Eyes-free text entry with iPad - Design

1. Introduction

In today's increasingly connected age, it's important to keep in mind that people cannot always be looking at their devices. In some situations, it comes as a socially impolite act of looking at one's cell phone during a conversation, but in other situations, looking down on a cell phone can lead to a potentially life-threatening situation, for example while driving or operating other heavy machinery. This project will look at potential solutions for an eyes-free text entry method that would work well during driving, and other situations of operating heavy machinery that requires a person's full focus on a primary task, where the text entry could be considered a secondary or even tertiary task.

2. Design Space

Within the design space of text entry on tablets/smartphones, there are a couple of input and output modalities worth having a look at.

2.1 Input modalities

Touchscreen - The predominant input modality on tablets is touch, through the touchscreen. Through recent advancement in technologies, now depth/force of the touch can even be measured through '3D Touch' technology on new iPhones (and possibly iPads in the near future). Keyboards are a common form of text input through touch, but these will be discussed separately in '2.3 Keyboard Paradigms'.
Stylus - Mostly used on a touchscreen to provide more accurate, and sometimes a more comfortable interaction as compared to a finger on a touchscreen, especially for individuals with larger fingers. Initially thought to replace keyboards through text recognition, now styli have become more advanced, with orientation and pressure sensors embedded in them as well. Advanced styli are mainly used for art or calligraphy purposes, although the Microsoft Surface Pen is still pushing for handwriting recognition as an important use of its stylus as well.
Mouse/pointing device - Generally only used for text entry in accessibility contexts, and not popularly used on tablet/mobile devices.
Voice Recognition - Through recent technological advancements, voice recognition has become more popular. In the context of text entry, voice recognition is used in the form of dictation, however some problems include lack of inline-editing on dictation, i.e. if

the system hears the user's voice incorrectly, there is very little support in error handling. It also requires an active internet connection for better dictation.
Motion/Orientation - Use of the accelerometer to detect the acceleration and direction of the device, could be used for input of data. This is mostly used for either gaming or activity tracking, but could also be used to sense 3D gestures by moving the whole device. However, using the accelerometer to input 3D gestures may be considered impractical on a tablet due to its relatively large size.
Physical buttons - The device has a couple of physical buttons (sleep, volume rocker, and 'home' button) which may require rooting the device to gain access to, and so we can assume that these will not be used for the application.
Camera - Using Computer Vision, visual cues could also be used as input. This could include 2D gestures drawn with the hand (e.g. drawing letters in the air) or something of the sort. This method would still be considered limited because the amount of computing power that would be needed to run this concept; due to which it may not be a real-time experience.

2.2 Output modalities
Screen - The screen can be used to display a textfield for feedback and a keyboard for feedforward, but for an eyes-free device we will have to assume that these are not available to us.
Sound - The predominant output modality for 'accessibility' systems that work on general consumer devices are sound based feedback systems. The iOS/OSX environment provides "VoiceOver", which involves the system repeating what has been entered. Voice pitches are used to signify different actions, i.e. a middle pitch signifies 'selected', a high pitch signifies 'entered' and a low pitch signifies 'deleted'.
Actuators - The actuators in mobile devices allow them to vibrate to send information to the user. Some users program their phones to vibrate in a certain way to differentiate between phone calls and messages, or certain numbers. Although all modern smartphones have actuators, some tablets may not have this.

2.3 Keyboard paradigms
QWERTY keyboard - The average keyboard layout. VoiceOver uses this as the main keyboard, and so an eyes-free user must know the qwerty keyboard pretty well in order to be able to use this.
DVORAK/AZERTY/Other similar keyboard layouts - There are many other similar keyboard layouts, where the letters are compacted into a rectangular area with 3 staggered rows of keys for the letters, a row for numbers above and a row for the spacebar and other miscellaneous keys below.
Chording keyboard - Seen in "The Mother of All Demos" NLS [1] the idea of the chording keyboard is to be able to input any letter with a combination of keys pressed

on the five-button device, the chording keyboard. However, this requires learning the entire system in order to use efficiently. For an eyes-free input device, we cannot assume the user will be able to memorize the chords, or learn through using a manual while using the system.

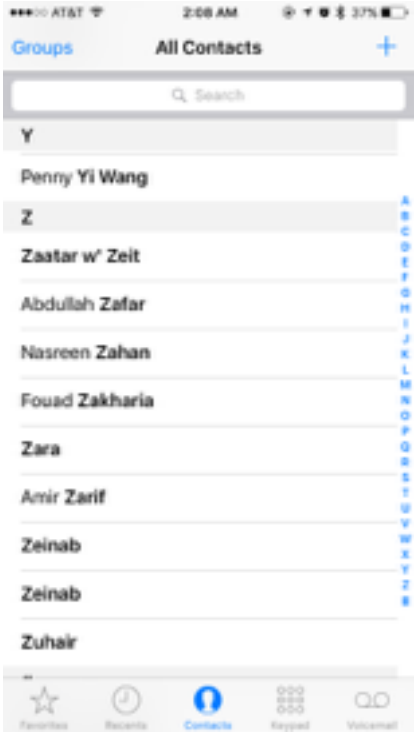
Braille keyboard - Naturally used by blind individuals, these are similar to the chording keyboard for input, and provides a braille output. However, similar to the above, using such a method has a steep learning curve.

3. Brainstorming

For the brainstorming, I decided to consider the inputs and outputs separately. This will allow me to converge on the best combination of input and output in a later phase. Within this, I will be exploring some input methods in different contexts and how they could be translated to the problem of eyes free text entry specifically on a tablet.

3.1 Input

Morse code input - A general problem with the current input system is the lack of spatial awareness while moving along the screen finding keys. This could be solved by removing all the keys, and treating the entire touchscreen as a single button which acts the morse code input. This would mean that the user would be able to tap anywhere on the screen, with no reason to worry about spatial awareness, as the entire screen is acting as a single button.



Keyboard Layout: Linear - Another method to approach the issue of the 'spatial awareness' of the QWERTY keyboard, would be to make it a linear keyboard. With the tablet, we have the luxury of a larger screen real estate, and we would be able to lay out all the keys over the length of the screen. Apple already does something similar with some of the long lists (e.g. contacts, music) and it should be relatively easy to do something like this as a main input method. Using this in conjunction with the VoiceOver feature on iOS devices may work better than using a standard keyboard with VoiceOver functionality.

Fig 3.1 A-Z keys for navigation on right side of screen.

Small square input area - Studying from the success of automaker Audi's design (below) for a touchpad on the console area, this may be a design worth considering.
In the context of a tablet device, I would be able to create a small square input area, and cover it with a paper, with a square cutout to create a 'bounding box' so the user knows that he/she is touching the correct area, similar to the EdgeWrite [2]. This concept is also visible in 3.2.1 above, where there are 'edges' around the touchpad so the user knows they are within the bounds. Within this area, I would consider using the $1 gesture recognizer [3] library, or any other form of single stroke gesture recognizer.

Fig 3.2.1 Audi MMI Touch system: Input

Fig 3.2.2 Audi MMI Touch system: Output

Keyboard layout: Radial - A final way to deal with the spatial awareness issue on the keyboard is to lay out the keys sequentially, but also radially in order to save space. This model would resemble some of the old style telephones, and also a style that BMW automotive has adapted for their *iDrive* [4] interface for their in-car entertainment system's text entry. Below is an example of their system:

Fig 3.3.1 BMW iDrive system: Input

Fig 3.3.2 BMW iDrive system: Output

Though the BMW iDrive system is made for sighted people, it is somewhere in the realm between eyes-free and sight-required - it does not require as much attention as a QWERTY keyboard, since the control knob system was designed to be used by individuals who may be driving while operating the system. In order to adapt this to a tablet setting (with the absence of a physical scroll knob), I took some inspiration from the volume control interaction technique on the Bang & Olufsen BeoPlay H8 headphones (below).

Fig 3.4.1 B&O H8 Headphones: Input

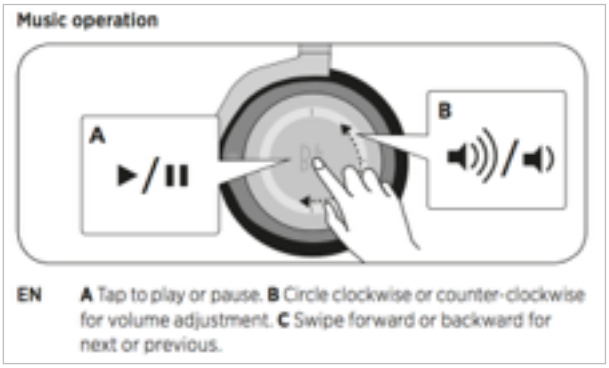


Fig 3.4.2 B&O H8: Interaction Technique

In order to adjust the volume, the user would move their finger in a circular motion alone the edge of the metal cap on the right side of the headphones (Fig 3.4.1). In order to guide the user, however, B&O created a very subtle texture on the surface itself, which can be seen as the slightly darker stroke that follows the circumference of the metal cap. It greatly helps make sure the user follows the circular movement in order to adjust the volume.
Using elements from both the BMW iDrive radial keyboard system and the B&O textured input system, I would consider creating a radial keyboard on the tablet, and laser-cut a stencil/guide to put on the screen that the user can follow to help with the 'spatial awareness' problem that they may face while using a large touchscreen with no guide. This would again be similar to the concept behind the EdgeWrite as mentioned above.

3D gestures by moving device in space - Using the accelerometers inside the device, it may be possible to 'draw' shapes in space and turn those into letters. The $1 gesture recognizer [3] library could be modified to accept 3D gestures in space. This would somehow resemble using a Nintendo Wii controller, but with the tablet/phone itself.

Voice Recognition - Dictation on iOS has come a long way through the improvements in Siri, however there has still been a problem with the editing, if the device interprets what the user dictates wrongly. Extending the functionality to allow for better

correction of mistakes will go a long way in improving the technology. This could be done by introducing 'escape phrases' that will be treated not as words to be transcribed, but as commands for text editing. Escape phrases could include:

- 'no, write _____ instead',
- 'no, delete _____' or
- 'read it out', etc.

As an interaction technique, this would work much better on longer forms of text entry, e.g. for writing an email or something similar. For shorter forms of text entry, e.g. searching for a contact name or a song in a playlist, it may become more tedious if the user needs to memorize escape phrases that may themselves be longer than the words/phrase that the user is trying to input.

3.2 Output

Visual output for text entry works well because, apart from requiring the user to look at the screen, it is not obtrusive in any other way. In this case, obtrusive refers to being a distraction to others around the user, e.g. a sound-based system would disturb those around the user (unless the user wears headphones) and an touch/vibration based system may also disturb those around the user, unless it is haptic feedback that only the user can feel (e.g. that provided on an Apple Watch).

Morse code through actuators - One way to deal with the lack of visuals is to use the actuators in the device to communicate to the user using morse code. Upon the input of each letter, the system would repeat it back to the user. If the system interpreted it wrongly, the user may swipe the screen and input it again. This could work well with the combination of the morse code input system. Since this project is designed for a tablet, haptic feedback would not be possible, and actuators are not present in many leading tablet devices (although they are present in most mobile devices).

VoiceOver - The most logical framework to use on the iOS platform is the VoiceOver accessibility system to assist with the input and output, which translates all text on the screen to voice when the user touches over it. In order to not cause a disturbance to others, this may be used with a set of headphones.

4. Convergence

In order to avoid setting a very high learning curve for the user, I have eliminated the idea of using morse code to communicate with the system, even though for people well versed in morse code this may be one of the most effective options.

In order to follow with my motivation for creating an eyes free interaction technique being used in certain situations where the user is not able to look at the screen, most importantly while driving, or operating other machines (transportation or otherwise) I will need to eliminate the movement of the device in 3D space, due to both limited

space in a vehicle and possible distractions, and since working in 3D space may require a slightly higher cognitive load from the user.
Understanding the context of my motivation, it feels like in an average driving scenario the user would not try to do long forms of text entry, but rather shorter ones like inputting the name of a contact, or a song in a playlist. Along with that, there may possibly be music playing, or outside noise that may reduce the accuracy of the voice recognition system. For this reason, we will have to eliminate the voice recognition interaction technique.

Through elimination, I removed the morse code input/output, the 3D gestures and the voice recognition technique. I believe at this point, it would be feasible to explore the two different types of keyboard layouts, as well as the gesture recognizer as input methods, and default at VoiceOver to handle the feedback. I believe one of the most important things to keep into consideration on a touchscreen setup is the spatial awareness, and for that reason, all three of these methods will involve some sort of template/stencil that covers part of the screen and exposes parts that the user can touch, utilizing the edges of the cutout as guides to help the user stay within the active bounds of the screen, similar to what Wobbrock et. Al did in EdgeWrite [2].

5. Implementation

After converging to the three interaction techniques noted above, I went on to try implementing them on an iPad/iPhone app. I chose to work with Swift/iOS in order to take advantage of Apple's powerful VoiceOver software, and the ability to make an app that works on both tablet and mobile, responsively. Using Xcode's development environment, I was able to quickly test out designs on both iPad and iPhone, in both portrait and landscape mode in order to see what orientation/keyboard layout would work most effectively and comfortably, compared to the baseline test we will be doing in Homework 2.3, and with the context of in-car eyes-free interaction techniques in mind.

Upon exploring techniques, I quickly decided to eliminate the Dollar Gesture Recognizer. I looked at an open source version of recognizer made for Swift (https://github.com/malcommac/SwiftUnistroke) but the problem seemed to be that this was not as accurate as I hoped - due to the fact that the recognizer is not angle agnostic, letters like 'C' and 'U' were getting confused. I also felt that this technique would require a steeper learning curve (as compared to a keyboard), requiring more time that the user has to practice before becoming good at it. This could cause a problem in the driving setting, where it may be hard for the user to deal with the 'recalling' the required gestures rather than 'recognizing' the correct keys as on a keyboard. Lastly, I

felt that the friction caused between the finger and glossy screen when doing freeform writing may become a frustration for the user over time, as will the constantly smudged screens requiring frequent cleaning.

Apart from the unistroke gesture recognizer, I was hoping to try out two different keyboard layouts with VoiceOver: a radial keyboard and a linear keyboard. I decided to move forward with a linear keyboard because it took less screen real estate, and was easily translatable between different devices, screen sizes and orientations. I tried both a horizontal keyboard and a vertical keyboard, and had the advantage of putting both a lower case and upper case keyboard next to each other.

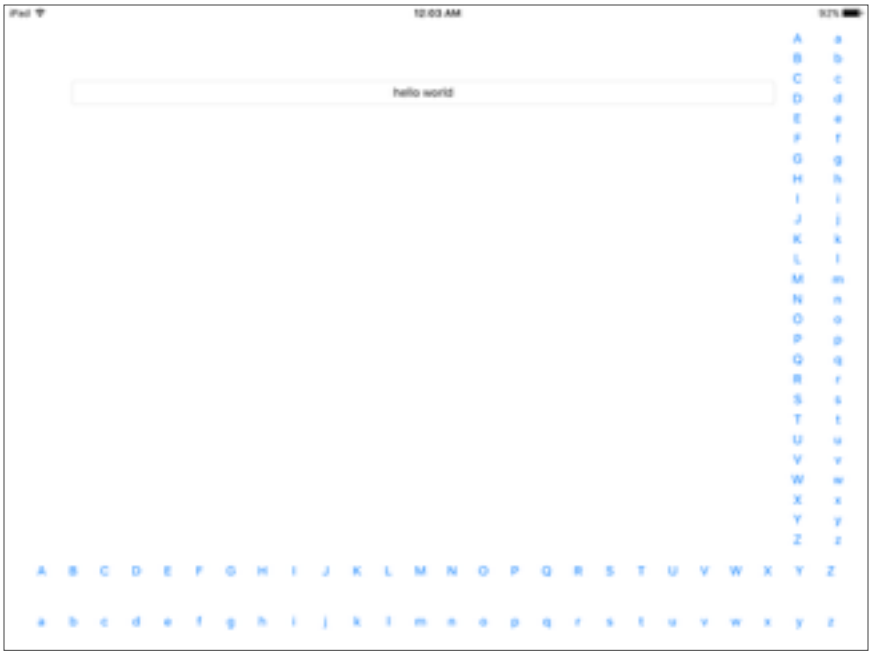


Fig 5.1 Screenshot of final keyboard layouts

Using Apple's VoiceOver software, I was able to make it such that the keys that the user touched were spoken back when touched, and spoken back at a higher pitch when clicked (through a double tap). To backspace, the user would swipe left, and to add a space, the user would swipe left. In order to maintain a sense of spatial awareness, I added a layer of tape to be used as an 'edge' for the keyboard (Fig 5.2). Given the fact that the context I was designing for is a driving situation where the types of actions one may conduct only involve around 2-3 words (searching through names of contacts or music), special care was taken to ensure I did not add extra features that may not be required for this purpose (e.g. text-to-speech for textfield or backspacing words at a time rather than characters).

Fig 5.2 Screen with tape adjacent to vertical keyboard acting as edge

6. Conclusion

[to be completed in 2.3]

7. References

1. Engelbart, D. C., & English, W. K. (1968, December). A research center for augmenting human intellect. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I* (pp. 395-410). ACM.
2. Wobbrock, J. O., Myers, B. A., & Kembel, J. A. (2003, November). EdgeWrite: a stylus-based text entry method designed for high accuracy and stability of motion. In *Proceedings of the 16th annual ACM symposium on User interface software and technology* (pp. 61-70). ACM.
3. Wobbrock, J. O., Wilson, A. D., & Li, Y. (2007, October). Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *Proceedings of the 20th annual ACM symposium on User interface software and technology* (pp. 159-168). ACM.
4. Niedermaier, B., Durach, S., Eckstein, L., & Keinath, A. (2009). The new BMW iDrive–applied processes and methods to assure high usability. In *Digital Human Modeling* (pp. 443-452). Springer Berlin Heidelberg.