Keyboard analysis and visualisation

EE2016 Applied Programming Lab '24

Sanjeev Subrahmaniyan S B

EE23B102

1 Objective/ Problem Statement

Analyse typing patterns for a given input text by plotting heatmaps representing the frequency of key strokes and computing the total finger travel distance for a given input.

2 Solution Idea

2.1 Heatmap and frequency

The problem of finding the heatmap for a given line of text is handled in the following steps:

1. Receive one line of characters as input.
2. Parse the sentence to create a dictionary object which contains the counts of the different key presses. Cases like special characters which require pressing a shift and another base key are covered.
3. Make a plot of a standard keyboard base using matplotlib and patches.
4. Create a blended heatmap using a Gaussian filtering and smoothening of the frequency data from the dictionary.
5. Overlay the heatmap on the keyboard layout and plot the reference chart.

2.2 Travel distance

The problem of travel distance is solved as:

1. Receive one line of characters as input.
2. Parse the sentence into a dictionary representing the keys pressed and the frequency with which they are pressed.

3. Use the given characters and keys dictionaries to retrieve the position of the pressed key and the position of the homerow key corresponding to the same finger that pressed the key under consideration.

4. Compute the distance between the two points. Special characters are handled by considering the shift and the base key to be two separate presses. Add all the distances to get the total travel length.

3 Key asssumptions

There are a number of assumptions that went into this solution due to this being a one week assignment. The most important ones are:

1. The input line contains only English alphabets and special characters on the standard QWERTY keyboard. Other language characters or special characters not covered in such a keyboard are assumed to not be used in the analysis.

2. Only one input line is given - any occurence of newline character or pressing the carriage return will assume the termination of the input and proceed with computation. This is done to comply with the given key and character dictionaries, which do not contain an Enter key.

3. It is also assumed that keys that are not specified in the keys and characters dictionary will never be pressed. They are not valid keys for the execution.

4. All other layouts that are given for testing are assumed to be identical to the example QWERTY layout given in the physical design. This is because some of the special keys have been specially handled to give the keyboard a more aesthetic feel and will fail when keys are not positioned identically to the sample layout. The additional folder has a couple of other layouts which strictly adhere to the current positioning.

5. The Gaussian heatmap is made to ensure the output looks smooth and informative at the same time. It is assumed that the reader can understand the heatmap from the reference chart on the sides. Specifically, the lowest part of the spectrum should be ignored(deep purple in my case), which is created to smoothen the effect.

6. The travel distance for a key is the distance between the homerow key of the same finger of typing to the key that is typed. Specifically, the distance is only considered one way and the return distance is not counted.

7. It is assumed that the user will follow the README file attached and execute the program in the exact steps specified.

4 Finding and results

The key results are based on what the computed finger travel distance is. The files wikipedia-axx.png have the layouts and the corresponding travel distances attached with them. It is clearly seen that the DVORAK and COLEMAK keyboard layouts outperform the QWERTY in terms of raw travel distances and hence increasing the typing speed.

This insightfully points to the fact that the keys in the homerow of DVORAK and COLEMAK are the ones most often encountered in simple english texts. It thus makes sense to place those keys in the homerow when there is no mechanical constraint involved(like that of the older typewriters choosing to use QWERTY).

Another finding points to the high frequency of the Spacebar characters in texts. Given that the average word length in the English language is about 5 characters per word, it is clear that every 1 in 6 key presses are of the Spacebar key. It thus makes sense to use the thumbs exclusively for the spacebar or make another optimal position for it.

It is also noted that the key balance is often not found in the layouts that I considered, with the left hand pressing a larger number of fingers.

I also note that keyboards such as DVORAK and COLEMAK have a lesser typing error rate due to the most frequently used keys requiring little to no movement of the fingers. These add to the net typing speed potetial of these boards.

All of the these keyboards are optimized to type english langugage characters as priority. The special keys and numbers are generally not best accessible in these keyboards and other layouts like programmer are better suited.

5 Conclusion

In all, I infer that it is often not possible to find a layout that is best for all applications. From my experiments over this assignment, I find that DVORAK is the best suited, offering good speeds and error rates over all kinds of inputs including literature and special character texts.