Final Project Report

Fangzhou Shi, Shangquan Sun, Renzhong Lu, Jiaxin Ye

Abstract

Touch screen keyboards are becoming increasingly popular these years. Instead of those pure touch screen devices, such as iPad or iPhone, some devices have a mix of a traditional keyboard and a touch screen input device. The MacBook touch bar and Surface Duo Keyboard are in this case. Many users complain about the performance of the touch bar. It’s hard to find out the reason why it’s not so good without a robust model to simulate the usage. Our aim is to build up a model for these mix keyboards to better predict the human efficiency on these new devices. We will mainly focus on the performance of the touch bar on MacBook Pro. Most of the previous works mainly focus on just traditional keyboards or just touch screen keyboards, which cannot be directly used on mixed devices. Users use tapping on the touch screen which lacks haptic feedback. In addition, we have a multilevel menu on the touch bar to interact with. Considering these effects, we have different models for cognitive and motor operators. By using an alternative strategy of visual search and long term memory, we will provide a solution to predict the efficiency and compare the solution with real-user experiment results.

Figure 1. Apple MacBook Pro Touch Bar

Introduction

Touch screen devices are becoming more and more popular in recent years. Not only the tablets, watches, and phones are using touch screens, but some types of laptops are also using touch screens as keyboards or assistant input devices. In 2016, Apple released a new generation of MacBook Pro with a new type of keyboard which is called 'TouchBar'. It's a stripe of touch screen embedded above the QWERTY keyboard in place of the function keys. People can use the touch bar to do multiple basic tasks, such as taking screenshots, turning the volume or control the music. For some specialized applications, people can also use the touch bar to do specific jobs. For example, when people are using Powerpoints, they can use the touch bar to quickly switch the slides with previews or start the presentation with one-click.

It is announced that the touch bar can improve the productivity of work. However, some people complain that this touch bar increases the time to work compares with the normal shortcut keys. It may be because of the error touch, or people may find it hard to understand the meaning of the keys. It's hard to analyze the real reason why people complain about it without a robust model to simulate the usage. We want to build up a model to predict the efficiency of using the touch bar.

The MacBook Pro touch bar is a new kind of keyboard. It can be considered as a mix of traditional keyboard and a touch screen keyboard. Many previous works just focused on only one kind of keyboard. In the work of Chen *et al*. [1], and the work of Lafreniere *et al.* [2], they modeled and gave two new designs of input methods on smartwatches. They discussed the keyboard layout on a small touch screen, but they just took Fitts' Law into account. The work from Soukoreff *et al.* [3] gave us the method to use Fitt's Law on pointing devices. These models cannot be directly used on the new input devices because more variables need to be considered.

The content shows on the touch bar keeps updating. People have to see it and learn it for different applications and it’s hard to do a blind typing because it’s easy to type a wrong key. Also, the transition time for visual search should also be considered. Current models that we have read in the articles don’t consider these effects.

In our model, we will take into account the perceptual operator, which includes the visual search time. Users will always search from left to right on the touch bar. In addition, we assume that users make errors with a probability. We hope that by using this model, we can correctly predict the efficiency of using hotkey with the touch bar. We can use the time modeled by our code to compare the time of the experiment. By taking in and out different factors, we can find out what reason causes the low efficiency of the touch bar. It can provide us with ideas on how to model other kinds of mixed input devices and have a general idea of their performance.

Related Work

How People Type on Virtual Keyboard

Anna Maria Feit *et al.* [7] create a dataset on people’s daily typing performance and present three behaviors of expert typists. Henze *et al.* [8] design a typing game and compute a touch distribution based on the obtained keystrokes. According to the distribution, the position of key labels could be modified and optimized. Choi *et al.* [9] compare the physical QWERTY keyboard and virtual one and suggest a new kind of virtual keyboard. Findlater *et al.* [10] study the difference between physical keyboard and virtual keyboard on glass screen without touch feedback. The study also computes a touchpoint distribution and shows the finding of expert typists’ habit of typing on a consistent distribution.

Shortcut on Physical Keyboard or Virtual Keyboard

Lafreniere *et al.* [4] study and compare two types of interactions in faster command selection on a small touch screen watch. One interaction technique is a one-step selection involving spatial memory and requiring users to know the locations of items. The other one is a two-step selection.

Adaptable toolbar

Giannisakis *et al.* [5] propose an integrated design of the toolbar interface trying to combine the components of icons of commands, their meanings, and their corresponding shortcuts. But the paper does not discuss the relations between users’ choices of shortcut and icon selection.

New Keyboard Layout

Jokinen1 *et al.* [6] model how novice users adapt a new keyboard layout based on switching from visual short-term memory to recall based search.

Solution

Based on the research on modeling human interaction with the keyboard, we build a model that can capture the human interaction with the keyboard and touch bar. We mainly focus on the touch bar. That is to say, we want to capture how humans use the touch bar while typing.

We are looking for a solution based on the model human processor(MHP) and Fitt's law. We will build a model that includes perceptual, cognitive and motor operators and Fitt's law will be used in the motor operator to calculate the elapsed time of finger movement. In addition to Fitt's law, we also need to consider the time to do the visual search on the touch bar, since the contents on the touch bar change a lot and we cannot use blind typing on it. We include long term memory in our model which can help the users to remember the location of the keys. We take error touch into consideration while we simulate the tasks.

We simulate several different tasks for the users to test, for example, turning up and down the volume when you are watching a video and listening to music, changing the brightness settings of the screen, pressing the escape key on the top left corner and pressing the "fn" key on the bottom left to press the "F1-F12" keys in the touch bar. After getting the data for each of the tasks, we will use it for testing our model. Then we will build a model that includes the three operators in MHP and apply Fitt's law to simulate the movement of fingers. It is hard to test our model on an actual touch bar, so we will build a virtual keyboard and a touch bar in Python and test the model on the virtual touch bar. At last, we will compare the results with what we get from the experiments we did with the human.

We are expecting to see that the results are not too far from one another. Then we can conclude that we have a valid model to simulate human interaction with the touch bar.

Tasks

We plan to start by designing tasks that are related to the touch bar that basically covers its full functionality. Tasks can be categorized into three sections.

First, we aim to measure the performance of pre-programmed tasks that are associated with the previous version of function keys.

1. Volume control including turning up/down and mute.
2. Lightening/Darkening keyboard/screen.

3. Play/Pause/Rewind Media

Second, we have the following tasks to measure the performance of tasks in an application. (e.g. Microsoft PowerPoint) that are undoable with a physical keyboard.

1. Present
2. End show
3. Next slide

Third, complete the tasks in section one in the PowerPoint. In this section, the touch bar will have the layout changes from task to task.

Simulated Keyboard Buildup

We reused part of the code from the assignment to build up the layout of the keyboard. The four different layouts are shown below.

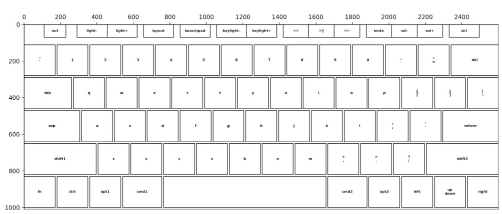

Figure 2. Touch bar menu

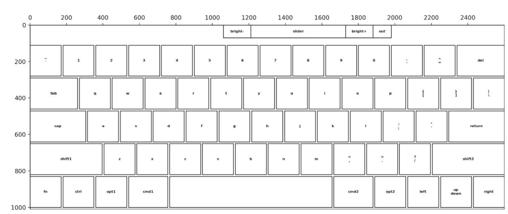

Figure 3. Touch bar brightness

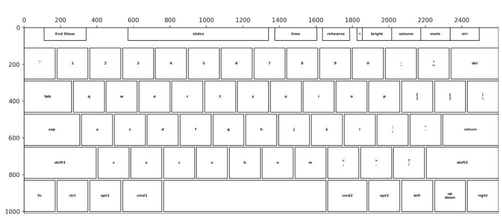

Figure 4. Touch bar PowerPoint present mode

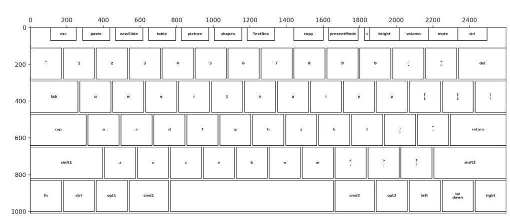

Figure 5. Touch bar PowerPoint menu

We measure the dimensions of the Macbook keyboard. All the length and width of the simulated keys are the same as the physical keys. We have a QWERTY keyboard on the bottom and a strip of touch screen on the top. These four layouts correspond to four different scenarios. The user will start from layout 4. To start the present mode, the user should tap the key “PresentMode” and then he will enter layout 3. To change the brightness, the user will enter layout 2. Layout 1 is prepared for the user who wants to change the brightness of the keyboard light. In this case, we build up four different children for the keyboard in the code. We will switch the layout according to the user’s performance.

Fitts’ Law and Model Human Processor (MHP)

Model human processor is a model that gives an integrated description of human’s psychological knowledge of human’s performance. It can be divided into three interacting subsystems: the perceptual system, the motor system and the cognitive system. The perceptual system consists of sensors and associated buffer memories. The cognitive system receives symbolically coded information from the sensory image store in its working memory and uses previously stored information stored in Long term memory to make decisions about how to respond. The motor system carries out the specified response.

Fitt’s law is used for calculating the movement time of finger moving between two keys. The equation is the following:

$$T = a + b\log_2\left(\frac{A}{W} + 1\right)$$

The hyperparameters a and b are determined by regression. The parameter A is the distance between two keys, W is the width of the target key.

Encode Operator

The encoding operator simulates the process which human uses to perceive the meaning of a key. In the TYPIST model, encoding operator is described as follows:

$$T = -K\log(f)e^{k\varepsilon}$$

In this equation, f represents the probability of a character appears in an English word. In our model, we use the probability of a key appears on one layout of the touchbar. The more frequently the key appears, the quicker the user can encode this key. We think it is the best way to simulate the duration of perceiving the meaning of a key.

Long Term Memory

In our simulation, we considered two aspects of long term memory that may be involved while using the touch bar. The model retrieves both the touch bar menu level and key position at the same time from LTM.

We consider long term memory regarding the key position searching. Before starting the visual search, the model attempts to retrieve the touch bar menu and location from LTM. When the experts are asked to complete a task that is decomposed into a series of keys to press, they spend time remembering the rough position of the key. They have prior knowledge of where the key should be. Since the touch bar menu is hierarchical, they remember which level menu the key lies as well. The time to retrieve layout L_i and location T_i depends on its activation and is obtained as

$$L_i, T_i = F \cdot exp(-fB_i)$$

Where F and f are individual scaling constants.

Visual Search Strategy

The model does not utilize visual short term memory(VSTM) for the reason that it is modeling a one-dimensional visual search. The only direction the testers could go is to the right.

Before visual search starts, the model queries the LTM and tries to retrieve the position of the target key. The contents are shown on the touch bar change between tasks. When the users are using different applications, the touch bar will show different keys on it. Therefore, we need to search for a specific key whenever we manage to perform a task if we do not have prior knowledge of the location. We move the visual focal point from the left of the touch bar to the right to search for a key.

Error Touch

Unlike the physical keyboard, pressing on touch bar results in a higher error rate due to the loss of tactile feedback. To simulate this situation, we propose to add noise when users press on the specific virtual button.

The depth of the touch bar hierarchy is at most two. Therefore, this could possibly lead to layout switching. As long as the tester presses the right button, we mark this as the completeness of the task. The landing position of the touch is the target location with noises added from a normal distribution with a standard deviation of σ.

The time spent to correct such error is added to the execution of the motor operator. Consider the fact that tester could press the wrong key multiple times and after each touch, they become less likely to make the same mistake again. We lower the error rate after the user fixes the mistake by half the standard deviation $\sigma' = \sigma/2$.

Experiment Result

To verify the quality of our model, we first conducted some real user experiments. We found 15 different users who have previously used the touch bar but not quite familiar with operation in PowerPoint. We let them perform 15 tasks in serial and measure the total time of operation. The results are shown below in the table.

Tester	Number of tasks	Time (s)	Tester	Number of tasks	Time (s)
1	15	72.8	9	15	78.3
2	15	80.9	10	15	70.2
3	15	73.1	11	15	79.4
4	15	71.9	12	15	77.3
5	15	62.6	13	15	71.0
6	15	75.2	14	15	69.9
7	15	72.4	15	15	72.3
8	15	74.9		Average Time:	73.5

Model Result

Based on the hypothesis we made above, we implement the code and run the model on the same amount of tasks. Our simulated result is 78.02s. We will discuss this result in the next section.

Discussion

Comparing the results from our model and the empirical results, we can discover that our model can give a description of how human interacts with the Apple Touch Bar not far from the actual results. The cause of the differences may be one or more of the following:

1. The visual search strategies might not be perfect. Some people might not search from left to right. Instead, they quickly scan the whole touch bar to get the position of the intended key. Another possible visual search strategy is search to left from the current fixation point and then search to right from the current fixation point.
2. The encoding method in our model is not perfect. While using the frequency of a key on all layouts as f is a reasonable solution, there are other factors that should be taken into consideration. For example, the shape and whether the shape is easy to understand is a very important factor. There was not enough time for our team to research on the readability of the keys and its relation to the encoding duration.

In addition to modeling the Apple Touch Bar, we also tested the difference between it and the traditional function keys used on the Macbook Air 2012. We asked the above 15 test recipients to do the same tasks on the actual function keys. The average time taken is 69.2s. We have the similar results from our model if we remove the error touch modeling and the result is 73.9s seconds. It showed that one of the biggest deficiencies of the touch bar is it does not have actual keys and people are prone to error touch. If we have the actual keys and we can feel the perimeters, it is less likely for us to error touch. It is one of the possible reasons why Touch bar seems more difficult to use than the old function keys.

Conclusion and Future Direction

In conclusion, our team proposes a model that evaluates the performance of keyboards that contain both mechanical keys and touch screens. We build the model mainly based on the model human processor and Fitts' law. We also consider the visual search and error touch in the model. Compared to previous work which mainly focuses on keyboards or touch screens only, our model will take both digital and physical parts into consideration for a better estimation of human performance on devices such as the new Macbook Pro. We test our model on the tasks we think could be representative of people interacting with the touch bar. The expected estimation is consistent with what we measured empirically. We find that users make lots of mistakes is the reason why the performance of touch bar is not so good.

As smart devices keep evolving at a fast pace, traditional human-computer interaction methods are changing. We present this model to give an insight into whether a new design is better. Our work could be extended in the future. Our current model just consider the performance on the touch bar. It can be extended to model the performance of touch bar when users are using two hands to type on the traditional keyboard. For example, inspect whether the ratio between digital and physical part be a factor that affects people's learning efforts. We believe that our work can be quite useful to model these new devices.

References

[1] Xiang Anthony Chen, Tovi Grossman & George Fitzmaurice. 2014. Swipeboard: A Text Entry Technique for Ultra-Small Devices That Supports Novice to Expert Transitions. Proceedings of the ACM symposium on user interface software and technology. ACM, New York, NY, USA. 615-620.

[2] Ben Lafreniere, Carl Gutwin, Andy Cockburn, Tovi Grossman. 2016. Faster Command Selection on Touchscreen Watches. Proceedings of the ACM CHI Conference on Human Factors in Computing Systems. 4663-4674

[3] R. William Soukoreff and I. Scott MacKenzie. 2004. Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI (Links to an external site.). Int. J. Hum.-Comput. Stud. 61, 6 (December 2004), 751-789.

[4] Benjamin Lafreniere, Carl Gutwin, Andy Cockburn, and Tovi Grossman. 2016. Faster Command Selection on Touchscreen Watches. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). ACM, New York, NY, USA, 4663-4674. DOI: https://doi.org/10.1145/2858036.2858166

[5] Emmanouil Giannisakis, Gilles Bailly, Sylvain Malacria, and Fanny Chevalier. 2017. IconHK: Using Toolbar button Icons to Communicate Keyboard Shortcuts. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17). ACM, New York, NY, USA, 4715-4726. DOI: https://doi.org/10.1145/3025453.3025595

[6] Jussi P. P. Jokinen, Sayan Sarcar, Antti Oulasvirta, Chaklam Silpasuwanchai, Zhenxin Wang, and Xiangshi Ren. 2017. Modelling Learning of New Keyboard Layouts. In Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17). ACM, New York, NY, USA, 4203-4215. DOI: https://doi.org/10.1145/3025453.3025580

[7] Anna Maria Feit, Daryl Weir, and Antti Oulasvirta. 2016. How We Type: Movement Strategies and Performance in Everyday Typing. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16). ACM, New York, NY, USA, 4262-4273. DOI: https://doi.org/10.1145/2858036.2858233

[8] Niels Henze, Enrico Rukzio, and Susanne Boll. 2012. Observational and experimental investigation of typing behaviour using virtual keyboards for mobile devices. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12). ACM, New York, NY, USA, 2659-2668. DOI: https://doi.org/10.1145/2207676.2208658

[9] Daewoong Choi, Hyeonjoong Cho, and Joono Cheong. 2015. Toward Designing a New Virtual Keyboard When All Finger Movements Are Known. In Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15). ACM, New York, NY, USA, 1663-1668. DOI: https://doi.org/10.1145/2702613.2732834

[10] Leah Findlater, Jacob O. Wobbrock, and Daniel Wigdor. 2011. Typing on flat glass: examining ten-finger expert typing patterns on touch surfaces. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11). ACM, New York, NY, USA, 2453-2462. DOI: https://doi.org/10.1145/1978942.1979301
[11] K., Card, Stuart (1983). The psychology of human-computer interaction. Moran, Thomas P., Newell, Allen. Hillsdale, N.J.: L. Erlbaum Associates. ISBN 9780898592436. OCLC 9042220.