DigiTaps

Vaspol Ruamviboonsuk
Computer Science and Engineering
University of Washington

May 10, 2014

Abstract

Visually impaired people uses touchscreen on smartphone with screen reader. The user uses audio feedback from the screen reader to navigate the phone. Since this method requires a lot of attention to the audio, it is difficult to use in a noisy setting. We introduce DigiTaps, an eyes-free number entry method that uses minimal voice feedback. Users can enter digits by tap or swipe anywhere on the screen. The digits are represented by two different DigiTaps codes. We conducted a preliminary study to compare DigiTaps method to the screen reader method. The participants achieved a significantly lower error rate by using DigiTaps method than using the screen reader method. To conduct a more rigorous study, we develop DigiTaps game. DigiTaps game is a number entry game designed for conducting user studies in the large. We made it available on the Apple App Store. We use it as a platform to collect the players data such as touch events and game playing statistics. With the data collected, we plan to analyze the data to look for interesting information and evaluate the two methods.

Contents

1 Introduction

Visually impaired people uses touchscreen on smartphone with screen reader. Screen readers, such as Apple VoiceOver [1], provides audio feedback for the user to navigate the phone. The user uses his finger to find the target item on the screen. The screen gives continuous voice feedback. Once the target is reached, the user performs a selection gesture, either a split-tap or a double-tap anywhere on the screen. We can see that using screen reader in a noisy setting is difficult, since the user has to pay careful attention to the voice feedback. Wearing headphones is not an option because blind users feel unsafe since they use the sounds around them to navigate and understand their surroundings [3].

We introduce DigiTaps, an eyes-free number entry method that uses minimal voice feedback. Users can enter digits by tap or swipe anywhere on the screen. The digits are represented by two different DigiTaps codes. For example, the digit 2 is entered by performing a two-finger tap anywhere on the screen. We conducted a preliminary study to compare DigiTaps method to the screen reader method. The participants achieved a significantly lower error rate by using DigiTaps method than using the screen reader method.

We developed DigiTaps game for conducting number entry method in the large. DigiTaps Game is straight forward. It shows a number to the player and the player uses DigiTaps code to enter the number. We collect data at every touch event that the player has performed along with the information on whether the player entered the number correctly or not. We plan to use the data collected for further analysis on the evaluation of the DigiTaps code.

2 Related Works

2.1 DigiTaps Code

Blind users use smartphones by using the screen reader that is built-in with the phone such as Apple's Voice Over [1]. Android also has TalkBack, which is a screen reader on Android [6]. Both screen readers use the interaction techniques that Kane et al. presented in Slide Rule [10]. To perform a selection, a user has to use one finger to explore the screen. The screen reader provides audio feedback of what is beneath the user's finger. The user has to carefully listen to the audio feedback. When the target is reached, the user performs a selection gesture which can either be a double-tap anywhere on the screen or a split-tap (tap another finger on the screen while the first finger is still on the selection target). Even though these screen readers are widely adopted in the blind community, using screen readers to enter text achieves a low the text entry rate. Blind users can enter text at only 4.5 words-per-minute using VoiceOver [4]. Furthermore, using screen reader method is onerous and error-prone [13].

Many eyes-free text entry method has been introduced recently. Azenkot et al. presented PerkInput, a chorded input for touchscreens where Braille cells are input one column at a time [4]. Southern et al. presented BrailleTouch, an application for typing braille based on the six-key braille keyboard [15]. These text entry methods require finger calibrations or allow the user to touch in only certain locations on the screen. In contrast, users are not required to calibrate their fingers with the phone and can perform the gesture anywhere on the screen. Furthermore, PerkInput and BrialleTouch requires knowledge on braille to use them while DigiTaps does not require the users to know braille.

MacKenzie et al. introduces an eyes-free text entry method using a joystick called H4-writer [12], which achieves a 20WPM text entry rate. Like DigiTaps, H4-writer uses an optimal prefix-free code, specifically Huffman Code, as the input to input text. Unlike DigiTaps code, however, the Huffman Code does not relate to the actual notations of the symbols. Thus, it is unlikely to be as intuitive as DigiTaps' code which all the codes are related to the actual notations of the symbols.

Blind users do not have the knowledge of how the print symbols look like. While using the print symbols in the text entry method is efficient for sighted users, blind users can have trouble performing them because their drawing skill would be less accurate than sighted users [11]. Graffiti and Unistroke [5] uses print symbols for representing each character. Unlike Graffiti and Unistroke, DigiTaps uses simple gestures like taps and swipes to represent its symbols.

2.2 DigiTaps Game

Henze et al. conducted a user study in the large on analyzing touch performance [7]. An text entry Android game was developed and distributed via Google Play Store. The game collected important data from the players such as touch positions to analyze the accuracy and the performance of the touch events. We adopted the study design presented in [8] and developed DigiTaps game as a platform for conducting number entry method user studies in the large.

3 DigiTaps Codes

Numbers can be represented in different ways using different numerical representation system such as base-2 or base-10. In the daily use, numbers are represented using the base-10 numeric system. The numbers can also be represented in base-5. The digits 0 through 9 will be represented as 0, 1, 2, 3, 4, 10, 11, 12, 13, 14 respectively. We call these representations as codes. There are two types of codes: fixed-length and variable-length. The length of all the codes represented by fixed-length are the same. For example, if 0 through 9 are represented with fixed-length code in base-5, we can write them as 00, 01, 02, 03, 04, 10, 11, 12, 13, 14 and the length of each code is 2.

In contrary, variable-length codes can have variable length. For example, if 0 - 9 are represented with fixed-length code in base-5, we can write them as 0, 1, 2, 3, 4, 10, 11, 12, 13, 14. The first 5 code have a length of 1 and the rest of the codes have a length of 2. We can see that variable-length codes are shorter on average. Since the variable-length codes are potentially shorter than fixed-length codes on average, we decided to use variable-length codes for representing the numbers.

Although variable-length codes can represent numbers with a shorter length than fixed-length codes. It introduces an ambiguity issue to the code. For example, suppose we have a program to decode base-5 variable-length codes to base-10 digits. We enter the number 1 into the program. The program cannot justify whether the number 1 suppose to be the code 1 or the first digit of 11. The problem does not arise in fixed-length codes because all the code have the same length. Thus, the digit 1 is represented by 01. To resolve the ambiguity, we use a variant of variable-length code called Prefix-free code.

Prefix-free codes do not allow a code to be a prefix of another code [9]. In our example above, if we use prefix-free code to represent 0-9 in base-5, there cannot be 10, 11, 12, 13, 14 because 1 is already a code by itself and it cannot be a prefix of another code. Since using prefix-free code removes the ambiguity among the codes and it is a variable-length code, we decided to use prefix-free code to represent our codes in DigiTaps.

3.1 Espresso

We develop a learnable prefix-free code called Espresso. With Espresso, the digits can be derived by adding the numbers 0, 1, 2, and 3. For example, 0 is represented by one finger swipe, 1 by a one-finger tap, and 2 by a two-finger tap. 3 is represented by three-finger tap and a swipe (3+0). Similarly, 4 is represented by three-finger tap and a one-finger tap (3+1). 6 is represented with two two-finger taps and one-finger swipe (3+3+0). The final swipe is required to ensure that the code is prefix-free. Finally, 9 is represented by three three-finger taps (3+3+3). A swipe is not necessary for 9 because three three-finger taps is not a prefix to another input.

The numbers 0 through 2 in Espresso are 1 gesture long. The next 3 digits, 3 through 5, are 2 gestures long and the rest of the numbers are 3 gestures long. This gives us an average gesture per digit of 2.1.

$$avgGestures_{Espresso} = \frac{1 \times 3 + 2 \times 3 + 3 \times 4}{10} = 2.1$$

Table 3.1: Espresso Codes

Digit	Code
0	1-finger swipe
1	1-finger tap
2	2-finger tap
3	3-finger tap + 1-finger swipe
4	3-finger tap + 1-finger tap
5	3-finger tap + 2-finger tap
6	3-finger tap + 3-finger tap + 1-finger swipe
7	3-finger tap + 3-finger tap + 1-finger tap
8	3-finger tap + 3-finger tap + 2-finger tap
9	3-finger tap + 3-finger tap + 3-finger tap

3.2 Cappuccinno

We develop an optimal prefix-free code that has a lower gestures per digit on average called Cappuccino. In Cappucinno, one-finger swipe represents either 0 or 10. A two-gesture digit is derived by subtraction, with the one-finger swipe representing 10. For example, 9 is represented by a swipe followed by a one-finger tap (10 - 1). Similarly, 8 is represented by a swipe and a two-finger tap (10 - 2) and 7 is represented by a swipe and a three-finger tap. In all other cases, one-finger swipe represents 0. For instance, the digit 3 is represented by three-finger tap and a swipe (3 + 0). However, the digit 0 is represent by two one-finger swipes. The second one-finger swipe is added to make the code-prefix free. Furthermore, 6 is represented by two three-finger taps. Espresso and Cappuccino have the same code for the digits 1 through 5.

The numbers 1 and 2 in Espresso are 1 gesture long. The next 3 digits, 3 through 6, are 2 gestures long and the rest of the numbers are also 2 gestures long. This gives us an average gesture per digit of 1.8.

$$avgGestures_{Cappuccino} = \frac{1 \times 2 + 2 \times 8}{10} = 1.8$$

Table 3.2: Cappuccino Codes

Digit	Code
0	1-finger swipe + 1-finger swipe
1	1-finger tap
2	2-finger tap
3	3-finger tap + 1-finger swipe
4	3-finger tap + 1-finger tap
5	3-finger tap + 2-finger tap
6	3-finger tap + 3-finger tap
7	1-finger swipe + 3-finger tap
8	1-finger swipe + 2-finger tap
9	1-finger swipe + 1-finger tap

3.3 Analysis of Prefix-free Codes

Espresso uses 2.1 gestures on average per digit. This is close to fixed-length base-4 notation, which is 2 codes per digit long. However, the optimal code uses 1.8 gestures on average per digit. However, there exists many distinct optimal codes, which can be derived from the following equation.

$$\binom{4}{2} \times 10! = 21,772,800$$

There are 6 ways to choose two one-gesture codes from the 4 gestures, and there are 10! possible combinations for assiging the 10 digits to the 10 resulting codes.

4 Preliminary Study

To evaluate the potential of the gestures, we compare the Espresso method to a standard accessible numeric entry method using (1) a theoretical analysis of the methods and (2) an empirical comparison with five users. We choose the Espresso method over the Cappuccinno method because it is more intuitive to learn.

4.1 Theoretical Analysis

In the standard method, there are two steps involved in entering a digit. First, the user has to explore the screen to locate the button where the digit resides. The seek time is hard to quantify, but it requires listening to the buttons touched until the correct one is found. Once the button is located, the user performs a selection gesture. This can be done using a split tap, hold a finger down on the target button and use another finger to tap the screen, or a one-finger double tap anywhere on the screen. At a first attempt, this method appears to be more difficult than the 2.1 taps per digit.

4.2 Empirical Evaluation

Our empirical evaluation consists of a study with five sighted participants. In each of the study, the participants entered 10 six-digit numbers using the standard and the Espresso methods. Participants held the smartphone beneath the desk so that they were not able to see the screen. After a brief practice session, the participants entered the numbers at an average rate of 1.99 seconds per digit ($SD = 1.25$) using the Espresso method and at an average rate of 2.77 seconds per digit ($SD = 1.24$) with the standard VoiceOver method. The error rates were far lower for the Espresso method.

The participants produced Mean String Distance (MSD) only 1% on average using Espresso method. Whereas, using the standard method, the participants produced a 14.2% on average of the MSD. Thus, the Espresso method out-performed the standard method. Unsurprisingly, all five participants perferred the Espresso method to the standard method [14].

While the results are from a preliminary study, they show that the Espresso method has potential to out-perform the standard numeric entry method. We decided to conduct a more rigorous study not only on the Espresso method, but both the Espresso and the Cappuccino methods.

5 DigiTaps

Since the preliminary evaluation of DigiTaps shows promising results, a more rigorous study is needed to evaluate the gestures more thoroughly. We adopted the user study model presented in [7] to conduct our user study. We developed DigiTaps Game, a platform for conducting number entry method user studies in the large.

The Espresso and the Cappuccino methods have been evaluated in a lab study as presented in [2]. However, the lab studies does not reflect the actual use of the gestures, so we want to design user studies that simulate real world use of the gestures. We assume that number entry speed is the prominent factor for the *real world use*, so we decide to incentivize users to enter the numbers as fast as possible by designing DigiTaps as a game.

In addition to simulating real world use, we gather data from the players by keeping track of the gestures they perform. We save the data in our database to do further analysis on the data. To gather as much data as possible, DigiTaps is distribute the game via the Apple App Store.

5.1 Flow of DigiTaps

The first screen shown to the player after launching DigiTaps is the main menu of the game as (see Figure 5.1). This screen consists of three choices for the player to choose (1) Tutorial, (2) Start Game, (3) Leaderboard. Selecting different button leads to different modes of DigiTaps and leads to different screens that the player visits.

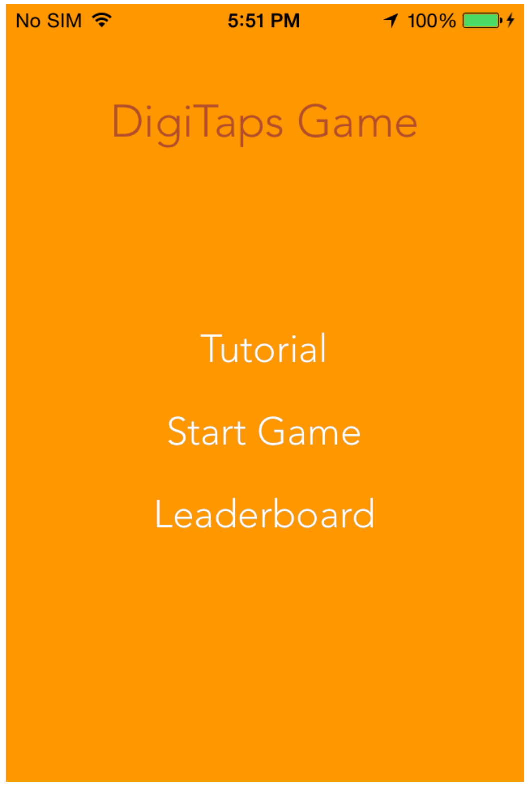

Figure 5.1: Main menu and the first screen of DigiTaps

5.1.1 Tutorial

We provide some tutorials for the players before jumping into the game. Each of the tutorial consists of two main parts the description of the tutorial and the practice mode for the tutorial. There are three tutorials total. The first tutorial is the overall description of the game. It describes the main gestures such as long-press to submit the number. The two other tutorials describe the DigiTaps method. There is a table at the bottom of each DigiTaps method tutorial. The practice screen is an empty screen for the player the play around with the gestures they learned. The screen gives the user feedback on the action the player just perform. For example, if the player tap the screen with three fingers, the screen shows that it is a three-finger tap (see the rightmost state in figure 5.2).

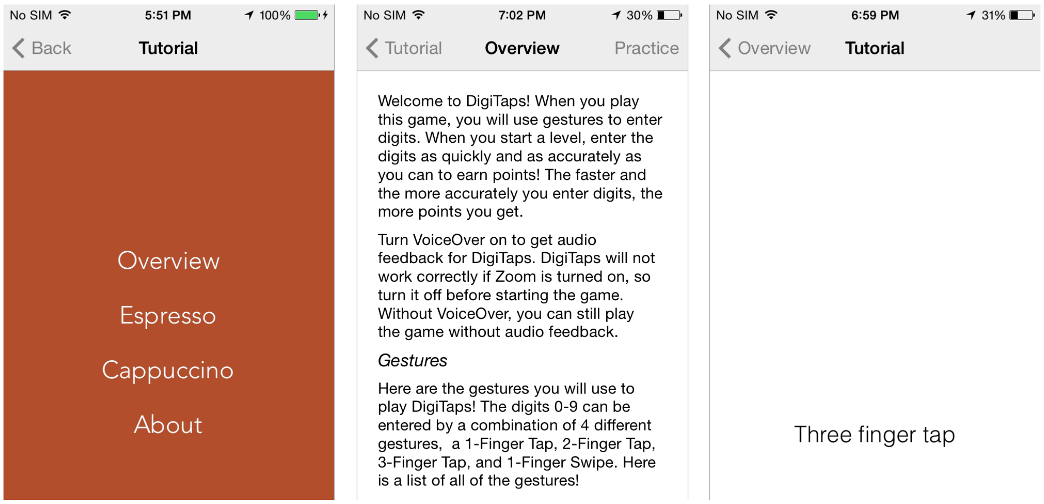

Figure 5.2: The state on the left shows the options the player can choose to learn about. The state in the middle is an example of a tutorial page where descriptions are provided. The state on the right is an example of a practice screen.

5.1.2 Game Play

(1) a DigiTaps method selection screen, (2) a level selection screen, (3) a game play screen, and (4) a summary screen. At the method selection screen, DigiTaps lets the player choose either the Espresso method or the Cappuccino method, in which the player is going to use that method to play the game. After the method has been chosen, the player can choose a level to start playing the game, indicated by the red arrow in figure 5.3 from the top row to the leftmost state on the second row.

At this point, the game play screen slides in and show the first number to the player (see the leftmost state of the second row in figure 5.3). The player performs the gesture to enter the number shown on the screen. To submit the number, the player has to tap and hold that finger on screen with one finger until the ring or the buzz sound occurs. This means that the number has been submitted and DigiTaps advances to the next number. The player has to enter 10 numbers per level and the number of digits per number varies based on the level. The numbers starts with 3 digits per number at level 1 and it increases by one digit at each level. At the last level, level 5, each number is 7 digits long.

After entering all the numbers required, the summary screen appears. The summary screens provides the player's performance on that level. The information includes the points earned, the accuracy and the average time per digit. On this screen, the player has the option to advance to the next level, replay the same level or quit to the start menu as (see the last state of figure 5.3).

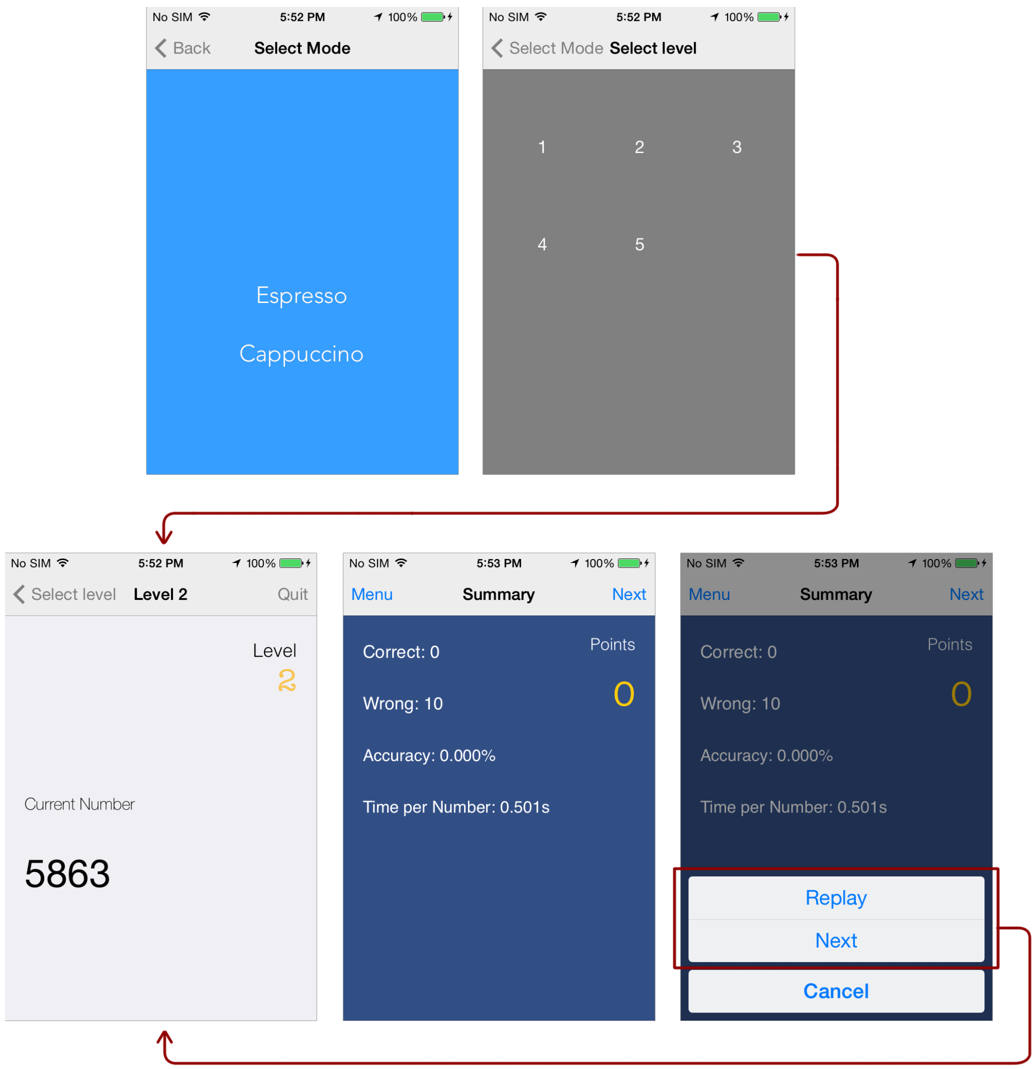

Figure 5.3: Shows the state diagram of DigiTaps gameplay mode.

5.1.3 Leaderboard

To provide the players with competition, we include Apple's leaderboard into DigiTaps. The leaderboard ranks the player's score with other players around the world who uses Apple's Game Center. Players are being ranked with respect to their scores they achieved in the game (see figure 5.4).

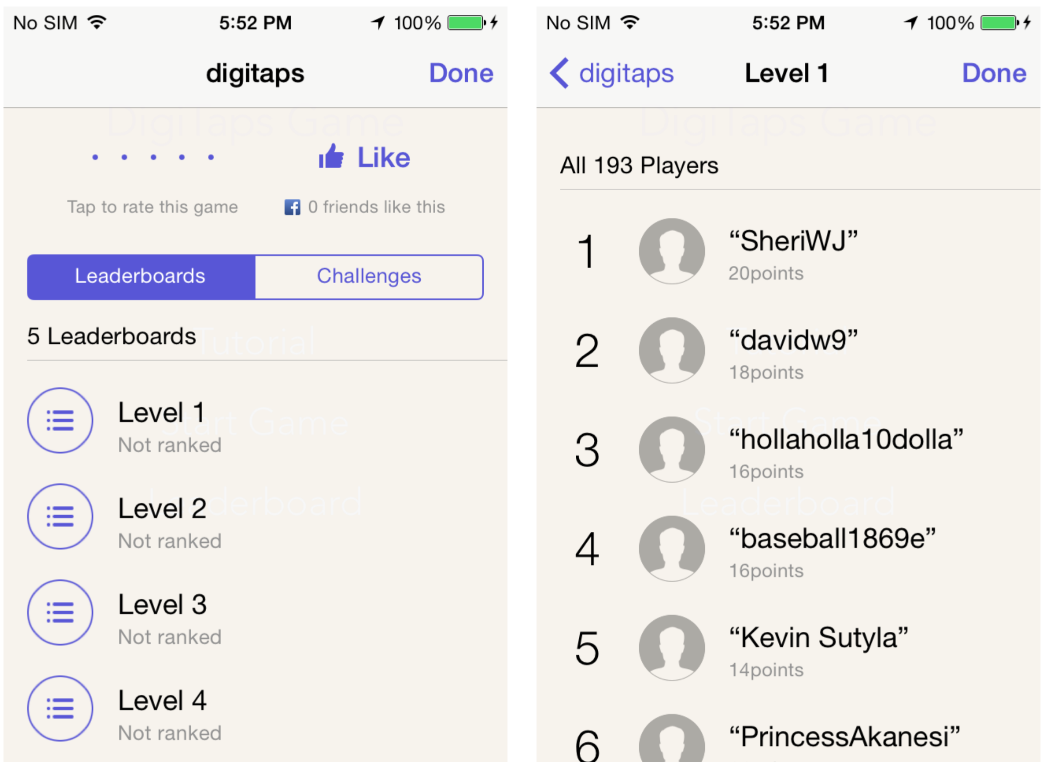

Figure 5.4: The state on the left shows the leaderboard of each level. The state on the right shows the rankings of the first six places for level 1 with their respective points.

5.2 Interaction Gestures

In addition to the DigiTaps gestures described in the DigiTaps code, DigiTaps has two other gestures for players to interact with the game. The first gesture is two-finger swipe in any direction. This gesture does two main action in the game. First, it acts as the backspace gesture. We have to use two-finger instead of one because the DigiTaps gestures have already used one-finger swipe in any direction. It also acts as the repeat gesture. When the player starts entering numbers, the number initially shown on the screen disappears. To see the given number again, the player has to delete all the digits inputted using the backspace gesture and do an additional backspace gesture to make the given number appear.

The second special gesture is the long-press gesture. This gesture is used for advancing to the next number in the level or if there is no more number on that level, this gesture takes the player to the summary screen. When the player finishes inputting the given number, the player holds one-finger down on the screen until there is a bell ring or a buzz sound from DigiTaps. The sound indicates that the game has advanced to either the next number or the summary screen. The bell sound indicates a correct attempt and the buzz sound indicates a failed attempt.

5.3 Data Collection

In order to conduct the user study, we collect two different information from the players. We collect the players or the study participant's demographics. This includes the player's age, gender, experience with accessibility in iOS. Each player is uniquely identify with an identification number. However, we cannot trace back to who the actual player was. This identifier is used to match the player to the other information of this player that collected.

In addition to the demographics, we collect the touch events that the player performs in the game. Every touch event is recorded. This includes touch-down, touch-move, and touch-up events. In each of these touch events, we record the number of fingers on that touch event, the location of the touches and many other information. Furthermore, we recorded the state of the game such as when the game starts, the level finishes, the number is inputted, and some other game states.

Using the information collected, we can gain some insight on how the players behave in the game. More important, we can evaluate the gestures with the data gathered to see which of the gestures performs better.

6 Preliminary Data Evaluation

As of writing this paper, there are 654 instances of DigiTaps installed. In addition, we collected 129,906 events which contains 96,892 gesture events. This means that each player performs $\frac{96892}{654} = 148.15$ taps on average. Since there are 3 numbers per level[1], the average number of taps per level is $1.8 \times 3 \times 3$, which can be broken down to each digit requires 1.8 taps[2], there are 3 digits per number and there 3 numbers per level.

6.1 Demographics

The following are the result of the 6 demographics questions we asked.

1. **How old are you?**
   47.71% of the all the players are under the age of 25 (see figure 6.1) with players aged between 18 and 25 years old is the largest population in this group. Since DigiTaps is a game, this result is not surprising. People at a younger age may want to try out the game more than the eldery.

2. **What is your gender?**
   The gender are split almost evenly between male and female. Male accounts for 48.01% and female accounts for 51.99% of all the DigiTaps players.

3. **How would you identify yourself?**
   We advertised DigiTaps through several blind organizations' mailing lists. It is not surprising that 48.5% of the players indentified themselves as blind (see figure 6.2). Sighted players contributes 37.9% of all the players.

4. **Do you use any accessibility tool on a daily basis?**
   Since 48.5% of the players identified themselves as blind, it is not surprising that the majority of the players uses VoiceOver on a daily basis. However, there are 61.3% players who use VoiceOver regularly, which is more than 10% greater than the number of blind players (see figure 6.3. One possible explanation is 13.6% of the players identified themselves

[1] DigiTaps has 3 numbers per level until version 1.1.
[2] Here we assume that we use Cappuccino method.

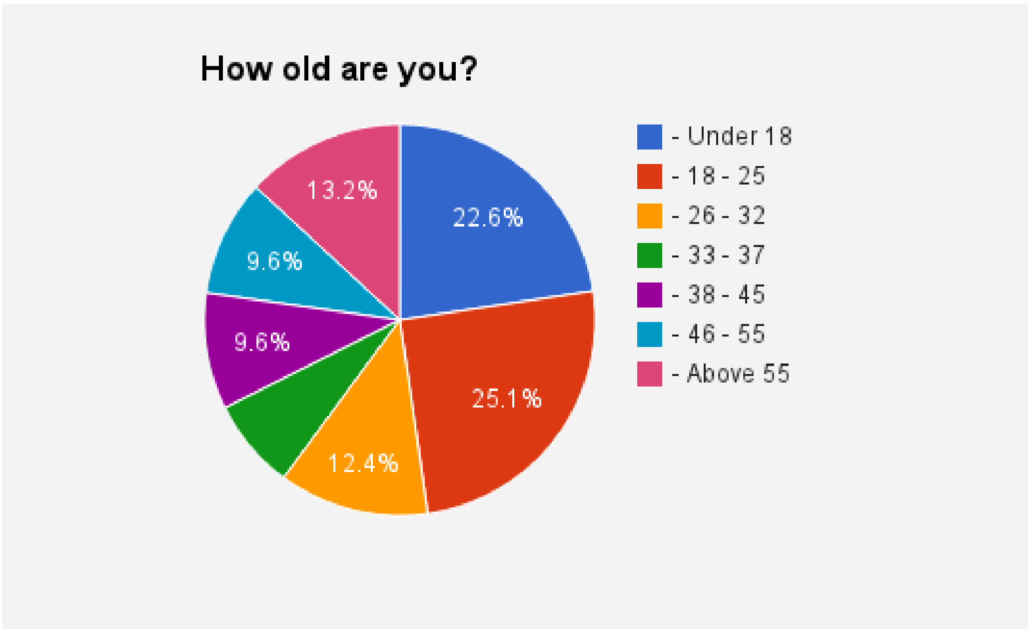

Figure 6.1: This figure shows the breakdown of the age among all the players.

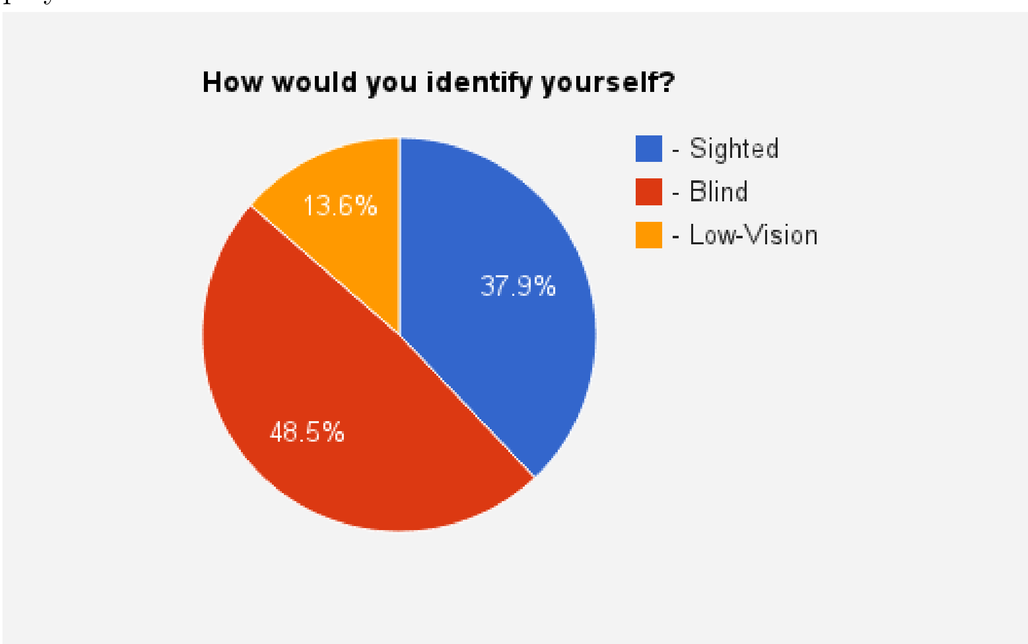

Figure 6.2: This figure shows the breakdown of players' identification among all the players.

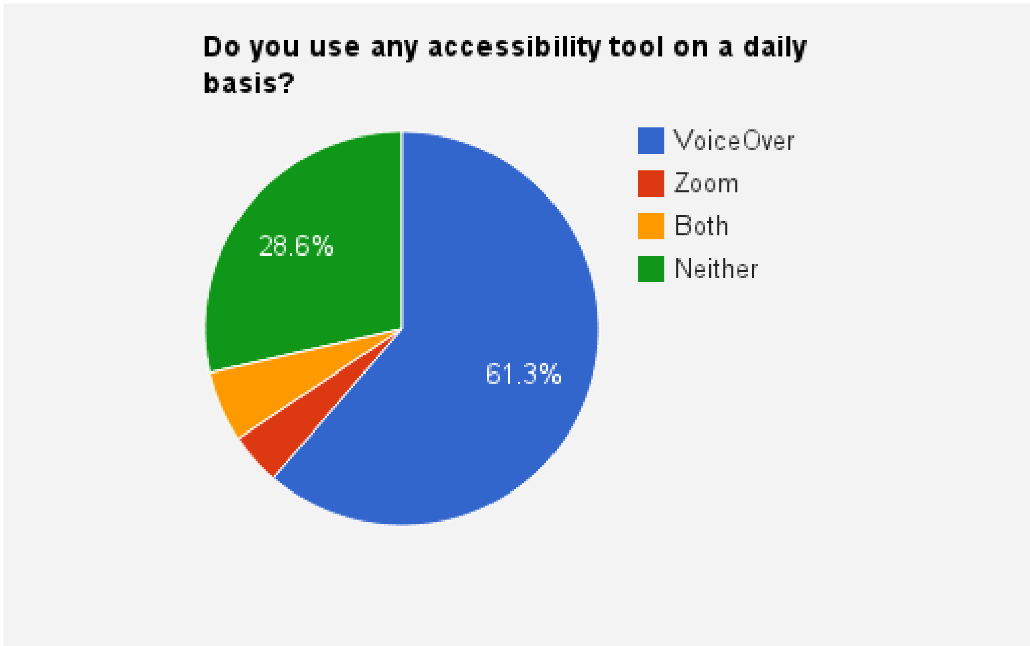

Figure 6.3: This figure shows the breakdown of the accessibility tool usage among all the players.

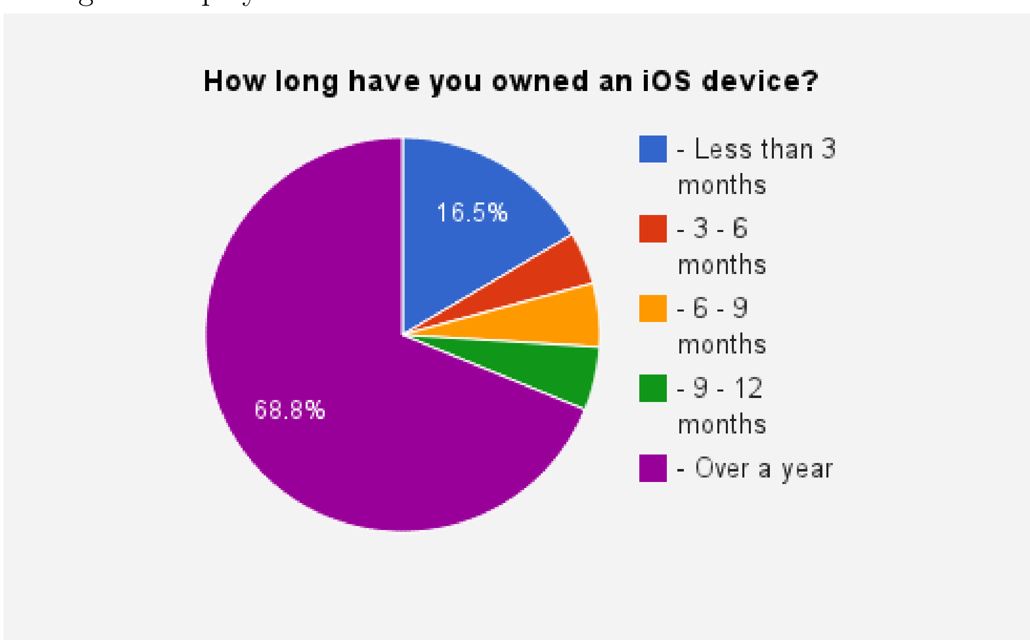

Figure 6.4: This figure shows the breakdown of how long the player possessed his device among all the players.

as having low-vision. Low-vision and blind players combined are 62.1% of all the players.

5. **How long have you owned an iOS device?**
   Since we only have DigiTaps on iOS, we asked how long that they owned the device, so we can measure their fluency with the platform. The majority, 68.8%, of the players own their device for over a year (see figure 6.4). The second largest population is the players that owns the phone less than 3 months.

6.2 Touch Events

Although we have 654 installations of DigiTaps, the number of touch events are too low to do any significant data analysis. There are only 96,892 touch events this is 148.15 touches per user on average. Assuming we are on level 1, which contains 3 3-digit numbers, the whole level requires $2.1 \times 3 \times 3 = 18.9$ taps[3]. Thus, only 7 games were played for each person, which is too low to do in-depth evaluation of the gestures.

Furthermore, we took a closer look into the data using the longitudinal lab study of DigiTaps as the baseline. The study consists of 6 session where each session is divided into 4 conditions. In each condition, the participant is asked to enter 15 random 4-digit numbers [2]. This gives us $4 \times 15 \times 4 \times 6 = 1,440$ digits total. Assuming we use the Espresso method, the player has to perform $2.1 \times 1,440 = 3,024$ taps on average. Hence, we only consider players that performed more than 3,024 gestures. Looking closer into the data, there are only 6 players that has performed the gestures more than 3,024 times. 6 players is too few to do any in-depth analysis on the data.

7 Conclusion and Future Work

We introduced DigiTaps code, a prefix-free code that uses minimal voice feedback. DigiTaps requires at most 2.1 taps per digit and uses only simple gestures in its codes such as swipe and tap. DigiTaps shows promising results in the preliminary evaluation with the entry rate of 1.99 seconds per digit ($SD = 1.25$) using the Espresso method whereas using VoiceOver achieves only 2.77 seconds per digit ($SD = 1.24$).

[3]Assuming that we use Espresso method

We then developed DigiTaps game as a platform for conducting user study in the large for the DigiTaps gestures. DigiTaps is a simple number entry game. It gives a number to the player and the player uses one of the DigiTaps gestures to enter the number. DigiTaps is distributed through the Apple App Store. We collected information on how the gestures were performed and other metadata such as the number given to the player in the game. At the time of writing this paper, there are 654 installations and 129,906 events and some preliminary evaluation has been done on the data.

Even though DigiTaps code and the DigiTaps game has been developed and studied to some extent, there are several aspects of the project that can be improved.

1. **Advertise and recruit recurring players:**

   As presented in the preliminary data analysis section, the data we gathered is still too little to do any in-depth evaluation on the gestures. We need to recruit more players and keep them playing the game.

2. **Implement DigiTaps codes in a real-world application:**

   DigiTaps code shows potential to be fast and works well in noisy settings. We can incorporate DigiTaps code to any number entering application such as a calculator, personal identification number (PIN) entry and making phone calls.

3. **Clean up DigiTaps framework and open source it:**

   DigiTaps was designed to be a platform to do user study in the large from the beginning. Users can use implement their own game and replace the DigiTaps game engine with the implemented game engine. In addition, users can define their own gestures to be detected in DigiTaps. However, at the current state, the framework still needs some cleaning up to make the framework easy to use as much as possible.

8 Acknowledgments

I would like to thank Professor Richard Ladner, Shiri Azenkot and everyone in the MobileAccessibility research group for the guidance and help. I also thank the Mary Gates Research Endowment for the Mary Gates Research Scholarship.

References

[1] Apple Inc. iPhone Accessibility. `http://www.apple.com/accessibility/iphone/vision.html`. Accessed on May 10, 2014.

[2] S. Azenkot, C. L. Bennett, and R. E. Ladner. Digitaps: Eyes-free number entry on touchscreens with minimal audio feedback. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 85–90, New York, NY, USA, 2013. ACM.

[3] S. Azenkot, S. Prasain, A. Borning, E. Fortuna, R. E. Ladner, and J. O. Wobbrock. Enhancing independence and safety for blind and deaf-blind public transit riders. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 3247–3256, New York, NY, USA, 2011. ACM.

[4] S. Azenkot, J. O. Wobbrock, S. Prasain, and R. E. Ladner. Input finger detection for nonvisual touch screen text entry in perkinput. In *Proceedings of Graphics Interface 2012*, GI '12, pages 121–129, Toronto, Ont., Canada, Canada, 2012. Canadian Information Processing Society.

[5] D. Goldberg and C. Richardson. Touch-typing with a stylus. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, pages 80–87, New York, NY, USA, 1993. ACM.

[6] Google. Android Accessibility. `http://developer.android.com/design/patterns/accessibility.html`. Accessed on May 10, 2014.

[7] N. Henze, E. Rukzio, and S. Boll. 100,000,000 taps: Analysis and improvement of touch performance in the large. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, MobileHCI '11, pages 133–142, New York, NY, USA, 2011. ACM.

[8] N. Henze, E. Rukzio, and S. Boll. 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, pages 2659–2668, New York, NY, USA, 2012. ACM.

[9] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101, September 1952.

[10] S. K. Kane, J. P. Bigham, and J. O. Wobbrock. Slide rule: Making mobile touch screens accessible to blind people using multi-touch interaction techniques. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility*, Assets '08, pages 73–80, New York, NY, USA, 2008. ACM.

[11] S. K. Kane, J. O. Wobbrock, and R. E. Ladner. Usable gestures for blind people: Understanding preference and performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 413–422, New York, NY, USA, 2011. ACM.

[12] I. S. MacKenzie, R. W. Soukoreff, and J. Helga. 1 thumb, 4 buttons, 20 words per minute: Design and evaluation of h4-writer. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 471–480, New York, NY, USA, 2011. ACM.

[13] J. a. Oliveira, T. Guerreiro, H. Nicolau, J. Jorge, and D. Gonçalves. Blind people and mobile touch-based text-entry: Acknowledging the need for different flavors. In *The Proceedings of the 13th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '11, pages 179–186, New York, NY, USA, 2011. ACM.

[14] V. Ruamviboonsuk, S. Azenkot, and R. E. Ladner. Tapulator: A non-visual calculator using natural prefix-free codes. In *Proceedings of the 14th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '12, pages 221–222, New York, NY, USA, 2012. ACM.

[15] C. Southern, J. Clawson, B. Frey, G. Abowd, and M. Romero. An evaluation of brailletouch: Mobile touchscreen text entry for the visually impaired. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services*, MobileHCI '12, pages 317–326, New York, NY, USA, 2012. ACM.