1-Dollar-Recognizer knows all paths that can be drawn and stores them as template paths. Then it calculates for a given input (for example the path that the finger drew) the score for each of the template paths. The template path with the highest score is the returned result. The name of the function will be shown on the screen to provide the user with feedback.

4.3 New Path Mode

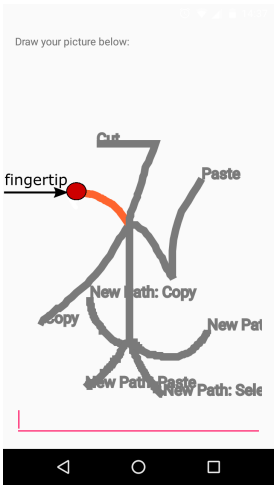

Figure 5: Only the new path is colored. All other paths are drawn in grey. When the finger is moved up, the new painted path will be saved.

If one of the four new paths are chosen, all paths are shown in grey and they can not be drawn with the finger. There is no prefix, only the feedforward in grey. The only path missing, is the colored path that should be drawn new. When the finger starts moving, the feedback is now the new path. In other words, the finger movement describes the form of the new path.
When the movement is finished and the finger is moved up, the new path is saved in its new form and will appear like that in the path menu when there is a new touch down.

4.4 Design

The design of TentaCular is based on OctoPocus. The prefix is painted in the color of the function whereas the back feedforward part is painted in the same but lighter color. The feedback path is drawn in black. The labels appear at the end of the prefix which leads to overlaping in TentaCular (in difference to OctoPocus). The label is also drawn in the same color as the prefix but does not include a bounding box as in OctoPocus.
The paths of TentaCular are not as smooth as the paths in OctoPocus. This is due to the fact that TentaCular allows the changing of paths in real time and they are not straightened by a function.

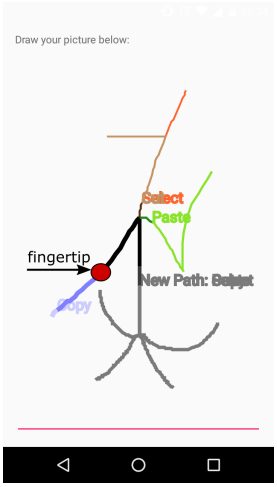

Figure 6: The finger tip followed the copy path. Therefore, the first part of the path is drawn in black, which signalizes the feedback and the prefix moved from the touch down of the finger (middle of path menu) to the current finger tip. All the other parts are still like in the beginning of the touch down. They can only be drawn if the finger would move back to the touch down and would choose another path.

5. CONCLUSIONS

When using TentaCular, it is quite straight forward. Wherever the finger touches the display for a certain time all possible paths will appear in novice mode. Following the specific paths is easy; although, the recognition is a bit slow, so one shouldn't follow the paths too fast. The advantage of this is that the user is less error-prone because it needs to be used slow. That is not the case when using the expert mode. If the finger doesn't rest at the starting point for a certain time, one can draw a specific path without any shown template. It can also work when moving fast, but sometimes it has troubles recognizing a gesture even if it seemed to be drawn correctly, since the user has only a vague image of the gesture in mind.

The "new path" function works as it was meant to be. Right after executing the "new path" function, e.g. for a new copy path, it can be drawn as the user would like to have it. Only too short paths are not saved. The reason for this is that a saved path, which is only a few pixel long, would always be executed when the user is trying to draw something. Moreover, this functionality helps to circumvent the occlusion problem. For example, if one is only using his thumb of the right hand, he can start drawing all paths in the lower right corner.

As mentioned before, a drawback of TentaCular is the design. Many paths can appear quite jittery, since they are not smoothed by some mechanism. Also overlaying labels can lead to confusion when the user tries to map them to the corresponding path.

We can conclude that TentaCular is working well in its functionality but could be improved in its performance.

6. FUTURE WORK

Our work could be extended in multiple different aspects. We will name some of them:

- Right now the paths are not very smooth. The reason is that the paths are generated by adding all points that are recognized by the fingertip which is quite noisy. One could straighten the paths by using a specific technique, e.g. interpolation between some chosen points
- For using the app multiple times with the same setting, one would have to store the latest paths. So that after the next starting of the app the paths would be the same as before.
- The labels could be adjusted so they would not overlay. This would improve the understanding of the paths and would help the user.
- Another extension would be to make the path menu running in the background so that the paths are visible even when starting other apps.
- Also it would be good to give some feedback when one is currently in the expert mode or when the system doesn't respond.
- An also very important feature would be the undo function. One could realize that as an additional undo path.
- When using TentaCular for different devices, one should scale the length of paths respectively to the size of the screen.

7. REFERENCES

[1] $ 1 recognizer. http://depts.washington.edu/aimgroup/proj/dollar/. Accessed: 2016-08-05.

[2] O. Bau and W. E. Mackay. Octopocus: A dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*, UIST '08, pages 37–46, New York, NY, USA, 2008. ACM.