7-2013

Portable Game Based Instruction of American Sign Language

Christyna Wilson
Clemson University, cnwilso@g.clemson.edu

PORTABLE GAME BASED INSTRUCTION OF AMERICAN SIGN LANGUAGE

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Masters
Digital Production Arts

by
Christyna Nichole Wilson
August 2013

Accepted by:
Dr. Brian A. Malloy, Committee Chair
Dr. Joshua Levine
Mr. Anthony M. Penna

Abstract

In this thesis, we address the effectiveness of portable gaming devices as a vehicle for learning American Sign Language. We begin by creating a multi-level game using a standard programming language, and use the game to illustrate various signed words. We translate the game to mobile devices using Kivy, a crossplatform framework. This allows the game to be compatible with multiple mobile devices and platforms. A secondary inspiration for creating a game as opposed to standard memorization techniques is to stimulate a quicker learning response as the user enjoys the sign learning process. In this thesis we provide a mobile avenue for learning American Sign Language and create a fun and stimulating learning process to strengthen the immersion of the user.

Dedication

Many of life’s failures are people who did not realize how close they were to success when they gave up.

Thomas Edison

Acknowledgments

I would like to acknowledge the following people for their help with my thesis: my advisor, Dr. Brian Malloy for his patience and guidance; my boyfriend, Austin Hackert for his support when I needed him most; my parents, Clarence and Deborah Wilson for building up my confidence, always believing in me, and supporting me when others did not; my family for cheering me on through the years; and lastly, my friend Arthur Webb for pushing me to continue when I thought all was lost. I love you all. Thank you.

Table of Contents

Table of Contents (Continued)

Page

List of Figures

Figure Page

Chapter 1

Introduction and Motivation

American Sign Language (ASL) is a language using hand motions, gestures, and symbols to convey the English language through visual communication to those who are deaf or wish to communicate with someone who is deaf. It is important for deaf children to have access to ASL in order for them to have a form of communication. Programs teaching ASL should be interactive since ASL itself is a visual language. Although some people with hearing disabilities can read lips, this process is tiring and does not always provide the expected results. ASL can be learned by anyone who may interact with a deaf person and is necessary in many communities.

Games as a source of learning are affective by creating a positive and relaxed environment that challenges the user to not only memorize the rules and stratedgy but also apply what they have learned in order to receive a reward. By placing content in game format, the user is able to concentrate on the fun and puzzling aspects of learning, all the while succeeding in retaining content. This type of learning can provide a more immersive process allowing the user to apply information again and again. In children, learning becomes more affective when they have a relaxed brain. Presenting a learning experience in a game allows the user to learn in exchange for some incentive. This keeps the player interested and masks some of the less exciting aspects of learning.

Portable game technology has become increasingly more popular over the last few years. This is largly due to the growing number of smartphones and other mobile devices available to children and adults alike. As technology grows, people become more dependent on having some sort of mobile device on them at all times. For families with children, this means providing entertainment while on the go as well as stimulating development disguised as games. Games like Candy Crush, Angry Birds, and Temple Run are fun games that provide the player with unseen learning experiences. Passing a level may mean earning points or watching a bird fly across the screen to knock a pig off a pillar. These types of visual stimuli are hiding the strategies and calculations the brain must make in order to do well in the game. Another positive to mobile gaming is the ability to memorize the tools and tricks to mastering the game. When the player sees that the blue bird will split into 3 birds going in the same general direction, he/she is not likely to forget that fact the next time a blue bird is in their sling shot. Likewise, the blue bird adds another level to the strategy formed in earlier levels. Now the player has to project not only where the blue bird is going to fly but also the projected path of each of the birds that split from the original path.

The purpose of this thesis is to combine an ASL learning experience with the growing mobile gaming industry. Many learning tools for ASL are solely based on tedious memorization of words and definitions matched with the corresponding sign. These examples are also available through applications on mobile devices. Although they are available, they do not have the same incentive and retaining qualities that a game environment provides.

There are also many resources that provide a word with a sign beside it. Even though these dictionaries of signs often have videos or animated gifs of the sign, it becomes more memorization without retaining the information. By simply assigning points to matching a sign with a word or letter, the entire dynamic of the learning experience changes. To address the issue of making the game available through various mobile platforms, a crossplatform framework will be utilized. The game previously attuned to the computer will be able to extend onto a platform that is available to many mobile technologies. This will allow the game to be accessible to a wide variety of users.

In the next chapter, we provide background information and definitions about terms and concepts used in this thesis. In Chapter 3, previous research similar to this will be discussed. In Chapter 4, we describe the process of creating the game for the computer. In Chapter 5, we explain the process of re-creating the game using the crossplatform framework. In Chapter 6, we describe the results of the computer to mobile device transformation. In Chapter 7 we discuss the results and describe problems with the transfer and summarize some approaches to solving the problems in future research. Finally, in Chapter 8 we draw some conclusions.

Chapter 2

Background

In this section, we review terms and concepts that we use in this thesis. We begin by describing the logistics of sign language and the technical qualities of using sign language. In the second section, we describe the programming language used to code the game, and we conclude the section by describing the language we used as a crossplatform framework.

2.1 Logistics of American Sign Language

American Sign Language (ASL) solves a very critical problem that is a part of our society. ASL forces the speaker to use visual stimuli as opposed to audio stimuli. For our purposes, we use the word symbol for an individual letter and sign for a sign with motion that generally means a word. Signs are very precise and must be performed correctly or else the word may end up with a different meaning. For example, the sign for soup and spoon only differ by the speed in which they are signed or the proximity of the two hands forming the sign. Just like the hearing version of English, sign language has different dialects based on the geographical location. Langue des signes francaises (LSF) or French sign and American Sign language are different. They may have the same signs that mean the same thing, or those signs may have completely seperate meanings. In this thesis, we will use solely ASL.

2.2 Creating Games with Python

Python is a programming language that allows the user to perform rapid prototyping and scripting while keeping maintenance at a minimum. It is a general-purpose, dynamically typed programming language widely used for application domains. It provides clear syntax, object orientation, hierarchial capabilities, exception-based error handling, high level dynamic data types, and the ability to extend using C or C++ among many other qualities. Python streamlines blocks of code making them into single lines or small functions. It has fast compliation speeds and runs code fast enough for most applications. Finally, Python works great with other programs as it is extremely flexible and accepts modules and extentions.

2.2.1 Creating Games with Pygame

Another key building block of this thesis is Pygame. Pygame was developed as a wrapper class for the Simple DirectMedia Layer (SDL) library. It is used in conjunction with Python to allow programmers to develop as a competitive real-time computer game platform without being slowed down with low-level lines of code such as C. It is highly portable within the computer world, and is beginning to be considered for mobile technology. It combines the ease of use and the workability of Python with a smooth user interface including things like sprites and animations.

2.2.2 Creating Games with Kivy

Kivy is an open source Python library used for the rapid development of applications. These applications are cross platform, business friendly, and GPU accelerated. It is built to allow things such as finger touch apps that run quickly and efficiently on any platform. Although Kivy utilizes Python like Pygame, the Kivy language is starkly different to anything we have seen before. It completely rearranges Python and adds its own finishing touches to create smooth apps that are interactive and quick. Since its creation, Kivy is constantly being tested to make sure the graphics stay top of the line. The Kivy language can create a game as simple as pong to something as intriguing as a color wheel chooser. This allows the user to create eye catching games that have function and form.

Chapter 3

Related Work

Alex Games investigated language and literacy in a novel game-based learning environment called Gamestar Mechanic [4]. His goals were to investigate how Gamestar Mechanic taught children to appropriate aspects of game deisgn within their language and literacy skills. He aimed to do so via instruction based on key principles of game design. Gamestar Mechanic was designed to introduce children to the "language of games" in order to enhance their natural ability to form ideas in a sophisticated manner. These goals are similar to ours in that the "language of games" is utilized to create a learning experience. In his research he found that Gamestar Mechanic, as well as computer game design, allows children to be more literate in accordance to the 21st century standards.

The Educational Gaming Commons (ECG), Education Technology Services, and Dr. Mary Shoemaker of Penn State University, developed a game called ChemBlaster [2]. This game is designed to assist college students gain a better understanding of Chemistry 101. They recognize that memorizing all of the terminology for needed to learn chemistry is difficult, and they combat this by creating a game that is both fun and easy to use. The ECG is dedicated to discover games and use them as tools in a learning environment.

With the game-based learning industry steadily growing, noteable organizations such as the Nobel Prize Organization are exploring the topic of educational games [7]. They offer a wide variety of educational games that allow the player to have fun while learning, ranging anywhere from a Blood Typing Game to Invar and Steel Alloys. This relates to our study by showing how important game-based education is becoming to our society. Similarly, the Federation of American Scientists has begun, as their motto states, "Harnessing the power of video games for learning" [6]. They also offer several options for educational learning.

Rankin et al. investigated the effectiveness of creating a serious game with an enjoyable experience parallel to development and learning [9]. They stress the impact that serious games have on the user, as well as the importance of creating serious games with strategic content that presses the user without going too far. They also utilize the strength of a Massive Multiplayer Online role play game as a tool to learn a secondary language. This is synonymous with our game based instruction of ASL. The ASL Game falls under the category of a serious game, allowing it to be both fun and useful.

Dr. Richard Blunt investigaed the effectiveness of game based learning with three studies [1]. The first study encompased a training environment for business students, the second study included economics students, and the third study was performed on management students. Each study included a group of subjects in a learning environment with a portion of those subjects randomly choosing to be administered the game. Then the subjects were compared to the subjects who were not taught with the game. Through all three studies, the subjects who learned with the game produced higher scores on average, and there were fewer lower grades in the game tested sample. This is indicative that game based learning does allow the user to learn and retain information better than standard methods of learning.

Ebner and Holzinger were able to successfully implement game based learning in higher education [5]. They tested the game during a Master's leve lecture involving 121 students. These students were given a pre-test and a post-test. They found that the lower end of the game based results matched the higher end grades with traditional teaching methods. For this particular study, the game based results were exponentially higher than traditional methods. They condone using game playing in education to reach increased learning results. This example shows that even people in higher education can benefit from game based learning.

Foreman explored the world of game based learning by discussing six major topics with five prominant researchers in the field of game based learning [3]. He discussed the downfalls of traditional methods, the strength of simulations, and the relevance of game based learning in our community among other things. Throughout the article, it is apparent that these prominant researchers have found information supporting the fact that game based learning is important to our society. They collectively believe that adding games to teaching methods will increase the level of learning and absorption of information in students.

Similarly, Squire compiled the studies of three game based learning companies [8]. The results concluded that game based learning provides positive results in many situations. He suggests that the new direction of game based learning creates compelling learning context, immerses users in complex information, allows for challenges that expand user knowledge and open the door for further understanding, and encompass the instructional education. By creating a game that inhabits these and more qualities, the user can enjoy a more positive learning experience. This is similar to the goals of this thesis. To provide a challenging yet fun learning environment to allow the user to better understand and retain information.

Chapter 4

Game Design

In this chapter we explain our methodology in designing the game and preparing it for the recreation with Kivy. The process has several steps to ensure a unique and interesting game. Step one is choosing which signs and symbols that would best suit the ASL Game. Our use of letters from ASL as well as signs from ASL is what drives the game and creates an intrguing user interface. The next step is the design and implementation of the game elements. The third step is to create each level of the game, and then create the text and button elements for the fourth step. Finally, we translate the Python/Pygame implementation into the Kivy language.

At Clemson University, the Digital Production Arts (DPA) lab provides several Wacom tablets that allow the user to create art by drawing on the screen. These machines provide a large surface that interacts with the stylus to allow images (among other things) to be created as if a pen and paper had been used. We used resources online to determine the correct way to portray each sign or symbol. These tools help us to create a game that immerses the user in the world of ASL.

4.1 Choosing Signs and Symbols

Step 1. The layout of the game is a grid pattern, similar to some popular mobile game such as Words with Friends by Zynga. The game has two different levels for beginner or intermediate ASL learners. The first level is kept monosyllabic to appeal to beginner users. Each word is made up of three letters that are some of the more common letters in the English language, such as ’a’ and ’s’. This allows the user to learn what each symbol means without adding too many syllables at once. The second level has a somewhat smaller grid in order to showcase larger signs. The larger space allows for signs that are animated. The next stages of the game design are noticeably more complex by trying to interpret the design style using Python along with Pygame.

4.2 Design and Implementation of Game Elements

Step 2. All elements in the ASL Game were hand designed by the author using the Wacom tablets available to the DPA students. This includes each frame of the level two signs. The remaining elements (other than the text) were created using GIMP editor. Each of the symbols in level one also have a secondary or alternate image to be displayed while the symbols were clicked. These symbols were also created specifically for the ASL Game. When the user clicks on a letter to see if it is the correct one in the word, the corresponding English letter is displayed.

4.3 Implementation of ASL Game in Python and Pygame

The next section explains in detail how Python and Pygame assist in creating a user-friendly gaming apparatus. The first step is creating the grid. Although creating a grid is simple in Python, this is where Pygame really shines for animated elements in a grid. Pygame allows the creator to use sprites to continuously update the grid on screen.

4.3.1 Implementation of Level One

Step 3. Each of the sixteen symbols in the level one grid is a sprite that holds its own position and English letter tag. For example, the symbol for the letter 'C' contains the symbol '1c1.bmp' file, 'c' as a string value, the position on the screen, and a few boolean values. The boolean values are used as flags to determine if the sprite is currently selected. If it is selected, it is highlighted by a purple rectangle.

To allow for easier understanding of each symbol in level one, the correlating English letter is passed as an image inside the sprite. During the event loop, the boolean value to test whether it was flipped or not is accessed. Each sprite is then added to a sprite group, also known as a non-ordered list of sprites. Next the event loop for level one is created. This event loop determines if a letter was clicked. If that letter has been clicked, the boolean values for drawing the rectangle and flipping to show the alternate image are set to True. Pygame has built in sprite collision detection. This in collaboration with Python's mouse position tracking allows for the collision to be detected on each symbol of the grid, affectively turning it into a button.

When each sprite of the group is updated and drawn, the code checks the booleans. If they are True, it then highlights the symbol by drawing a rectangle around the sprite and inserts the alternate image in its place. The next part of the event loop keeps a running string of each of the buttons as they are pressed. The final step requires testing if the string created by the users clicks matches the word being sought by the ASL Game.

All possible words for the layout of the level one grid are predetermined. Those words are then stored in a list of words. Once three letters are selected, the string created by adding each letter together runs through a function to test it against the current word the ASL Game asks for from the list of words possible. For example, if the user is tasked to find the word 'cat', they click on the symbol for 'c', 'a', then 't'. The game then tests to make sure it said 'cat' and returns that the user is correct. Accordingly, if the user chose the incorrect letter combination, the game returns that the user was incorrect.

4.3.2 Implementation of Level Two

Step 4. Level two is very similar to level one, however, there are a few key differences. The sprites need to be able to accept frames of animation and continue to animate throughout the selection process (from here on known as a multisprite). To do this, the sprites hold an image list. Depending on which point in time we are currently updating, the multisprite captures the next frame on the image which contains four frames.

For this level, each multisprite holds the image file, the corresponding English word, and its position on the newly sized grid. Like the sprite, it also has a highlight boolean. When the multisprite is selected, the boolean tells the game to highlight the multisprite with a purple rectangle on the next update and draw. Since the user chooses only one multisprite to select the current word the ASL game is searching for, there is not enough time inbetween selections to flip the multisprite to display its corresponding word. Shortly after choosing the multisprite, it unhighlights to show that the board is ready for the next selection. To determine if the user clicked the correct multisprite, another list of words is created specifically for level two, the ones to be animated via multisprites. If a multipsrite is selected, the game then sends the corresponding English word through a function to be tested and returns whether the user was correct or incorrect.

4.3.3 Adding Text and Button Elements to Pygame

Step 5. This step allows the user to better understand the game. First, the word to find is printed just above the grid. Second, the scoring system is implemented. Third, a hidden answer check is incoporated with 'CORRECT' or 'Incorrect'. Finally, a menu of buttons is added to allow the user to quickly move between levels, reset the current level, or quit the game all together. Since the text elements were also being updated regularly, they are also implemented as sprites. Each of the text sprites are either visable or hidden from the first updating and drawing of this event loop. When an answer is chosen, one of the hidden sprites displays until the next mouse click.

Each button element is created as a sprite similarly to the elements of each grid. In the same manner as before, the game checks to see if there has been a collision between the sprite and the current mouse position upon a click creating three new buttons.If the user clicks the 'Level 1' button, they either reset their score and start the same level over or they are switching to another level. As expected, the 'Quit' button quits the game.

In order to be able to switch between each level, the main event loop has a seperate event loop for each level. In essence, while the program is running, it checks to see if which level the ASL Game is currently, then runs the corresponding event loop. Now the user is able to switch between each level seemlessly.

Chapter 5

Evaluation

5.1 Kivy Interpretation

The five steps addressed in Chapter 4 are revisited to determine the best plan of action for translating Python/Pygame into Kivy. Kivy is very different Pygame, and transforming it into a portable crossplatform framework means coding each section to be as similar to the original game design as possible. By setting up the thesis in this manner, the Python and Kivy versions of the game ae going to be slightly different, but each affective in their own platform. The grid, the text, and the options are still incorporated even though they may not look identical.

Creating an interpretation of the ASL Game in Kivy requires interpretting the elements as well as the gameplay into Kivy. Kivy uses widgets, buttons, labels and other such elements as opposed to sprites. It also uses an attribute called canvas that is the main area for creating the app. First the grid is formed to replicate the grid in the oringinal version. After the grid is created, the secondary areas for text and menu buttons and widgets were inserted. To maintain a similar look to the oringinal game, multiple GridLayouts are created within the canvas. Each element is added to the layout as a button that can be pressed. The next portion of the translation is to add the images to the buttons in the grid. In Kivy, once the grid size is set, the grid will stay the same. The images were placed on the buttons with the alternate images being placed in the background.

Chapter 6

Results

In this section we provide results from creating a Python/Pygame application and how it is translated into Kivy. Although the layout and design of the two different versions are not identical, each game is best suited to its respective platform. The Python/Pygame version is straightforward and looks great on the screen where as the Kivy version can adjust itself to fit onto whichever platform it is ported into.

6.1 Results of Python/Pygame ASL Game

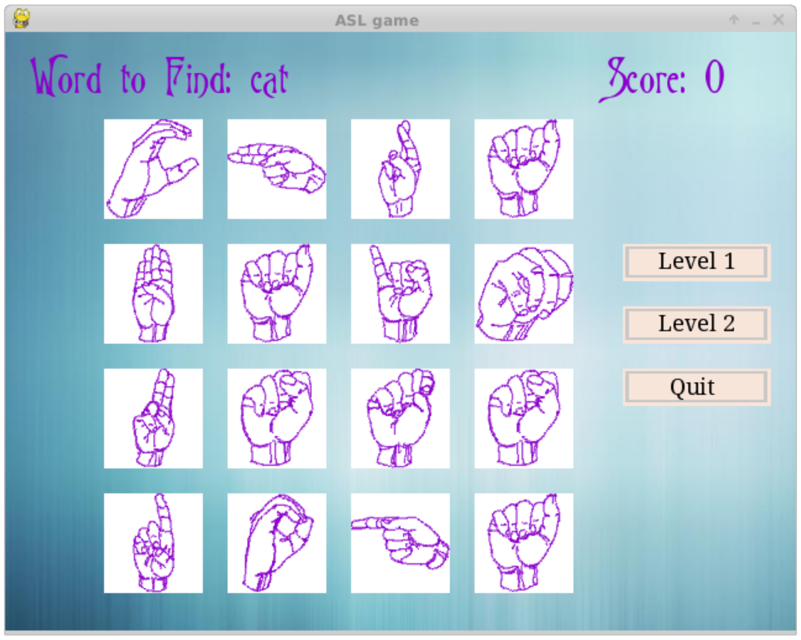


Figure 6.1: The introductory screen of the ASL Game in Pygame

.

The results for the incoporation of ASL in a game-based learning environment were as expected. The game interface is both easy to use and easy to remember. The image in Figure 6.1 shows the introductory screen of the Pygame version of the ASL Game.

6.1.1 Results of Level One

Level One is the beginner learning level for the ASL Game. In Figure 6.2, the first and second letters the mouse collides with are the 'C' button and the 'A' button. Those symbols flip to show that the letters chosen are in fact the letters 'C' and 'A'. Also shown here is the highlight produced from clicking one of the symbol buttons. The two images in Figure 6.3 show the final button choice is 'T'. Then it displays what the user sees right before the highlights disappear and the buttons flip back to their symbols. Also, the buttons on the far right of the screen are used to reset the current level, choose another level, or to quit the game. The Figure 6.4 shows the hidden text that only appears once a user has chosen in the top middle of the screen.

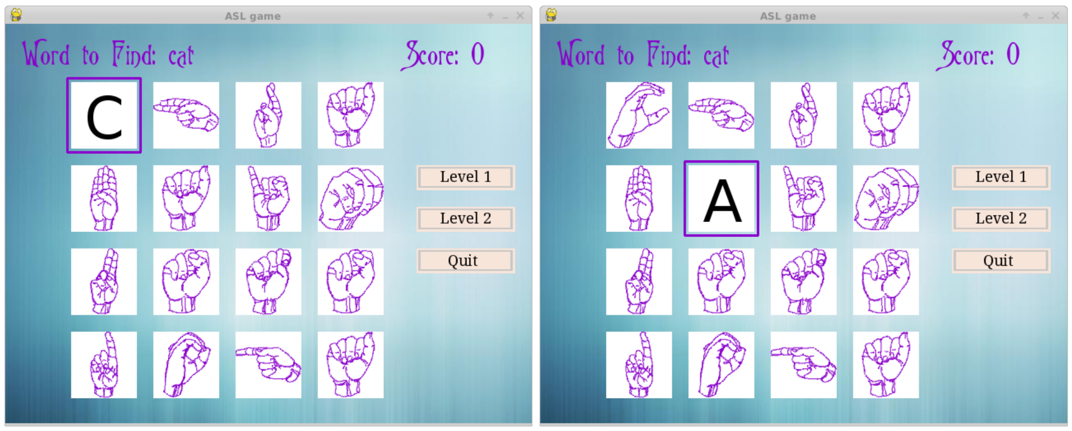


Figure 6.2: The first and second Button hits to find the word cat in Pygame

.

6.1.2 Results of Level Two

Level Two is the intermediate level. What is not easily portrayed in the Figure 6.5 is the fact that each one of the buttons has a constant animation. Each button holds a sign that is slowly animated in order to portray the sign correctly but in a manner that allows the user to understand each sign individually. As with level one, level two has a highlight when each button is selected, however there is no image flip as seen in Figure 6.6. The results from this level are as expected. Each sign is clear and consise.

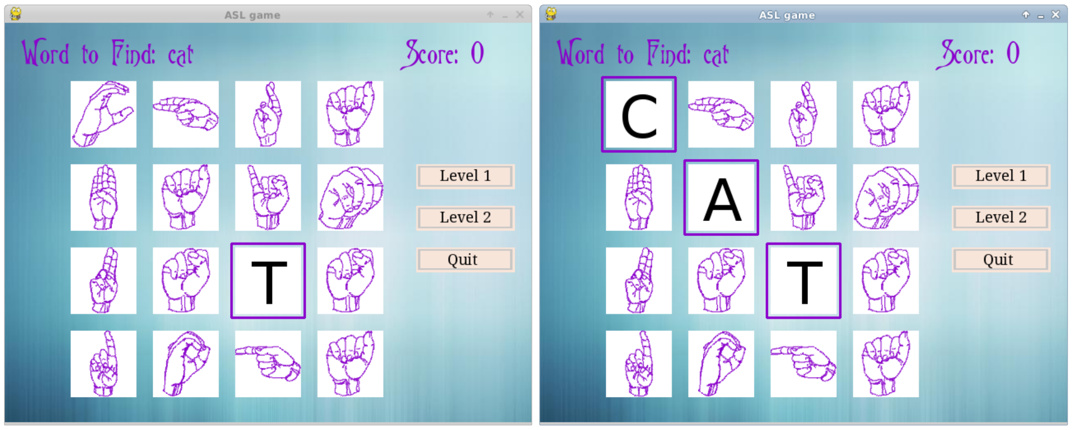


Figure 6.3: The first and second Button hits to find the word cat in Pygame

.

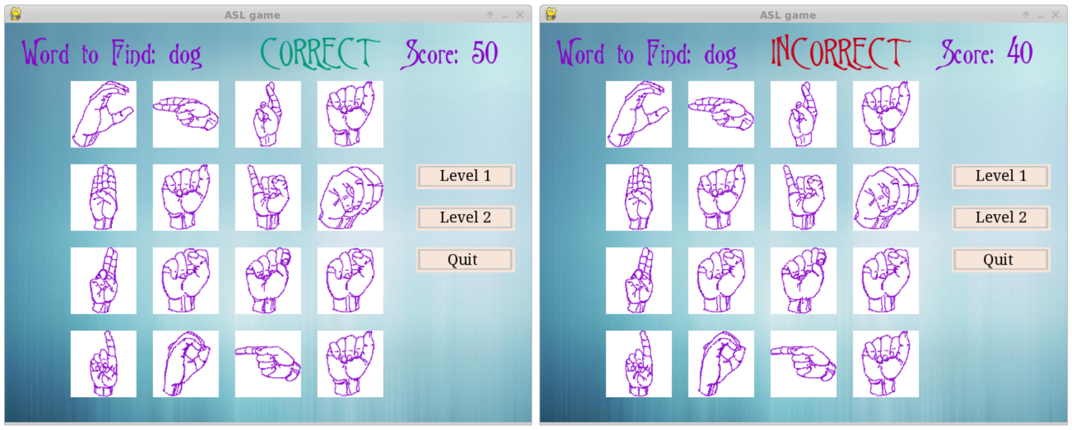


Figure 6.4: The hidden text appears in the top middle area of the screen once a word is found correct or incorrect

.

6.2 Results of Kivy

The results for the translation into Kivy was slightly less uniform than expected. Even though a grid was able to be created, each of the games were vastly different in appearance as seen in Figure 6.7. Another unexpected aspect to translating the Python/Pygame to Kivy is how different the coding languages are. Figure 6.8 shows just how different they are.

In Kivy, the grid is composed of buttons, however they are not neatly spaced or highlighted. Creating a grid in Kivy is a relatively simple task, however modifying the grid to appear different is very difficult. Figure 6.9 and Figure 6.10 show that the buttons are able

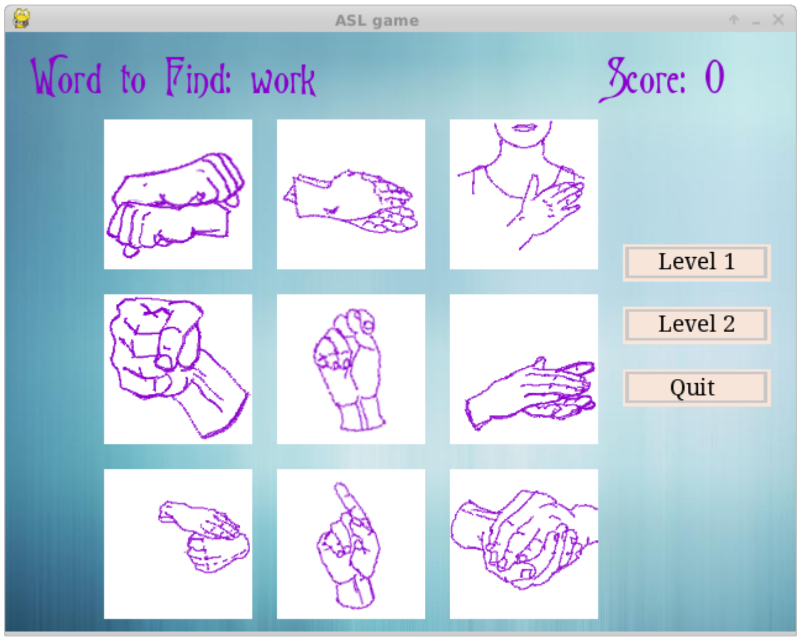


Figure 6.5: The level two screen of the ASL Game in Pygame

.

to present the corresponding English letter.

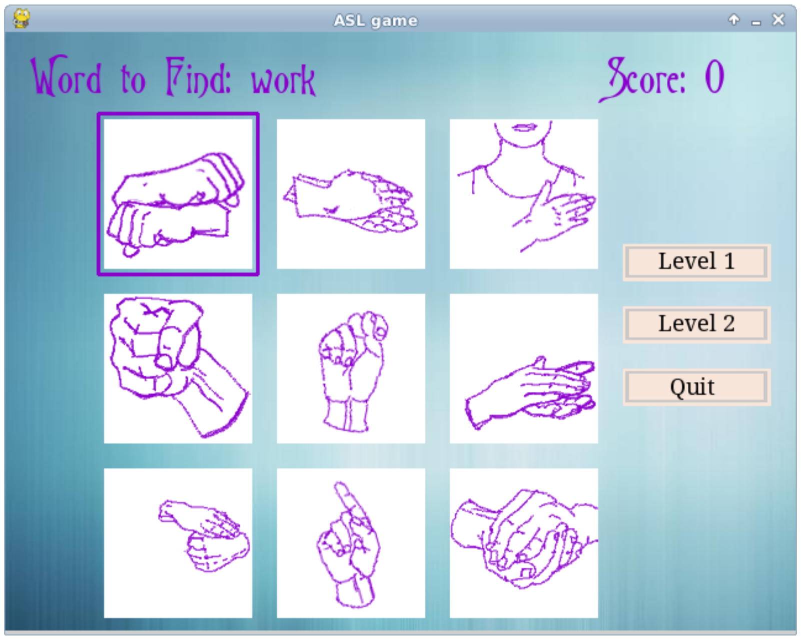


Figure 6.6: The Button hit to find the word work in Pygame

.

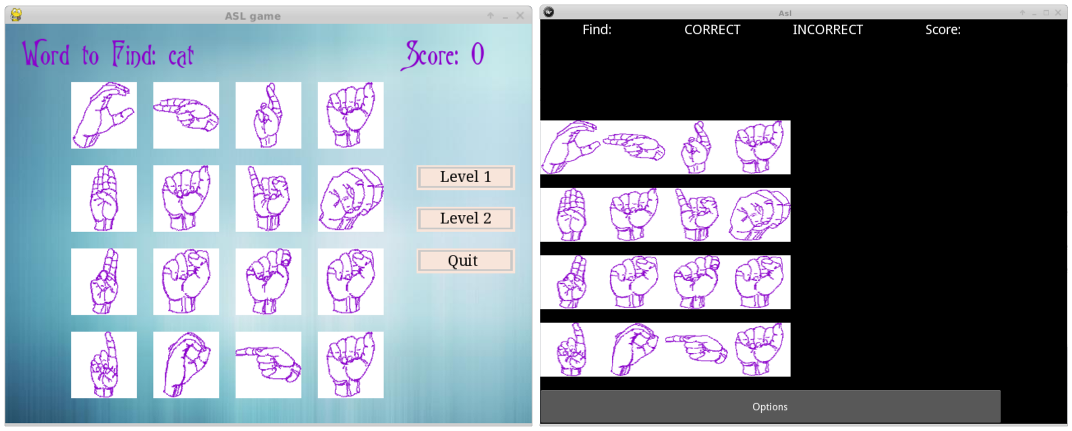
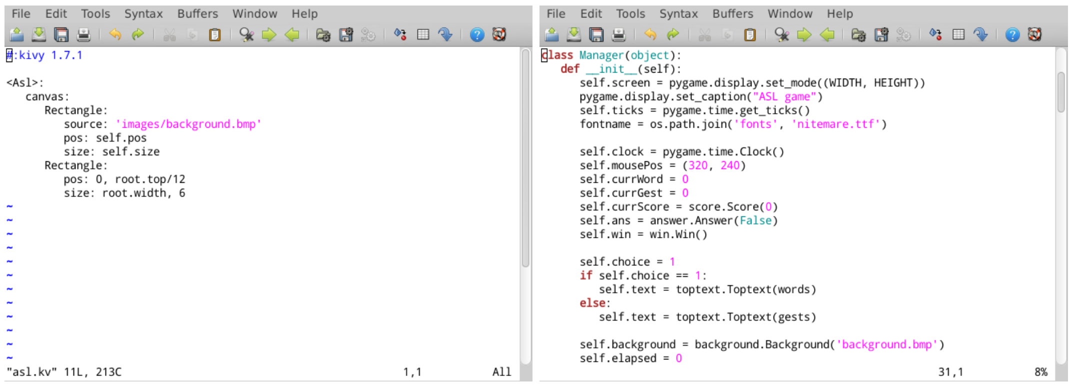


Figure 6.7: The introductory screen of the ASL Game in Pygame and Kivy

.



```
#:kivy 1.7.1

<Asl>:
   canvas:
      Rectangle:
         source: 'images/background.bmp'
         pos: self.pos
         size: self.size
      Rectangle:
         pos: 0, root.top/12
         size: root.width, 6
~
~
~
~
~
~
~
~
~
~
~
~
~
"asl.kv" 11L, 213C                                         1,1           All
```

```
class Manager(object):
   def __init__(self):
      self.screen = pygame.display.set_mode((WIDTH, HEIGHT))
      pygame.display.set_caption("ASL game")
      self.ticks = pygame.time.get_ticks()
      fontname = os.path.join('fonts', 'nitemare.ttf')

      self.clock = pygame.time.Clock()
      self.mousePos = (320, 240)
      self.currWord = 0
      self.currGest = 0
      self.currScore = score.Score(0)
      self.ans = answer.Answer(False)
      self.win = win.Win()

      self.choice = 1
      if self.choice == 1:
         self.text = toptext.Toptext(words)
      else:
         self.text = toptext.Toptext(gests)

      self.background = background.Background('background.bmp')
      self.elapsed = 0
                                                            31,1          8%
```

Figure 6.8: The differences between the Python and Kivy languages

.

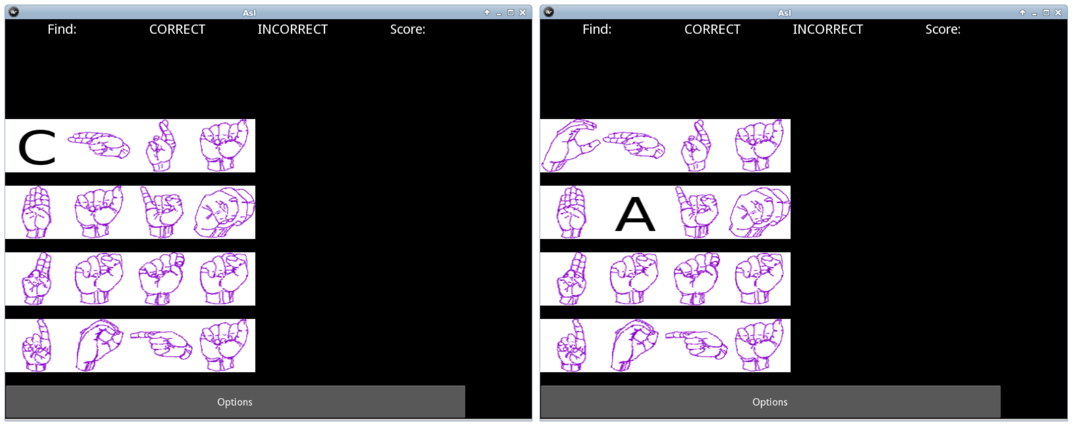


Figure 6.9: The first and second Button hits to find the word cat in Kivy

.

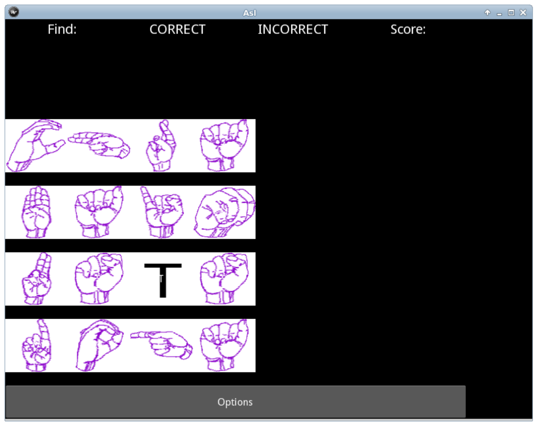


Figure 6.10: The third Button hit to find the word cat

.

Chapter 7

Discussion

In this section we discuss the findings from the results section. We address and provide some rational for the outcomes and suggest improvements for future studies.

7.1 Discussing the ASL Game Environment

First we explore creating an application that teachs a user ASL in a game setting. After creating this game, there are a handful of letters and words in level one and two that prompt the user to increase his/her score by choosing the correct answers. Each level has the potential to introduce the user to ASL by immersing him/her in a reward driven game. By increasing each correct answer by 50 and decreasing each incorrect answer by 10, the user can increase his/her score even with a few incorrect answers. This is important in attempting to keep the user's attention. ASL provides a vast world of knowledge that can fuel any game environment.

7.2 Discussing the Use of Python and Pygame

Python is a very useful tool in creating this game. It provides a smooth set of options for creating applications. Together with Pygame, there are very few limitations on what we are able to achieve. It gave us the option to create objects that could be quickly updated throughout the whole game. As a creator with no previous knowledge of Python before beginning this thesis, there was never a point where Python let us down. I am able to take previous coding knowlege of C and C++ and find a suitable substitute for the processes previously learned. In fact, Python/Pygame takes several processes that would have used significant more memory and line space and reduces it to one or two lines of code. As a result of the fewer lines of code, each function is handled differently than in C/C++, but I never felt hindered in the development process.

Pygame is a big factor in keeping the Python learning process possible. Because it has a sprite class implemented with collisions in tact, we are able to focus on making sure the sprites are user friendly and presentable in every way. We still create each base class, however, some of the functions such as collision detection are built in. Python is the right language in which to program this thesis especially utilizing its module, Pygame.

7.3 Discussing the Use of Kivy

Transporting the game from Python/Pygame to Kivy is a large undertaking. The code does not port directly into the Kivy library, so it is necessary to recode everything previously written. Even though the base language of both game versions is Python, the code is very different from Python/Pygame. Kivy has a great niche in crossplatform design. It combines an already portable language with the important aspects of mobile technology. For future research, I will futher explore completeting the rewrite of Python/Pygame into Kivy in order to port it to multiple mobile platforms. Also, utilizing Kivy from Step 1 will make any project intending to use Kivy as a crossplatform framework produce more desired results.

Chapter 8

Conclusion

In this thesis we investigated the use of portable game based learning to facilitate an understanding of American Sign Language. We played to the trends of technology by making the game accessible to any mobile device. We used a high-level programming language to create an environment conducive to learning while enjoying the game. This language choice allowed a light, fast-running game to be pursued. While this game provides the user with a scoring system to determine their current progress, we encouraged the user to learn at their own pace by leaving the game untimed. This corresponds to the theory that people learn more efficiently when their brain is relaxed. The primary goal of this thesis was to allow the user to create a correlation between ASL signs and symbols and the English language all while receiving a score. The secondary goal was to make the game accessible to multiple platforms of mobile technology. Our results indicate that creating a game in Python/Pygame and transferring it to Kivy in order to become portable, is entirely possible. This study and studies in the future on game-based learning should help validate the use of games while teaching; whether it be a language or just stratedgy, games such as Words with Friends and Angry birds are perfect examples.

Bibliography

[1] Richard Blunt. Does Game-Based Learning Work? Results from Three Recent Studies , 2013. http://patrickdunn.squarespace.com/storage/blunt_game_studies.pdf.

[2] Education Technology Services Educational Gaming Commons and Dr. Mary Shoemaker. ChemBlaster, 2010. http://gaming.psu.edu/projects/chemblaster.

[3] Joel Foreman. Game-Based Learning: How to Delight and Instruct in the 21st Century .

[4] 21st century language and literacy in gamestar mechanic: Middle school students' appropriation through play of the discourse of computer game designers., 2009. http://gamestarmechanic.com/teachers/about.

[5] Andreas Holzinger Martin Ebner. Successful implementation of user-centered game based learning in higher education: An example from civil engineering. 49, November 2007. http://www.sciencedirect.com/science/article/pii/S0360131505001910 .

[6] Federation of American Scientists. Educational Games. www.fas.org/gamesummit.

[7] Nobel Prize Organization. Productions. www.nobelprize.org/educational.

[8] Kurt Squire. Video Game-Based Learning: An Emerging Paradigm for Instruction , 2013. http://website.education.wisc.edu/kdsquire/tenure-files/09-PIQ-Squire-submitted.pdf .

[9] Marcus W. Shute Bruce Gooch Yolanda A. Rankin, McKenzie McNeal. User centered game design: evaluating massive multiplayer online role playing games for second language acquisition. *Proceeding*, pages 43–49, 2008. 10.1145/1401843.1401851.