Acoustic Keyboard Shortcuts

Jiajia Liang
CS6501 Final Project
Charlottesville, VA
jl9pg@virginia.edu

ABSTRACT
The acoustic Keyboard Shortcut is an alternative approach to use the keyboard shortcuts, achieved by detecting acoustic signal made by users. The project is developed as the final project for CS6501-Engineering Interactive Technologies offered in Spring 2020 at University of Virginia, taught by Professor Seongkook Heo.

INTRODUCTION
Keyboard is a major kind of input device for computer and human interaction. A typical keyboard layout contains typewriter keys, function keys, and cursor control keys. Typewriter keys include numerical keypad, character keypad and other keys such as Tab, Shift, Ctrl and Alt. Undoubtedly, people are very familiar with the keyboard and use the keyboard for all types of works.

There are three interesting observations about keyboard usage that will be briefly discussed. Firstly, the usage of a keyboard is heavily dependent on what type of work people are doing. For example, for a writer, the most common keys might be the characters keys,space-bar and backspace key; For a programmer, Shift + Enter are commonly used for running code snippets; For a gamer, cursor control keys might be the dominant keys. Therefore, people use the keyboard for different purposes, and the patterns of usage vary accordingly.

Secondly, hotkeys are very popular, especially for people whose work involved a lot of typing. Some of the most common shortcuts include Ctrl-C for copy, Ctrl-V for paste, Ctrl- for undo, Alt-Tab for change between windows that are open on the desktop, Ctrl-Shift-Right arrow for select the previous content word by word.[3, 2] However, hotkeys for different tasks lack similarity and it is often hard to memorize the shortcuts, especially when the shortcuts involve Shift, Ctrl and Alt keys. In addition, the mapping of hotkeys is different by the operating system. For example, the copy command on Mac is Command + C compared with Control-C, and Command-Shift- for see all open tabs in one window. [1]

Thirdly, there are some simple tasks that people want to achieve using keyboards, while their hands are not available to type the keys. For example, imagine a person is watching a Youtube video while cooking. It would be hard for the person to press the left arrow if they miss some steps and want to re-watch the previous few seconds.

Based on the above three observations, there is a need to create an alternative shortcut approach which is flexible, simple, and accessible. Among those three goals, accessibility and flexibility is the primary goal, followed by simplicity. Acoustic keyboard shortcuts project aims as an alternative to traditional hotkeys, and it allows users to define hotkeys for the keyboards using acoustic signals. Without the need to physically press the keys, the program is accessible in situations where people can't type on the keyboard but want to perform some simple tasks. The users can create and define their own acoustic patterns with the keyboard commands, which makes it flexible. Lastly, the simplicity goal was hoped to be achieved by using common and easy to learn acoustic beats. More details about the three goals will be further discussed in the evaluation section.

DESIGN
The acoustic keyboard shortcuts used Phyphox app on the phone to collect acoustic data, and processing and simulating the interaction will be performed on laptop. The system overview is shown in Figure 1.

Phyphox App
The program uses the acoustic stopwatch on Phyphox App to collect data. The acoustic stopwatch gets the time between two acoustic events, such as clicks, claps, and tappings. By setting the threshold and minimum delay time, it collects data including the time between two loud sounds in seconds, the average interval, average rate and counts in total. These data will be passed to the processor using remote access built in by the Phyphox app.

Tapping processor
After obtaining the data from the phone, a tapping processor was built using Python to detect the tapping patterns and pair the command value with the tapping patterns. The processor will calculate the time between each acoustic event and categorize it into either "long gap" or "short gap".

Example1: - - - - is encoded as long short short

Example2: - - - - is encoded as short long short

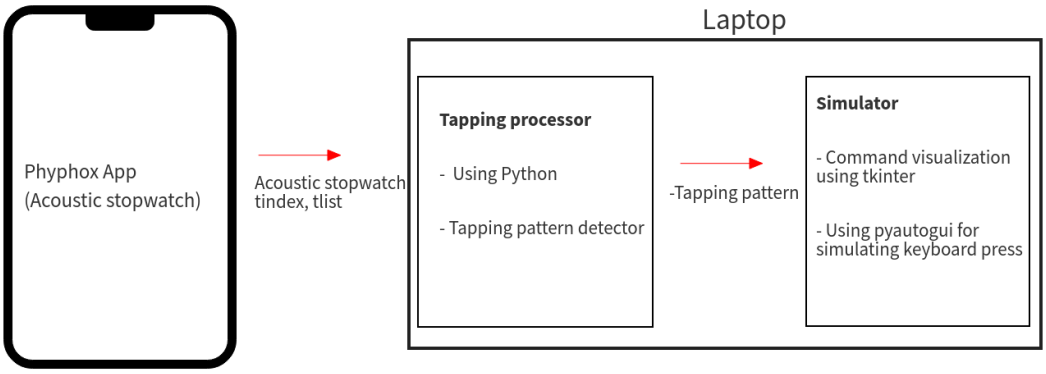

Figure 1. System overview

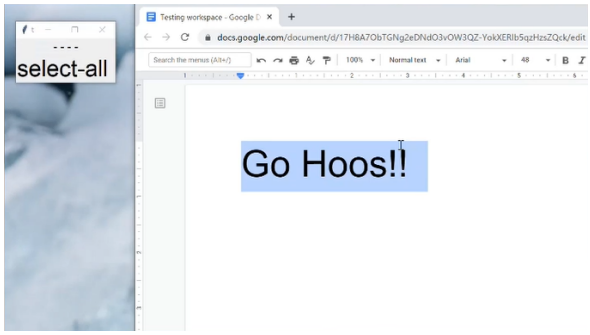

Figure 2. Sample screen visualization when select-all acoustic signal was detected

Simulator

In order to give more positive feedback to the user when experience with the system, a GUI visualizer was added to the system. Once any meaningful pattern is detected, the command will be performed by python pyautogui library, and a pop-up visualization will be shown to inform the user certain task has been successfully performed. The sample screen visualization when a task is performed looks similar as Figure 2.

IMPLEMENTATION

The program was implemented using Python, and involved communication between Phyphox application and Python program on laptop. Below is a brief explanation of the implementation for each part of the system.

Phyphox App

There were two parameters can be set on the acoustic stopwatch: the threshold value and minimum delay value. The threshold value is used to control the environmental noise, and it should be above the environmental noise and below the triggered noise. The minimum delay was used to avoid triggers shorter than the minimum delay value, such as due to echo or reverberation. After the wizard of oz study, setting the threshold value to 0.05 a.u and minimum delay value to 0.05 s can achieve the optimal outcome, where the triggers can be captured most accurately, not interfered with the background noise or the background music.

The PPwas used to configure the phyphox app. This allows the remote accessing data from phone to laptop and communicating the data collected. For the purpose of the project, the tlist data and tindex data obtained from the Phyphox app are passed into the PP_CHANNELS. Tlist is the list of time when the sound events happen. The time is tracked in sequences and stored in seconds, with a very high precision. Tindex is the count of the sound events, which starts with value 0 and will add 1 every time a new sound event is detected.

After the configuration, the data are collected in real time. The receiving Tindex and Tlist were stored into count variable and time variable, and both values are passed to the tapping pattern detector.

Detector

The pattern detector will use the tlist data and tindex data collected and communicated from the phone. The interval detector will calculate the time interval between each two events, and then the pattern detector will return the particular tapping pattern and task command associated with it.

1.Interval detector

The tapping pattern detector takes the count variable and time variable as parameters. Once the count variable has increased, indicating a new acoustic event happened, the program calculates the time interval between the previous events and the new events. The intervals were grouped either as a long interval, a short interval, or a non-significant interval.

Based on the Wizard of Oz study, the time threshold for the three type of interval were summarized as following:

- Short interval (S): interval between 0.01 to 0.25 second;
- Long interval (L): interval between 0.25 to 0.6 second;

- Non-significant (N): interval that is either too long or too short

The information of N acoustic events were transformed to (N-1) interval list using symbol “S””L” and “N” to represent, and the list were being appended every time a new acoustic event was detected. In addition, the pattern detector analysing this list concurrently.

2. Pattern detector

As a prototype, the program is designed to recognize the acoustic signal that includes exactly four beats. Therefore, the pattern detector will analyze the last four elements of the interval list returned by the interval detector. So a full list of possible meaningful combinations “S” ”L” ”N” internal would be NSSS, NSSL, NSLS, NSLL, NLSS, NLSL, NLLS, and NLLL, which the first index were designed to distinguish the start of a new command.

Once the four digit pattern is detected, each sound patterned can be paired with particular tasks which are implemented in the simulator part. Patterns that are not one of the previous 9 patterns will be considered as meaningless and mostly likely created by background noise or unintended triggered sound.

Simulator

1. Command Implementation

Each sound pattern is associated with a task. The keyboard shortcuts tasks were implemented using pyautogui library, which can specify which key or keys the program should press, either simultaneously or sequentially.

2. Visualizer

Once a meaningful pattern is detected, the program triggers a popup window implemented using tkinter library. The pattern and the task that was defined by the users were shown on the popup window as labels. The popup box will automatically close after 1 second.

EVALUATION

Two sets of tests were performed to test the usability of the new interaction.

Scenario one: document editing

In the first scenario, the user is typing and needs to perform basic editing tasks while writing. The acoustic beat and the corresponding task are listed below:

- - - - select all

\- - - - copy

- - - - undo

- - - - paste

The testing involved users typing some words, and then doing the following task in sequences: undo, select-all, copy, paste, paste, undo, and select-all.

- Accessibility: the program is very accessible to use. If the user’s hands are placed on the keyboard, they can easily make acoustic signals by tapping on the outer layout of the keyboard and make distinguishable acoustic signals to trigger the hotkeys. If the user is using a mouse to select content to be copied and pasted, the user can use another hand to make the taps. In addition, in both cases, the users can make acoustic signals with voice, if this is the approach they preferred.

- Flexibility: In terms of the acoustic signal, the program is very flexible and can take in any acoustic trigger made by either tapping on the laptop, on the mouse, on the table, or made by human voice. However, as a prototype, the program arbitrarily defined the acoustic beats with the tasks, such that only copy, paste, undo and select-all can be performed. Future research should improve program flexibility such that the users are able to define their own beats and the tasks associated with each beat.

- Simplicity: Compared with traditional hotkeys, using tapping beats to control might be easier for people who are not familiar or accustomed to the hotkeys, since the tapping beats is more natural to them. However, for those whose work involves lots of typing, traditional hotkeys are more simple since they are already familiar with it, and the time it takes to press two or three keys is shorter than making four acoustic sounds through tapping. Therefore, the simplicity depends on the user group.

- Reliability: By setting the threshold value to 0.05 a.u and minimum delay to 0.05 second, the program is relatively reliable. During the seven trials, the false positive(incorrectly indicated a trigger which is in fact not present), the false negative (miss a trigger that was made by the user), and incorrect identification (incorrectly identify a trigger) were showed in Table 1.

  Therefore, for all 49 tasks performed in the total 7 trials, there are 0 false positive instance, 5 false negative instances, and 0 incorrect identification. So the false positive rate was 0%, false negative rate was 10%, and incorrect identification rate was 0%.

Trail	Results
1	all tasks passed
2	all tasks passed
3	1 false negative on paste
4	2 false negative on paste
5	1 false negative on copy
6	1 false negative on select all
7	all tasks passed

Table 1. Accuracy summery for scenario one experiments

Scenario two: remote control

In this scenario, the user needs to play, paulse, fast forward and fast backward the video. The scenario commands are:

- - - - play/pause

\- - - - fast forward

- - - - back forward

The test involved users watching a video, and performed the start, pause, start, back forward, back forward, pause, and fast forward in sequence. Compared with the previous test, this scenario has more background sounds since the user is watching video with the sound on. Eight testing trails were performed. For the first four trials, the acoustic sounds were made by clapping, and the last four trials were made directly by tapping on the phone.

- Accessibility: Since the scenario was designed assuming it is not convenient for the users to type on the keyboard, either because their hands are not available, or because they are not physically near the keyboard. Therefore, by giving an alternative option for the user to control the keyboard easily by clapping and tapping, the acoustic keyboard controls are accessible to users.
- Flexibility: The flexibility is the same as scenario one.
- Simplicity: In this scenario, we can assume that the user does not need to perform tasks intensely, most likely they just need to use the keyboard control a few times during the entire movie. Even though making an acoustic signal takes more time than actually pressing the key, based on the assumption of usage scenario, the program can be considered as simple and efficient to use.
- Reliability: the threshold value is set to 0.05 a.u. And the minimum delay time is set to 0.05 seconds. The accuracy of the eight trials are summarized in Table.2. Therefore, in the overall 56 instances performed in the 8 trials, there are total 1 false positive, 4 false negative, and 0 incorrect identification. The false positive rate was 1.7%, and false negative rate was 7.1% and 0% incorrect identification rate.

Trail	Results
1	all tasks passed
2	all tasks passed
3	1 false negative on play
4	all tasks passed
5	all tasks passed
6	1 false negative on back forward
7	1 false negative on fast forward; and 1 false positive on play
8	1 false negative on play

Table 2. Accuracy summery for scenario two experiments

DISCUSSION

False positive and false negative rate

Based on the experiment, the program has a good performance in terms of the false positive rate, meaning that it is very unlikely the program performs a task without the intended triggers. However, the false negative rate is relatively large, about 10% false negative rate, meaning that sometimes the users made a command but the program was not successfully captured and performed the task. However, based on the consequences of false positive and false negative, it is more acceptable to miss a command than performing a command that is not intended. Therefore, the threshold values were set to relatively high to control the sensitivity of the program.

Efficiency concerns

From the experiment, especially when the users need to perform a lot of typing, making a sequence of beats is not efficient, compared with traditional keyboard commands in which most keys are being pressed simultaneously instead of in sequences. However, there are still some cases, such as shown in scenario two, where typing on a keyboard is not an option and the demand for typing is not large. In such a situation, it makes sense to compromise efficiency with convenience. Therefore, the acoustic keyboard shortcuts should be considered as an alternative shortcut option, come along with the traditional keyboard shortcuts, but not as a replacement.

Limitations and Improvement

Due to the time constraint and the limitation on performing users testing during the global pandemic, there are few limitations and improvements that can be made in the future. Firstly, instead of arbitrarily setting the tasks with the beats, the program should allow users to define the sound pattern and the tasks associated with it. In addition, currently, the pattern detector was implemented to only recognize sound patterns with 4 acoustic events, and it should be able to detect a flexible number of acoustic signals.

Secondly, the perception of "long gap" and a "short gap" is relatively individual and varied. Since the program was not tested by other users and the thresholds between a long gap and a short gap are arbitrary defined , it remains unclear how well the program performed in more general situations.

CONCLUSION

In this paper, I designed a new keyboard shortcuts approach that aims to be accessible, flexible and simple to use. The program was evaluated in two different scenarios, one is more typing based, and one is more controlled based, and the experiment showed that the program has a high accessibility and moderately high reliability, and its flexibility and simplicity needs further improvement and evaluation. Overall, controlling hotkeys with acoustic signals is worth further development and can be used as an addition to the traditional keyboard shortcuts method.

REFERENCES

[1] Jonny Evans. 2019. 28 keyboard shortcuts Mac users need to know. (6 Dec 2019). **https://www.computerworld.com/article/3023544/28-keyboard-shortcuts-mac-users-need-to-know.html**.

[2] University of Geneva. 2009. Special keys, Keyboard shortcuts, Function keys and Hotkeys. (2009). **https://www.issco.unige.ch/en/research/tutoriel-informatique/EN/special_keys_keyboard_shortcuts_function_keys_and_hotkeys.html**.

[3] Michelle Starr. 2013. Most useful keyboard. (27 Aug 2013). **https://www.cnet.com/news/most-useful-keyboard-shortcuts/**.