SENIOR CAPSTONE TECHNOLOGY REVIEW

NOVEMBER 3, 2019

GESTURE RECOGNITION KEYBOARD

OREGON STATE UNIVERSITY
CS 461, FALL 2019

PREPARED FOR

SCOTT FAIRBANKS

PREPARED BY

GROUP 11

JEREMIAH KRAMER

Abstract

This document outlines my three responsibilities for the project: programming language, prediction engine, and character mapping. Each of these pieces have several options for technologies or methods that our team will need to choose. I researched and reviewed each piece in detail and offered my recommendation. Additionally, I identify and explain my individual role within my team and our overarching goal for the project.

Contents

I. Introduction

A. Purpose

The purpose of this document is to describe the research for the technologies or methods that might be used in my groups project and review them accordingly. This document will also identify my role in the group and a high level overview of what our group is trying to accomplish.

B. Scope

This document will be an examination of the technologies, methods, or options of three specific pieces for the project. These three pieces are only a fraction of the project and the rest of the pieces will be addressed by my other group members. Also, this document will only briefly explain what my team is trying to accomplish, since this has been discussed in more detail in a previous paper.

C. Overview

This paper is structured such that my role in my groups project and a high level view of what our group is trying to accomplish comes first. Next, the programming language being used for the software is discussed. Third, the prediction engine for how to match input data with the closest base gesture will be identified and reviewed. Then, the paper will discuss character mapping and shifting usability. Finally, I wrap up the document with a summary of what was discussed.

II. Individual Role and Project Overview

This section will identify and explain my role in the project and a high level overview of what my team is trying to accomplish.

A. Role

First, my role for the project will consist of mainly developing the engine for matching user input with a corresponding character mapping. In other words, I will help with developing and integrating the prediction algorithm. I will also be involved with app development. This includes UI design and communicating to the user how to use the app as effectively as possible.

B. Goal

The goal of this project is to provide an additional mode of input to users on their mobile devices. By doing this, we hope to improve the accessibility of text input, all while minimizing the effect on typing speed. We aim to achieve this by providing a set of easy-to-use motion gestures, and creating an intelligent engine that can decipher user inputs. Specifically, we will create an application that has a simplistic and intuitive design, provides a high degree of accuracy with minimal impact on typing speed, and enhances the users' overall text generation experience. The application will serve as a keyboard extension on the users phone so that the user can switch to our application to generate text by moving their mobile device in space.

III. Programming Language

The first piece of this project that I will address is the language of the software. The language that our team uses depends on the platform that we choose. The different platforms are discussed by another one of my group members in more detail, but we are deciding between iOS and Android, the two most commonly used platforms.

A. *iOS Research*

If we choose iOS as the platform for developing the app, then our options are: Swift, Objective-C, Javascript with React Native, Dart with Flutter, or C# with Xamarin [1]. Each of these languages has pros and cons. Swift is the most popular language used in most modern applications on the app store today. Further, it was developed and is maintained by Apple. Swift is also extremely robust and is always improving. Many libraries are available with Swift because it is an open source project open to anyone. Objective-C is the first language released by Apple back in 1984. This language is similar to C/C++ and many older apps are written in this language. Objective-C is well tested and is a stable language. A drawback of Objective-C is that it is losing popularity and there aren't as many Objective-C proficient developers. Javascript with React Native is a language and library combination. Javascript is a language mainly used for full stack development and web development. React is a library heavily influenced by Facebook. There are some drawbacks to this language. First, memory management is handled in a different way and there isn't a browser that will automatically handle memory efficiently. Second, performance is lacking compared to other languages. The main goal of React was to utilize a unified code base for both iOS and Android apps. However, the language has shown to cause more problems than it solves. React is in its early stages and will need to continue to get better. Dart with Flutter is a language and library combination created by Google. Dart is a language that is very similar to Java and C++ (object oriented languages). Flutter is a library that makes is very easy to create appealing-looking apps in a short amount of time. Google claims that this language is productive, approachable, and fast. This language is a newer language that is growing rapidly. Finally, Xamarin was created eight years ago. It is open source and is able to utilize ”hardware accelerations” that yields efficient performance. The drawback is that this language has decreased in popularity.

B. *Android Research*

If we choose Android as the platform for developing the app, then our options are: Java, Kotlin, C++, C#, Python, or Corona [2]. First, Java is the most commonly used language for Android app development. It is the official language for Android application development. Java is also the most supported language by Google due the many of the apps in the Play Store being built with Java. Java is a fairly complex language but there is a lot of support from online communities. Kotlin is an alternative to Java. The reason for Kotlin is to make it a little simpler than Java to include more beginners to Android app development. C++ cannot create an app by itself, it requires ”Android Native Development Kit(NDK).” Some drawbacks of C++ include increased complexity and thus more bugs. C# is very similar to Java as well. C# has many great features as Java such as garbage collection thus less chances for memory leaks. However, C# has an easier-to-read syntax than Java, so it is easier to understand. Like C++, Python requires a tools and packages in order to create applications. A common library that is used for developing apps is named Kivy. A drawback to this is that Kivy isn't natively supported. Finally, Corona is a development kit that is paired with a programming language named Lua. Lua is also simpler and easier to learn than Java. This language is mainly used for graphic-heavy games.

C. *Recommendations*

Following programming language research per platform, there are some good options to choose.

1) iOS: Given that our team chooses to develop in an iOS environment, I would suggest that we use Swift as our programming language. The reason for this is because Swift is a very popular language among iOS applications. This means that there is a lot of support from Apple and online communities.

2) Android: Given that our team chooses to develop in an Android environment, I would suggest that we choose Java or C++ as our programming language. The reason to choose Java is because of its immense support due to its popularity and similarity to C++. The reason to choose C++ is because we learned programming in C++ throughout our computer science collegiate careers. This means that all group members will have a C++ foundation, which will benefit our team.

IV. Prediction Engine

The second piece of the project that I will address is the prediction engine. This prediction engine will consist of an algorithm that can intelligently match specific user input to its corresponding character mapping.

A. Algorithm Background Information

This algorithm will be a machine learning algorithm. This means that, the algorithm needs to predict a possible output based on user input. Of the types of machine learning algorithms, this algorithm that our team needs is a ”supervised learning” algorithm as we need it to find a target or outcome variable predicted from independent variables generated by user input [3]. The independent variables will be inputted into this algorithm that generates desired outputs. This algorithm model will need to be trained to generated the desired outcome until we reach a high level of accuracy. Some examples of supervised learning algorithms include: Regression, Decision Tree, Random Forest, KNN, or Logistic Regression.

B. Motion Gesture Algorithms

Since there are multiple different supervised learning algorithms that we can use, we need to find possible algorithms that relate to motion gestures the closest. In an academic journal named ”Motion-Based Gesture Recognition Algorithms for Robot Manipulation,” there are three algorithms that are discussed. These algorithms include: Dynamic Time Warping, Hidden Markov Model, and Distance Metric Learning [4]. The journal assesses the accuracy of these accelerometer-based algorithms. Dynamic time warping and hidden Markov model algorithms are two classical pattern recognition methods.

1) Dynamic Time Warping: The dynamic time warping algorithm is a geometric approach to predicting an output. Fundamentally, the algorithm uses two different time series and will ”warp” one time series to make it resemble the other series. When warping, the goal is to make the accumulative distance between them minimal. The more the algorithm succeeds in its goal, the more accurate the algorithm is at predicting the desired outcome. With this algorithm, dynamic programming is used. The output is a matrix that indicates how much the two different time series differ when they are perfectly aligned. This algorithm fits perfectly with motion gesture application because it can use a set of known pattern templates and recognize an unknown pattern based on the known templates. It chooses the template that best fits one of the known patterns. Our team would leverage this by mapping specific gestures to a corresponding character.

2) Hidden Markov Model: The hidden Markov model algorithm is a statistical approach to prediction. The hidden Markov model is an extension of the observable Markov model. An observable Markov model is a discrete, first order Markov chain that has: a set of states, a set of state transition probabilities, and a specification of an initial state. ”A Markov chain is a mathematical system that experiences transitions from one state to another according to certain probabilistic rules” [5]. The output of the model’s process is a set of states at an instant of time and each state corresponds to a certain physical event that is observed [4]. The process becomes ”hidden” when the observation is a probability of the state. Therefore, the state isn’t directly observed. In a nutshell, based on specific probabilities of states at given times, a physical event can be predicted.

This fits well with motion gesture recognition. Since the hidden Markov model can have different typologies and models, the model that fits the best with motion gesture recognition is a model that describes observations in a chronological sequence. This model is named the left-to-right model. With this algorithm, our team would save specific patterns that correspond to a character mapping by setting precise probabilities of space and time.

3) Distance Metric Learning: The distance metric learning algorithm is a newer technique that recognizes patterns based on a group of known patterns. The unknown pattern is then assigned to the closest known pattern. The difference in the distance metric learning algorithm is the underlying closeness metric distance from Euclidean distance to the Mahalanobis distance metric. This new metric provides more accurate results. The drawback to this algorithm is that it is very complex compared to the other algorithms.

C. Review and Recommendation

The academic journal that suggested these three algorithms performed tests on each of them separately and reviewed their accuracy. The distance metric learning performed with 98.26% overall average recognition rate. The hidden Markov model achieved 94.44% overall average recognition rate and the dynamic time warping algorithm achieved 89.89% overall average recognition rate. In order to decide the algorithm to choose and implement, my team needs to weigh algorithm performance versus its complexity. This means that our team may choose an algorithm that doesn't yield the highest accuracy if another algorithm is easier to implement. The reason for this is because of overall time constraints for our project. With that being said, our team should wisely start with the distance metric learning algorithm or hidden Markov model.

V. Character Mapping

The third and final piece of the project that I will address is character mapping. Character mapping is extremely important because it affects user experience and algorithm success. Since the goal of our application is to create alternative user input via motion gestures, our team needs to map motions to letters and words. Given this, we have some different options for implementing character mapping.

A. Straight Line Simple

One approach is to map 30 unique gestures to 30 different characters. The reason we chose 30 is to allow for 26 different letters in the English alphabet and 4 additional gestures for special characters such as periods and commas. In order to make it easier for the user to learn the gestures, the gestures would be simple straight line movements of their device. However, mapping 30 straight line gestures would be an issue because 360 degrees divided by 30 is only 12 degrees of space for the user to attempt to generate text. This would be infuriating for the user because the degree for error is much too small. To take this a step further, our application could leverage the gyroscope embedded in mobile devices and initially flip the device on its side, facing upright, or facing downward. Then only 10 movements in space would be needed. That means that users would have 36 degrees (360 degrees divided by 10) of space on the devices axis to gesture. This is one possibility. However, if users wanted to add shortcuts as new gestures, this would limit them.

B. Series of Combinations

Another approach that our team can take is to map a series of gestures to certain characters and words or phrases. In other words, if the device is moved forward, right, right in succession, that would result in a corresponding letter. With this idea, the user would only have to move their device in six different directions: forward, right, left, backward, up, and down. This would allow our algorithm to be more successful. However, the drawback is recognizing when the next ”gesture” occurs (either within the combination series, or between characters). In order words, from the previous example, the algorithm would need to know when the user stopped gesturing forward, and started gesturing to the right. Also, the algorithm needs to know when to select a character and allow for the user to gesture the next character. Further, the amount of combinations could become complex, especially if the user adds many shortcuts. This would make it difficult fo rthe user.

C. Object Pattern Complex

Another idea is to gesture specific objects and map them to their respective character. For example, gesturing a ”circle” would result in the letter ”o.” This idea came from the academic journal that was referenced prior [4]. Although this is a valid approach, I think this would be very complex for the user. Even though this might result in a higher accuracy for the algorithm, text generation speed would suffer greatly.

D. Review and Recommendation

After review, mapping gestures to characters, words, or phrases, is difficult for users and the algorithm. As a team, we know that this will be a challenge and users and the algorithm will need to continuously learn from one another. With that being said, users should be able to map custom gestures to the application. This would result in a more user friendly application and increased usability. Our application should have default character-gesture mapping and allow for custom gestures. Also, the user should be able to edit the default mapping. The default mapping should be only moving the phone in simple straight lines with the phone’s initial position altering.

VI. CONCLUSION

This article discussed three pieces of my groups project and their related technologies or methods. These pieces don’t comprise the entire project. They were researched and reviewed in detail. In addition, this document discussed my role in my group for this project as well as a high level overview of my groups goal for the project.

References

[1] E. Chung, “What language are iOS apps written in?” [Online]. Available: https://www.zerotoappstore.com/what-language-are-ios-apps-written-in.html

[2] H. Kaur, “Top Programming Languages for Android App Development,” 19-May-2019. [Online]. Available: https://www.geeksforgeeks.org/top-programming-languages-for-android-app-development/

[3] S. Ray, “Essentials of Machine Learning Algorithms (with Python and R Codes),” 09-Sep-2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/common-machine-learning-algorithms/

[4] Tea Marasović, Vladan Papić, and Jadranka Marasović, “Motion-Based Gesture Recognition Algorithms for Robot Manipulation,” vol. 12, no. 5, p. 51, 2015.

[5] Henry Maltby, “Markov Chains,” https://brilliant.org/wiki/markov-chains/, 8-November-2019.