Jordan Hopley

Gesture Detection with Multisensor Fusion

B.Sc. Computer Science

22nd March 2019

Abstract

This report documents the ideas behind multisensor systems including multiple aspects of data fusion. Leap Motion sensors are examined and used to create a multisensor system which can better detect sign language than single sensor systems. An implementation of data fusion methods involving data, feature, decision and temporal fusion are designed and manufactured. This system is critically examined and evaluated with reference to related work. The results of this report highlight the need for a better understanding of multisensor fusion in technology and prove its effectiveness with rigorous testing.

Contents

1 Introduction

1.1 Project Overview

Innovative computerised systems are becoming more and more popular throughout the world lending help to almost every aspect of our lives. Technology offers great commercial, personal, communicative and palliative benefits. The use of sensors to allow a computer to access complex real-world data is increasing rapidly with the use of driverless cars. Sensors capturing different types of information have been at use for decades within the military providing efficient tracking surveillance. GPS satellites use data from multiple satellites and combine their findings to provide relatively accurate descriptions of peoples' locations no matter where they are in the world.

151,000 people regularly use the British Sign Language (BSL) in the home [1]. Of these, around 87,000 are deaf. This is a stark contrast to the number of people who cannot recite any part of the BSL. There have been many attempts at reducing this divide between the deaf community and the non-impaired through the vast number of devices which attach to the hands. This often requires wearing glove like sensors on the hands which can be uncomfortable and unoptimized. Roy Allela, an engineer, has earned worldwide praise due to his invention of gloves which detect the gesture that a person is doing – and reads it aloud [2]. However, this has received criticism from the deaf community as it does not accurately portray the intricate details of sign language [3]. It may seem that the only way to shorten the gap in communication is to give more people access to tools to learn sign language. In 2018, the UK government announced that they are 'open' to the idea of introducing a new GCSE aimed at teaching sign language in schools [4]. However, with the Disable and Elderly Assistive technology market expected to reach over USD 30 billion, it is clear there is room for an innovative technological solution which captures the needs of its users [5].

1.2 Project Aims

The overall aim of this project is to create a system which allows gestures to be interpreted by a computer and to evaluate different methods of data fusion. The effectiveness of a single sensor system will be examined and scrutinised with respect to a multisensor system. The multisensor system will use different fusion techniques to, hopefully, further increase the accuracy and efficiency of gesture detection. If the following aims are met, then this project may be considered a success:

- The use of Leap Motion sensors to provide input.
- Gestures can be created and saved.
- Gestures can be compared by similarity to already known gestures.
- Gestures can be created using a single sensor.
- Gestures can be created using multiple sensors at different angles.
- Multisensor gestures can be created using different fusion techniques.
- The system should not be overly slow or inefficient.
- The results of testing should be valid and allow interpretation.

1.3 Report Structure

This report will be organised into distinct chapters as described below:

- Background – This chapter will explore the different models define data fusion, their benefits and applications. Gesture recognition techniques will be analysed and examined as well as the intricacies of the Leap Motion software.
- Related Work – In this chapter, recent related studies will be analysed and critically assessed. Their techniques and methods will be evaluated and subsequently compared to the system proposed in this project.
- Design – The design chapter will detail the major decisions made that affect the implementation of the system. Here the different types of fusion used will be discussed as well as the strategy for gesture recognition.
- Implementation – This chapter will explore the creation of the system and how the decisions from the previous design section have been implemented.

- Testing and Analysis – The testing and analysis chapter will detail the results from rigorous testing of the system. It will compare the use of a single sensor to a multisensor setup. It will also highlight the effects of fusion of different levels of data.

- Conclusion – The conclusion will allow reflection upon the proposed system as well as the project as a whole. Furthermore, the system will be evaluated to find out whether the aims of the project have been fully met.

2 Background

2.1 Data Fusion

The Oxford English Dictionary defines fusion as 'the process or result of joining two or more things together to form a single entity' [6]. Fusion has its uses in a multitude of fields such as chemistry, physics, music and computing.

The Data Fusion Subpanel of the Joint Directors of Laboratories (JDL) provides an effective lexicon of various terms relating to the field of data fusion[7]. The JDL defines data fusion as: 'A process dealing with the association, correlation, and combination of data and information from single and multiple sources to achieve refined position and identity estimates, and complete and timely assessments of situations and threats as well as their significance.' The aim of data fusion is to ultimately produce an output that is richer in information than the input data with the output data having more authenticity. This ensures that the data more accurately portrays the scene that it is representing. In the context of this report, a scene defines the specific point in time and space that the sensors are actively detecting. For example, if a CCTV camera were to be the sensor, the scene would be the physical space that the CCTV camera is pointed towards and capturing images for.

Multisensor fusion is the synergistic use of multiple sensory devices in order to perform a task [8]. Karshney examines the use of multisensor fusion, its applications and what makes it successful and has stated that for a multisensor system to be more successful than a single sensor system, it should be more robust and provide improved reliability in the event that one of the sensors fail [9]. In this scenario, the system may continue to function but with less efficiency. Multisensor environments also allow for greater coverage of a scene by utilising multiple positions and angles to increase the percentage of a scene that is ultimately captured by the sensors leading to an increase in data availability and limiting the effect of occlusion. As the amount of correlating data increases, the probability that the data accurately captures the scene also increases. This is often, but not always, true with multiple sensors as they can be used to confirm or challenge the data that other sensors receive. In addition to advantages of multisensor systems mentioned by Karshney, multisensor systems allow for a vastly greater number of fusion techniques that are not limited to type, position or time.

Despite its attractiveness, multisensor fusion is a debatable topic in which many researchers in this field disagree. As Fowler proclaimed, one should ‘be wary of proposals for synergistic systems. Most of the time when you try to make 2 + 2 = 5, you end up with 3 ... and sometimes 1.9.” [10].

2.1.1 Levels of Data Fusion

Because of the vast range of subjects and disciplines that can benefit from the use of data fusion, there is no general classification that fits each one. The fusing process can differ drastically depending on the aims of the system, the type of data that is collected and the availability of data. Therefore, there have been many proposals to classify the different models that represent the levels of data fusion techniques. Perhaps the most common and useful model was proposed by the JDL which features two tiers in which the top level defines the data fusion process [11]

- Level 0 - Source Pre-processing
- Level 1 - Object Refinement
- Level 2 - Situation Refinement
- Level 3 - Impact Refinement
- Level 4 - Process Refinement

Source pre-processing usually occurs at the lowest level, in terms of abstraction, of data fusion as it concerns itself with the raw data that is collected from the sensors [11]. The aim is to prepare the data for the next stage by removing any unwanted information and unnecessary data. Object Refinement aims to collect data from one or more sensors and reduce them into proper data structures. The fusion of data at this stage has the benefit of creating accurate and reliable objects that represent the information, such as position and velocity, gathered from a scene. This includes transforming the data so that it adheres to a common unit. Situation Refinement explores the information gathered in the previous level to find relationships and patterns in the data which allow for inferences to be made about the data. Impact Refinement evaluates and describes the effect that the inferences made in the Situation Refinement level would have on the system as a whole. Many methods exist to assess the impact and threat assessment of an inference such as using Neural Networks or Blackboard systems [12]. Finally, Process Refinement evaluates the performance of the system and aims to improve the entire process of fusion [11]. Despite the numbered ordering of the different levels, it is not necessary to perform each level in the order given. Contrary to this, it is not uncommon for fusion system manufacturers to assume that the ordering is strict and must be adhered to [12]. This has led to criticism of the JDL model. Furthermore, Azimirad and Haddadnia conclude that the model is too generic and that is should be considered as a functional model instead of as a process model [12].

Durrant-Whyte provides a three-part classification system for data fusion regarding the relationship between the different sources of information and the data they gather [13]. This model focuses on a multisensor system and allows a suitable interaction between the data gathered from all of the sources:

- Competitive Information
- Complementary Information
- Cooperative Information

In a multisensor setup in which all of the data is gathered from sensors pointing at the same target, the information received is said to be competitive [13]. Competitive Information may be used to improve the confidence that the system has in its assessment of a scene. If two or more sensors provide different information about a particular subject, then the information that the sensors receive are considered to be Complementary as one set of information builds upon another. This is particularly useful when parts of the subject are obscured, and one sensor cannot accurately determine all aspects of the scene. In this scenario, another sensor is used, and positioned differently, to provide the information about the scene that the first sensor cannot detect. Cooperative Information occurs if a sensor is dependent on the information received from another sensor in order to represent, accurately, a scene. An example of this is the multimodal combination of audio and image inputs to create a single output consisting of the physical images of a scene and the sounds that the scene creates. This allows for a much more diverse and complex representation of a scene which harnesses more information than any of the sensors individually.

Dasarathy explores the consensus within the fusion community on a model for fusion systems which feature three distinct levels: Data, feature, and fusion [14]. Data fusion refers to the use and fusion of raw input data from the sensors. Feature fusion combines certain characteristics from one set of data with specific characteristics from another set of data. In a multisensor system, using feature fusion can produce cooperative information. Decision fusion takes two high level interpretations of a system in use and combines all of the results into a single final decision. This model is very simple yet effective. It covers multiple levels of abstraction which allows for a wide range of fusion algorithms and processing methods. However, Dasarathy argues that this model may be too ambiguous and proposes a fusion system that more closely at the inputs and outputs:

- Data In-Data Out (DAI-DAO)
- Data In-Feature Out (DAI-FEO)
- Feature In-Feature Out (FEI-FEO)
- Feature In-Decision Out (FEI-DEO)
- Decision In-Decision Out (DEI-DEO)

DAI-DAO focuses on the fusion of multiple data in order to produce a single set of data and is the at the lowest level of abstraction [14]. DAI-FEO also takes multiple data as its input but this fuses to produce a feature which represents a characteristic of the data. FEI-FEO allows the fusion of two features to produce a single feature. This allows for features that represent entirely different types of data to be fused together, creating a more meaningful interpretation that encapsulates multiple aspects of the scene. FEI-DEO uses multiple extracted features in order to make a decision, meaningful conclusion or logical analysis about the system. DEI-DEO allows multiple decisions to be made about a system. These decisions are taken and analysed to produce a final overall decision with the aim of better representing the original scene. In addition to this five-level hierarchy, Dasarathy suggests another form of fusion called Temporal Fusion which is the fusion of data, information, features or decisions that are fostered over a specific period of time. Temporal fusion can occur at any level using a single sensor or multisensor system. It can also increase the availability of data as more data is being collected.

Deciding on the correct model to use depends on the system requirements and limitations, the data that is available, and the purpose of the system. Each model offers their own advantages and disadvantages and should be carefully chosen. As well as this, care must be taken when using fusion algorithms to ensure that the inputs are properly fused as there are a number of different possible complications. Khalegi, et al suggest representing data fusion as a model of the challenges that must be overcome to properly and correctly fuse two inputs [15]. The model has a tree like structure and begins with four basic concepts with are evaluated in more detail as the tree is traversed downwards. These four concepts are as follows:

- Imperfection
- Correlation
- Inconsistency
- Disparateness

Imperfection in data gathered from sensors is almost unavoidable as there are many external factors which affect the integrity of the data [15]. Background noise, which would be non-existent in a perfect system, is often captured by sensors and will always affect the processing of data if it has not been properly accounted for. There are also possible hardware issues regarding the sensors as some of their components may become dislodged or malfunctioning. As well as this, sensors may also become unclean or dirty. This may cause a sensor to perform much less efficiently and possibly incorrectly due to the interference from the dirt. Cameras often suffer from this as a smudge close to the lens can blur the input data enough that it is almost unrecognisable to the scene that it is supposed to be interpreting. Correlation in data constitutes one of the major issues representing fusion systems, data correlation highlights the fact that sensors in a multisensor environment are all exposed to the same biases and therefore do not represent a higher degree of confidence that just using a single sensor. Inconsistency in data can occur because of spurious data causing outliers. Significant outliers can impact a system greatly when fused with correct data. Inconsistency can also be caused because of the incorrect sequencing of data which is plausible in a multisensor system. The effects of fusing out-of-sync data can render the system useless however the solution is simple. As a sensor collects data, a timestamp should be recorded and used to match datasets of the same, or similar - depending on the needs of the system, timestamps. Finally, the fusion of conflicting data can also cause inconsistencies. Conflicting data describes a scenario in which two sets of data depicting information about the same scene disagree and produce different outputs. Disparateness in data occurs with the fusion of data that may not be compatible with each other. Incompatibility is caused when data from multiple sources represent different information. For example, fusing together audio and visual data at a low level may not produce any useful output but fusing these at a feature level produces information that is useful and can be interpreted by the system.

2.2 Leap Motion

Leap Motion is a company based in the United States that builds a sensor which is capable of processing the motions of hands and fingers to use as an input. The sensors provide an alternative to using a computer mouse and keyboard that requires the hands to be suspended in 3D space rather than touching a controller or physical device. Leap Motion also provide an SDK which allows developers to use the Leap Motion sensor in an application.

The sensor is a small USB device that uses three infrared light emitters and two infrared cameras. The sensor sends out signals from the device and these signals bounce off any surface that they collide with. Infrared signals are then detected by the cameras in the sensor, allowing the sensor to receive data about anything that is in its field of view and interaction box.

2.2.1 Leap API

Leap Motion uses a right-hand Cartesian coordinate system in which the origin (0, 0, 0) lies directly in the middle of the top of the sensor [17]. The API measures four different variables:

- Distance in Millimetres
- Time in Milliseconds

- Speed in Millimetres per Millisecond
- Angle in Radians

As the sensor is active, it records multiple frames per second which contain information about the properties of objects in its view. A frame represents the core of the Leap Motion API as it takes a snapshot of a scene in any given time. Frames can capture the properties of the users' hands and it is possible for more than two hands to be present and properly captured. Frames can be identified using a timestamp or a unique ID which allows for easy extraction of specific frames which can be used to exhume information about a scene. Because of the occlusion problem, the sensor may not be perfectly accurate in its interpretation of a scene. Leap Motion, therefore, attempt to combat this by combining the input from the user with a model of the anatomy of a human hand. Since, sometimes, the Leap Motion software cannot be fully confident in its interpretation, a confidence value for each hand is included in each frame. A hand, according to the Leap Motion software, consists of five distinct fingers – each with four identifiable bones. This includes the Metacarpals, Proximal Phalanx, Intermediate Phalanx and Distal Phalanx. In reality, the human thumb does not have a Metacarpal however this is represented as a bone with a size of zero in the Leap Motion software as it allows for easier programming since all fingers will have four bones associated with them.

As of March 2019, the current release of the Leap Motion software does not natively allow for multiple Leap sensors to be used. However, in December 2018, an experimental release revealed support for multiple Leap sensors on one computer [18]. Because of the date of this release, it was not considered as an option for this project.

2.3 Gesture Recognition

A gesture is a form of non-verbal communication in which a body part conveys a particular meaning. Sign language is a type of communicative gesture. Communicative gestures are produced intentionally in order to communicate a certain message or piece of information. Alternatively, an informative gesture can often occur subconsciously at times and provide no extra meaning to what the subject is communicating. This includes actions like scratching the head or rubbing the chin.

2.3.1 Distance Calculation

Distance Calculation is a very simple technique used in many gesture recognition systems. To discover the similarity between two gestures, the distance between them can be calculated. In a scenario in which the distance between two gestures is zero, then the gestures are equal. The distance between two points can be interpreted in many different ways. The standard representation of the distance between two points is known as the Euclidean distance. This is the straight-line distance between the points. It represents the shortest distance between two points disregarding any obstacle that may be in between. The Euclidean distance formula is very useful as it can be used to measure distance in two and three dimensions. The formula to calculate the Euclidean distance in three dimensions is as follows [18]:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \tag{1}$$

If a gesture is made up of points in space, then the distance between two gestures can easily be calculated using this formula. Each point in space of one gesture corresponds to a point in space of another gesture. The distances between all these points represent the distance between two gestures.

2.3.2 Hidden Markov Model

Named after Russian mathematician Andrei A. Markov, The Markov Model can be defined as a ‘stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event’ [19]. The Hidden Markov Model (HMM) is a type of a Markov Model in which the states are hidden.

2.3.3 Kalman Filter

A Kalman filter is an optional estimation algorithm proposed by Rudolf Kálmán [20]. It can be used to estimate the value of a variable when it cannot be measured directly, but where indirect measurements are available. Kalman filters may also be used to estimate the nature of a state when there is noise in the system. The Kalman filter is used extensively in tracking, navigation and computer vision systems.

3 Related Work

The available technologies for gesture tracking and recognition has risen sharply in the last few years. The real-world applications are immense and ever-growing however they are impeded by the inaccuracies of the devices used and the improper fusion of data. Knowing the limitations of using a single sensor, many studies have begun to focus on using multiple sensors to gain a better understanding of their surrounding environment. This literature review will focus on a number of studies which detail the use of a multisensor system. It will focus on the two main sensory devices: Leap Motion and Kinect. Both of these sensors are relatively cheap and easily obtainable by the public – making them the ideal choice to create purposeful software for. Dispiritingly, the density of research projects within the field is scarce in relation to the use of multiple Leap Motion devices. However, a mix of both Leap Motion and Kinect devices is much more studied as is the use of multiple Kinect devices.

Fok, et al use high level data fusion in a multisensor setup to create a system that accurately detects numerical digits in the American Sign Language [21]. In their experiment, two Leap Motion sensors are used with the sensor with the highest confidence in its data being selected as the main sensor. The data from the other sensor is automatically aligned and then fused with the data from this main sensor at the fusion level. The fusion process uses a Kalman filter to estimate the values of unknown states which are passed to the HMM gesture recognition system. The results show that using two sensors provides more accurate gesture detection than using one sensor for every American Sign Language digit. Despite the use of a high level of fusion being successful, the system may have benefited more with a lower level since data from both sensors are of the same type [22]. This is because high level fusion tends to be more abstract and hold less information than the data level. The first ten numerical digits in American Sign Language are expressed using a single hand [23]. Whilst occlusion can occur, it is much less destructive than occlusion using two-handed gestures as the unknown states in single handed systems may be easier to identify. Perhaps an experiment using a range of gestures types would have been more indicative to the success of the system, however American Sign Language digits are visually similar so this does prove a challenge to their system.

Mohandes, et al propose a method of Arabic Sign Language recognition using fusion at the feature and classifier levels [24]. The decision level of fusion is also known as the classifier level [25]. For the purposes of consistency, this section will use to term 'classifier' as the paper does also [24]. Their system focuses on the recognition of static single-handed gestures. In comparison, the BSL is much more complicated than this as it contains a range of different gesture types. However, the Arabic Sign Language alphabet contains 28 signs with a number of signs looking distinctly similar to each other. This provides a challenge to their proposed system as any slight deviation could easily cause the gestures to be mismatched. The results show that their feature level fusion method was more successful than using singles sensors and more accurate than using classifier level fusion. The feature fusion level uses Linear Discriminant Analysis (LDA) which itself makes predictions using Bayes Theorem [26]. The classifier level fusion uses the Dempster-Shafer theory which allows the combination of data from different sources and assigning a 'belief' value denoting the strength of the evidence of the result [24].

Jin, et al use multiple Leap Motion devices to provide teaching and reference points to a robot performing object manipulation tasks [27]. Data is gathered from two sensors placed facing an object on a table capturing 3 possible gestures. Their results show that their fusion recognition system provides more accurate results for each gesture than using a single sensor placed beneath the hand and when a single sensor is placed at the side of a hand. The data is fused using a data level fusion of the original positions and directions of links and joints of the fingers. Reducing the noise the system requires capturing 5 frames and finding the average value. This can be done in a very short space of time. This study also highlights the effect that occlusion has on the Leap Motion sensors however their experiment shows how this problem can be mitigated with the fusion from a sensor placed facing the occluded part of the gesture. Finally, since the reason for gesture recognition, in this study, is to teach a robotic arm how to perform certain gestures, it is not unreasonable to suggest that the improvements made by the fusion system are not adequately sufficient. For example, the gesture recognition rate for the action of 'grasping an object from the side' is 83.3%. If the system were to recognise the gesture incorrectly then the robotic arm may act unpredictably. However, improvements made by the fusion recognition justify its use.

Marin, et al explored the idea of combining Microsoft's Kinect with a Leap Motion Controller to provide more accurate real time interpretations of the American Manual Alphabet (AML) [28]. Similar to the study by Mohandes, et al, this study uses only one-handed gestures to test their system [23]. This provides a slightly simpler challenge than using two-handed gestures. Their system uses feature extraction from both the Leap sensor and the Kinect sensor [28]. Three separate features are taken from the Leap sensor including the positions of fingertips, the position of the palm and the direction of the hand. Data from the Kinect sensor is used to provide two further features. These are: the correlation to the correct gesture; and the curvature of the hand contour using data about the fingers and palms. These features are then combined with training data and testing using a multi-class Support Vector Machine classifier. Results show that the combination of all five features produces the best results when compared to the accuracy of each feature separately.

4 Design

In this chapter, the major decisions affecting the design of the proposed system will be made while considering the aims and objectives of the system. Justification will be provided for these decisions as well as potential disadvantages or obstacles. The main part of this project relies on the design of the fusion implementation so the design of the fusion system will be finalized and explained. The set of data that will be used to test the systems' accuracy will also be discussed as this provides evidence of how well the proposed system will perform.

4.1 Requirements

Defining a clear set of requirements allows for easier development and distinct targets to aim for. Whilst similar to the aims of the project as a whole, the following requirements show the needs of the system specifically and how the system must perform in order to meet the project aims.

4.1.1 Functional

- The system shall provide real-time feedback on the confidence of the users' hands.
- The system shall allow the user to create a gesture.
- The system shall allow the user to test a gesture against already made gestures.
- The system shall receive sensory input for two devices at once.
- The system shall send data from one computer to another.
- The system shall utilise different fusion strategies.

4.1.2 Non-Functional

- The system shall perform the recognition algorithm in a short space of time.
- The system shall load efficiently every time the program is executed.

- The accuracy of gesture detection using fusion methods should be increased when compared to using one sensor.

4.2 Data Fusion Model

Because of the wide range of fusion models as described in the previous chapter, it is necessary to detail the best model for the new proposed system. Establishing a clear and concise model allows for a definitive direction in which to begin implementing the system. It also allows for anticipation of potential challenges and obstacles within the system.

The model of fusion that is best suited to the task depends on the problems with single sensor recognition. When using a single sensor to track multiple hands, it is possible that part of the hands will not be seen by the sensor. In this case, the Leap Motion will estimate the positions of hands and fingers that it cannot see. This can create uncertainty as well as incorrect data if its inferences are incorrect. There are two possible solutions to this: using a gesture recognition technique which can estimate unknown variables such as the Hidden Markov Model; or adding a second sensor to provide complementary data about the parts of the hands that the first sensor cannot accurately track. Due to its simplicity, Durrant-Whyte's model describing complementary data will be used to show the relationship between the two sensors in a multisensor setup [13]. However, this model is not enough to create a meaningful model to be used. After considering the different models, the widely accepted model describing data level, feature level and decision level fusion will be used with complementary data [14]. Despite its criticism by Dasarathy, this three-level model perfectly describes three different and clear levels at which fusion can occur. Dasarathy's modifications do provide increased detail in the specifics of input data and output data but this slight improvement does not warrant a complete change in system. However, the idea of temporal fusion that Dasarathy proposes for the reimagined model can easily be added to the old three level model. This temporal fusion will use competitive data meaning that the fusion process can occur, on this level, using one sensor as well as two. In conclusion, the four fusion types to be implemented include temporal, data level, feature level and decision level fusion.

Temporal fusion will be used to provide more accurate input to the gesture recognition system. It will do this by allowing the Leap sensors to record multiple frames and finding an average of these frames. The following diagram shows the proposed gesture recognition system with gestures created using the three levels of fusion, data, feature and decision:

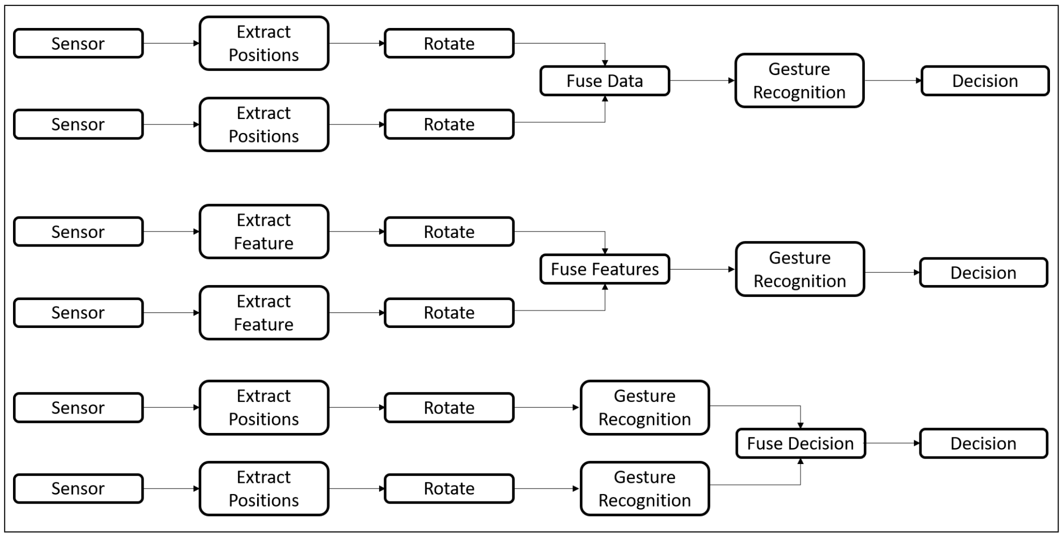


Figure 4.1 Data Fusion Model

In this model, data and feature fusion occur at the same point in the algorithm.

4.2.1 Data Level

At the data level, the lowest level of abstraction of data available from the Leap sensors are fused together. It could be argued that this data level should be classified as a middle level fusion rather than the low data level fusion as data level implies the rawest form of data that the sensors accumulate. However, the lowest form of data available to the user from the Leap sensors are frames which contain positional data. Because this is the lowest form of data available and for the purposes of this project, fusion of positional data from frames will be considered as data level. This provides a number of distinct advantages over a single sensor system. Firstly, the amount of data collected about a scene is higher allowing for more potential useful data. Additionally, the second sensor provides additional information about points of the hand that the first sensor cannot clearly see. In theory, this should help to reduce the impact of outliers in the data as they are counteracted by the more precise measurements of the other sensor. Furthermore, the data collected at this level will be of the same type with the only different between the two datasets being the angle at which the sensor has been positioned. Transforming these angles to a common angle ensures that the data from both sensors is of the same type, direction and size which allows for a much easier, less error-prone fusion.

4.2.2 Feature Level

Feature level fusion will extract features from the two data sets obtained from the two sensors and then combine them together. The fused features are then sent to the gesture recognition step. Features often include less data than that used at the data level as unwanted or unnecessary data is disregarded. Features, or important characteristics, are discovered about the data and from both sensors separately. These features include information that the other feature does not and combining these features allows for more accurate representations. Examples of features include fingers, hands, direction of fingertips and any other characteristic that describes the nature of what the sensors are tracking. Doing this allows a more detailed fusion than the data level as only the best data, in terms of accuracy, from each sensor are used in the final fused gesture.

4.2.3 Decision Level

The final level of fusion is the highest in terms of abstraction. The data from each sensor is extracted and interpreted separately only to fuse the decisions that are obtained during the gesture recognition stage. Decision level fusion allows for two different interpretations of the same scene. These interpretations lead to final decisions about the nature of a scene. This allows both decisions to be scrutinised and they provide confirmation about the validity of each others decision. If both sensors decide that the position of a hand is with a certain coordinate, then the probability that the hand is being tracked accurately increases. The decisions made by both sensors can be averaged to produce a final decision.

5 Implementation

The system that will be created shall allow the creation and testing of various gestures using a single or multisensor setup. Easily interpretable results should be presented to the user in a clear and concise manner.

5.1 Hardware

There are multiple options to overcome the fact that the native Leap Motion software does not allow more than one device to be active on a single computer. These include using multiple computers with their own installations of the Leap Motion software along with a connected sensor to each. It is also possible to connect two Leap Motion devices to one computer using a Virtual Machine in which the host has control over one device and the Virtual Machine has control over the other. Data collected in the Virtual Machine can then be transferred to the host over a web socket. However, this method has proved to be unreliable as both the host and Virtual Machine would compete for control over the Leap devices, rendering them unusable. Because of this, two computers are used to capture data separately.

5.2 Development Tools

The Leap Motion SDK offers support for a range of languages including C++, C#, Objective-C, Java, Python and JavaScript. Leap Motion recommends development using C as the previously mentioned languages are no longer actively supported however, they still provide an API for each legacy language [29]. The language decision is based on the API available, features and limitations of the language and familiarity of the language. Because of the complex nature of the implementation, it was decided that the familiarity of Java outweighed the advantages of C. Therefore, development of this system was performed using Java in Intellij.

5.3 Data Transfer

The data collected from the sensors must be sent to a single machine in order to be processed and fused together. If both Leap Motion sensors were to connect to the same computer, then the data gathered would be readily available on that computer to be used. However, since the standard Leap software does not allow this, the data from the second sensor must be sent to the host computer from the second computer. In this system, the second computer acts solely as a data collection tool gathering the information from the sensor and sending that information to the host. The data that it sends are frames as this is the lowest level of data retrievable from the sensor. Data is transferred using a Java TCP/IP socket. TCP provides a reliable connection between two computers that allow data to be sent over the internet [30]. Data flow, however, is highly dependent on the quality of the network and the presence of congestion. This may introduce small delays and offset the synchronization between the two different data sets however the amount of data per second that the Leap Motion sensor collects is high enough to ensure that any potential errors can be caught and removed without destroying the entire data set. UDP may offer an increase in throughput but this protocol is not as reliable as a TCP connection since the sender does not check if the packet has been successfully received by the receiver. This would have adverse effects on the data and may lead to its imperfection as data could be lost in the network. The two hosts could also be connected via a LAN using an ethernet cable although this has not been chosen as it increases the amount of hardware needed.

5.4 Gesture Creation

The following section discusses and explains how to create a gesture using the system. In this system, a gesture is made up of one or two hands. Each hand has 22 different points associated with it which represent its 3D position relative to the sensor. The fingers of a hand make up the first 20 points. Each finger contains 4 bones with the exception of the thumb which contains 3. However, the fourth non-existent bone in the thumb is represented as a point on the hand that is still relevant to this systems' interpretation of a gesture. The hand also contains coordinates for the positions of the palm and wrist. Overall, this gives a large number of different points that can be used to uniquely identify a hand.

There are two ways to create a gesture in this system. This includes using a single sensor with the sensor placed on the desk facing upwards creating a simple reference gesture. When using this technique to create gestures, it is recommended to use the visualizer which Leap Motion provide in order to verify that the sensors are interpreting the users' actions accurately.

The visualizer shows a 3D skeletal model of the users' hands in real-time. The second method of creating a gesture uses the fusion techniques with two sensors. The steps to creating a gesture with just one sensor are described below.

5.4.1 Capturing Frames

The first step to creating a gesture requires capturing the frames from the sensor. Frames are continuously developed by the sensors as the system is running but are only accessed when the user has begun creating the gesture.

5.4.2 Frame Optimisation

Since the Leap sensor typically captures at a rate of 80 frames per second, a lot of data can be gathered in such a short space of time. These frames are optimised using a process which identifies the confidence values of each hand in each frame. The confidence function, provided by the Leap API, describes the Leap software's own confidence in their interpretation of each hand – essentially highlighting how accurate their tracking is. Confidence is measured as a number ranging from 0 to 1 where 1 is the optimal confidence and represents that the tracking is as effective as possible, whereas a value of 0 shows that the tracking system has no accurate data on the scene. The optimisation function checks each frame for their confidence values. If there exists at least one frame where both hands have a confidence value of 1 then this frame is used, and the others are discarded. However, this is not always possible due to inconsistent tracking, smudges on the sensor or disruptive lighting conditions. In a situation where the confidence values are all below 1 then the function recursively checks each interval of 10% starting from 0.9, 0.8, 0.7... to 0 until a suitable frame or frames have been found. If no frame contains any hand which has a confidence value greater than 0, then all of the frames are rejected, and the algorithm does not continue.

5.4.3 Position Extraction

Each optimised frame is examined to extract the coordinates of all the points which make up a gesture. These points are stored as vectors in a dynamic array.

5.4.4 Temporal Fusion

Because there can be multiple frames left over after the optimisation stage, it is necessary to combine these frames into a single frame since a gesture is made up of only 22 points per hand. The positional data from the previous step is averaged to create a gesture. Temporal fusion is used by using multiple frames with different timestamps allowing for a more accurate gesture.

5.4.5 Rotation

Depending on the orientation of the sensor that was originally used to capture the data, rotation of the gesture must be performed. Whenever a gesture is created, it is always rotated along the Z-axis so that the yaw of the gesture is parallel to the sensor, creating the illusion that the gesture was performed with the sensor on the desk facing upwards. The angle of the sensor is obtained through input from the user and is combined with a vector (0, 0, 1) to create a transformation matrix which can transform the direction of any vector so that each vector faces a common direction.

5.4.6 Saving a Gesture

The coordinates of a gesture are converted into a custom vector type which allows for serialization. The dynamic array containing these custom vectors can then be serialized and saved to a file which can then be loaded by the program on its execution.

5.5 Gesture Recognition

Gesture recognition is the interpretation of a gesture into a mathematical model or algorithm. The recognition of a gesture, in this system, is a very simple procedure. It uses the distance calculation algorithm to measure the Euclidean distance between the two gestures. Since a gesture is made up of 22 points per hand, each point of the hand is compared to the corresponding point from the second gesture calculating the distance between each of these points. Adding all of these distances results in the total separation between the two gestures. Using the total separation as a comparison between gestures, in this system, would produce inconsistent results as the gesture that the user performed would have to occupy the exact same 3D space above the gesture with little room for movement. If the user were to perform the exact same gesture as the gesture it is tested against but, for example, the users' hands were 10cm further away from the sensor, then the separation of the gesture would imply that the gestures are not the same. To overcome this, the total separation of gestures is not used. Instead, the variance between the distances between the points is calculated. As the variance measures the spread of the data, gestures with points that are all equal distance away from another gesture are interpreted as being equal to gestures that are performed in the exact points in 3D space. The results can then be interpreted to mean that the gesture with the lowest variance is the most similar to the gestures it is tested against.

5.6 Data Fusion Processes

This section documents the implementation of the fusion model described previously. The model is split up into three different levels: data, feature and decision.

5.6.1 Data Level

The data level fusion process is similar to the gesture creation method. Data from both sensors are optimised individually and fused after their positions have been extracted and the positions have been rotated. The positional data is gathered from both sensors and is averaged to produce a single gesture. It is crucial to properly rotate each set of points as the sensors can be placed at a number of different angles.

5.6.2 Feature Level

In this system, the features that are fused are the hands. Each hand is considered as a unique feature of the data. Feature level fusion takes the most optimal hands, regarding their confidence values, from each sensor and fuses them to create a whole gesture containing only the best features. Consider an example in which the first sensor tracks the hands with confidence values of 0.7 and 0.2 for the left and right hands respectively; and the second sensor is tracking the hands with confidence values of 0.5 and 0.4 respectively. In this case, the first sensor has retrieved more accurate data about the left hand of the user whereas the second sensor has better tracking for the right hand. The left hand from the first sensor is then fused with the right hand from the second sensor – creating a new gesture. Fusion at this level may not occur if one sensor has better confidence values for both hands.

5.6.3 Decision Level

Decision fusion allows each sensor to process their data individually. The gesture recognition system is performed twice – one for each data set. The decision being made in this system is the value of the variance between the distances of the two gestures. Fusing these decisions requires averaging the variance value to provide a new score for the similarity of the gesture.

6 Testing and Analysis

The following chapter documents the testing process and provides analysis of its results. The testing process is split into three parts. The first part shows how the system performs when recognising gestures using a single sensor. The second part uses both sensors along with the different fusion techniques to produce results. Comparisons between both parts are made and the success of each level fusion is determined.

6.1 Test Set

In order to properly evaluate the success of the system, a set of gestures will be required. British Sign Language (BSL) provides a challenge to gesture recognition systems due to its complexity. The British Deaf Association estimate that there are around 151,000 BSL speakers in the UK [31]. Therefore, the test set for this system will be the signs for the alphabet in BSL as it may provide real-world benefits to the sign language community. The BSL alphabet can be categorized by four qualities [32]. These are:

- One-Handed
- Two-Handed
- Static
- Dynamic

One-Handed gestures are performed using just one hand. The only example of such in BSL is the letter C. Two-Handed gestures are performed using both hands. This includes every letter except C. Gestures performed with two hands will be the primary target for this system as these gestures suffer more from occlusion. Static gestures are such that the start of the gesture is the same as the end. Almost all of the gestures are static with the exception of H and J which are dynamic. Dynamic gestures have a different end point to what they start with meaning that there is some movement involved. This causes some concern regarding the effect of temporal fusion as the movement feature will be mitigated. This system will focus mainly on the use of two-handed static gestures, however results using one-handed and dynamic gestures will still be presented.

6.2 Single Sensor Recognition

6.2.1 Sensor Testing

For the first part of the testing, the sensor lay on the desk facing upwards as this is the position that Leap Motion recommends for optimal tracking. Each gesture was performed 10 times resulting in a total of 260 gestures tested. The number 10 was chosen to allow enough data to be collected whilst also being time efficient since a gesture takes roughly 4 seconds to perform and this allows one single gesture to be tested in less than a minute. The results were automatically saved into a file and transferred to an Excel spreadsheet. The following chart shows the overall results for a single sensor system.

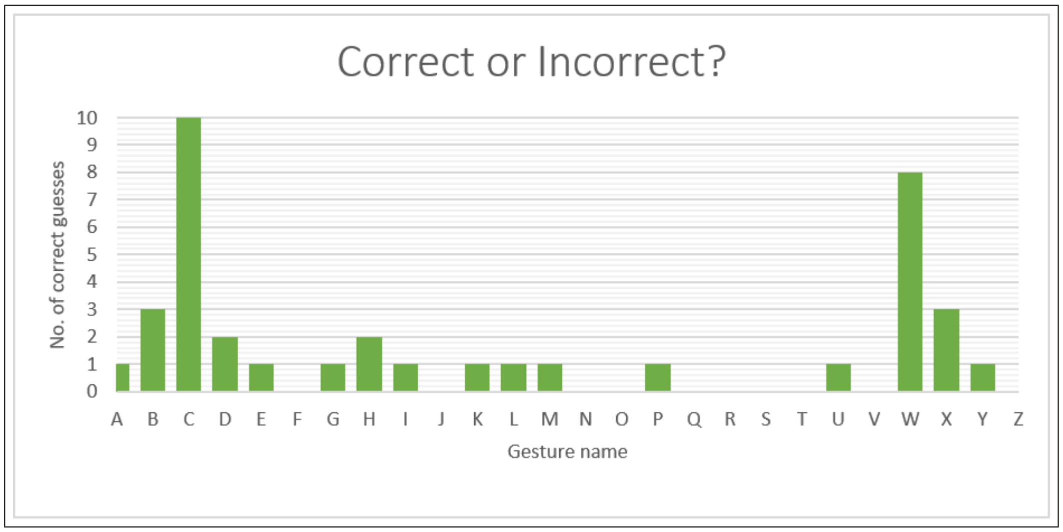


Figure 6.1 Single Sensor Results

Out of a possible 260 gestures, the system correctly guessed the users' gesture 38 times giving an accuracy rate of just 14.6%. The letter C is the clear winner with the system correctly guessing the gesture every time. This is likely due to the nature of the test set as C is the only one-handed gesture meaning that any input from the user involving multiple hands is likely to cause less similarity. It is interesting to note that the letter H was correctly guessed twice despite special consideration for dynamic gestures not being implemented. Furthermore, the letters B and T show unexpected results. These gestures do not contain much hand occlusion so, in theory, should be tracked well. However, the results show that these gestures suffered from inaccuracies at some point in the creation of the gesture. This may be due to the limitations of the Leap sensor or improper lighting conditions. The letter W was correctly guessed 8 out of 10 times providing some justification that this system works for two-handed gestures.

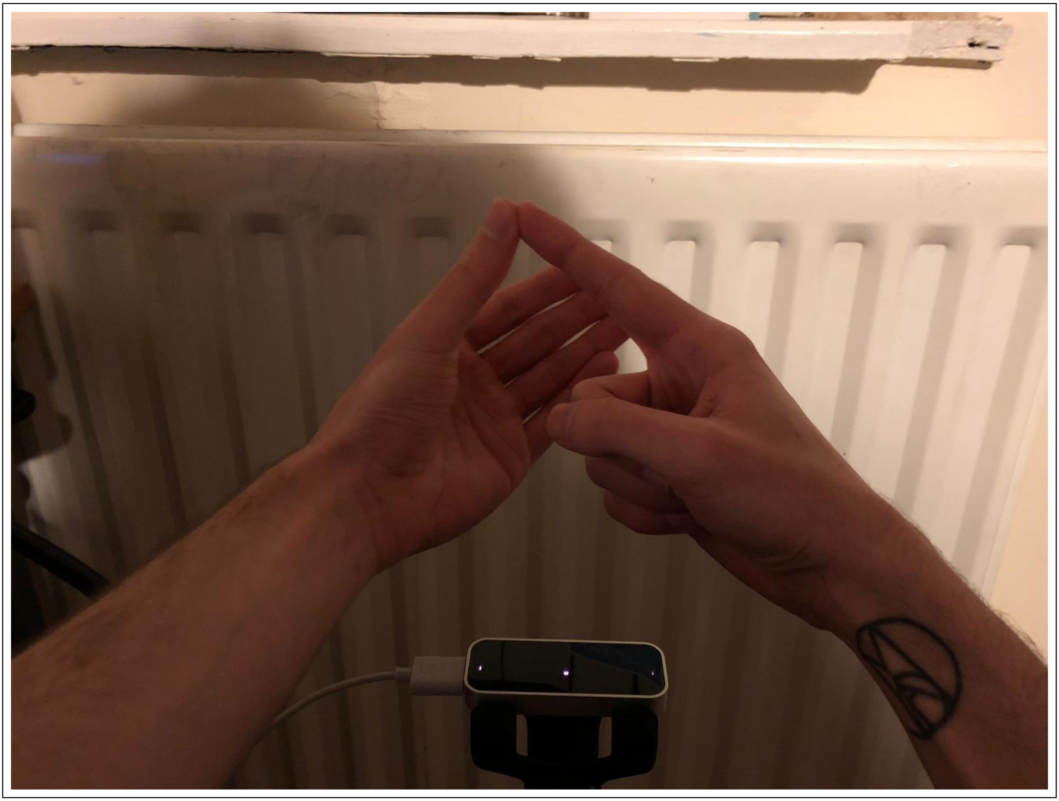


Figure 6.2 Letter A performed in BSL

It is possible to see how similar the users' gesture was to each letter of the BSL. The similarity ranking (rank) for each gesture shows this ranking. For example, if the rank is 1 then the system correctly guessed the gesture. If the rank is 2 then the system decided that there was 1 gesture - meaning the system guessed wrongly. The higher the value of the ranking, the worse the system performed at guessing the users' gesture.

Gesture	A	B	C	D	E	F	G	H	I	J	K	L	M
Rank	2	1	1	1	3	3	2	1	15	18	2	5	4

Gesture	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Rank	2	8	11	1	3	3	21	9	25	1	16	15	3

Table 6.1 Similarity Ranking for BSL

This table shows that, on average, only 6 gestures were correctly guessed. However, it also shows that 17 out 26 gestures were within one of the top 5 choices for their corresponding sign. This highlights how precise the gesture detection must be in order to correctly guess the gesture. There are many stages between acquisition of data and gesture detection that allows for errors. Generally, the lower the abstraction stage at which the error occurs, the greater effect it has on the final outcome of the system. These mediocre results may be due to a number of problems such as smudges on the sensors, poor tracking, noise in the system and the lack of noise reduction.

Breaking down the results for a single letter more closely allows for further analysis of the results. The following chart shows the average variance values from 10 results for the letter U. The Y-axis shows the average variance and the X-axis shows the letter which the letter U is compared against:

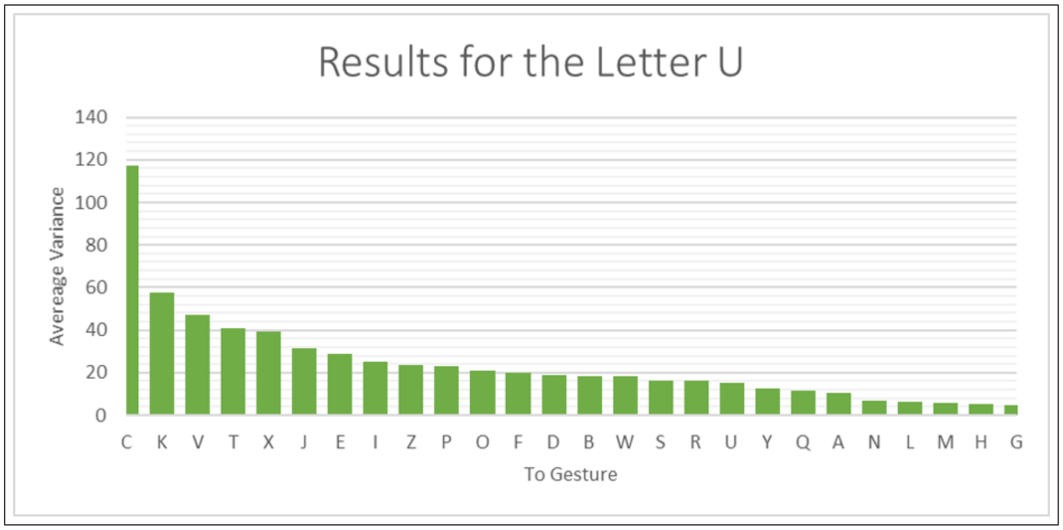


Figure 6.3 Results for the letter U

Visually, the letter U is similar to the letters A, N, L, M and H. This is highlighted by the gesture recognition system placing them close to the winning letter. It shows that the letter U was interpreted as a G despite the gestures being wildly different. A possible reason for this may be the nature of the letter G in sign language and how it effects the tracking accuracy of the Leap sensor. In right-handed BSL, the right hand in the G sign is completely occluded by the left hand. Instead of omitting the hand when the Leap sensor cannot see it, it attempts to estimate its position which can produce severe tracking abnormalities. The optimisation stage of this system mentioned during the implementation phase attempts to combat this problem however, there is a flaw in its design. The optimisation process allows the frames with the highest confidence values to be used. Using this method means that the highest confidence frame can still be very low if all the other frames are also very low. A possible solution to this would be to implement a minimum acceptable confidence value such as 0.5 to ensure that the tracking data is accurate enough.

6.2.2 Further Testing

As the accuracy rate for the detection of the BSL was so low, a new test set was devised to verify the authenticity of the gesture detection algorithm. One possible reason for such a low accuracy could be due to improper tracking as mentioned in the previous test with a potential solution involving a minimum confidence value for captured frames. Testing the gesture recognition algorithm itself can be done using a less complex test set. This second test set features 10 gestures specifically design to have minimum hand occlusion whilst still be easily identifiable and unique.

The following results show the number of times the system correctly guessed the gesture that the user was performing:

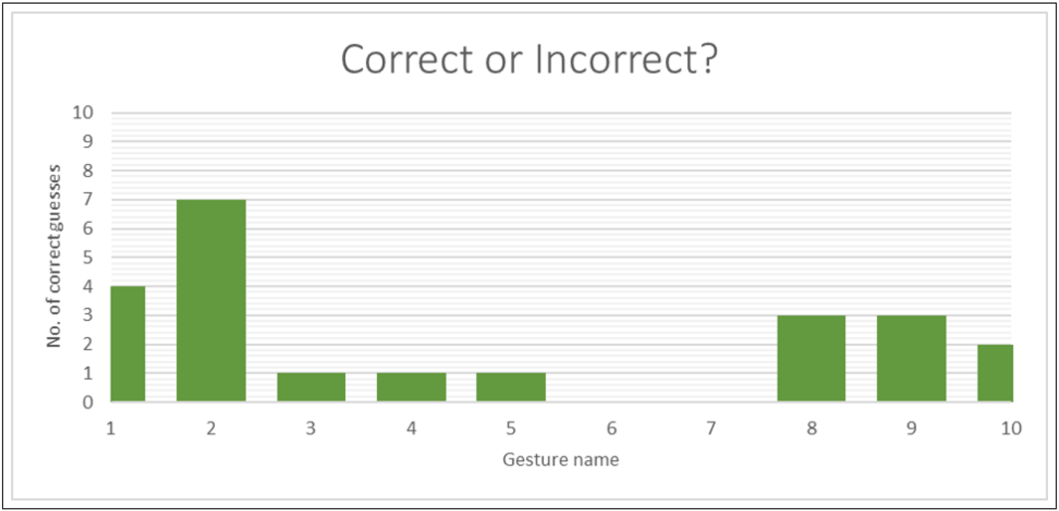


Figure 6.4 Results for Test Set 2

This test set was recognised by the system at a rate of 22% as there were only 22 correct guesses out of 100 which is a small increase compared to the accuracy of the previous test set. Comparing the average result from the 10 different tests gives the following ranking:

Gesture	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Rank	1	1	2	2	1	1	1	1	3	2

Table 6.2 Similarity Rankings for Test Set 2

Over the course of 10 tests, the gesture recognition accuracy rate rose to 60% with all tests placed within the top 3 ranking. These results lend merit to the theory that the BSL may be too complex for the system as this simpler test set proved much more successful. This also gives credibility to the gesture detection algorithm.

6.3 Multisensor Recognition

For the second part of testing, two sensors which are placed at varying angles will be used. There are 8 possible different angles at which a sensor can be aligned in this system. These angles start from 0°, in which the sensor faces directly upwards, and increase by intervals of 45°. Firstly, orthogonal sensor positioning was decided as the studies in this report have confirmed its suitability. On top of that, further refinements of 45° angles were decided to decrease the impact of side-on hand collusion. As there are 8 possible angles with just 2 sensors, then the total number of unique angle combinations is 7^2, or 49. For the purposes of testing, however, 4 of these combinations have been chosen and are presented as follows:

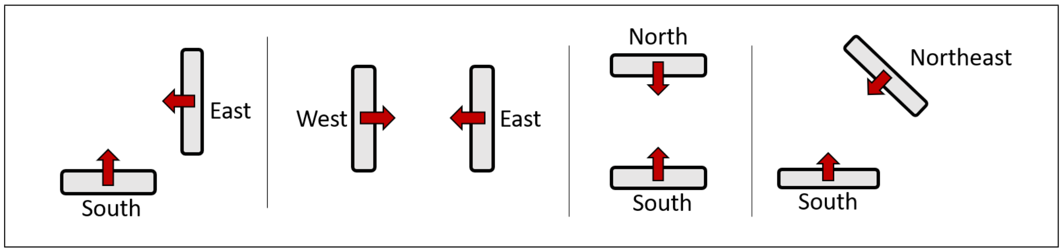


Figure 6.5 Angle Combinations

The letter V was chosen to be the subject of testing as it performed the worst in the similarity ranking as shown in Table 6.1 where it ranked an average of 25th place. The reason for this poor recognition could be due to the nature of the BSL sign of V as it is visually similar to the signs of M, N, L and H. The main feature of the letter V is the 'V' shape made with the fingers that rest on top of the palm. This 'V' shape is not detected when the sensor is placed below the user as the left hand, in right-handed signs, blocks the right hands fingers that perform the 'V' entirely.

The following section shows the results of the multisensor fusion algorithms with different angle combinations. The gesture 'V' was tested 10 times per angle combination and 10 times per method. The used are data fusion, feature fusion, decision fusion as well as including the results from sensor 1 and sensor 2 separately before fusion.

6.3.1 Config: South - East

The South-East combination was chosen because the right hand, in right-handed signs, is often blocked by the left hand. The blocked right hand can, in theory, be properly tracked by the sensor to the east while the southern sensor can track the left hand.

Method	Similarity Ranking
Sensor 1	25
Sensor 2	25
Data Fusion	21
Feature Fusion	2
Decision Fusion	25

Table 6.3 Results for South - East, Letter V

These results are very positive. As shown already in the previous testing, the letter 'V' again is not identified by each sensor individually. Using data fusion with the two inputs creates a minor improvement as the similarity ranking improves from 25 to 21. This could be caused by the mitigation of errors, outliers and noise - creating a less diverse, but more average dataset. The feature fusion method, however, performed exceptionally compared to the other methods, only being beaten by one other letter. The reason for this is that the feature fusion method allows the best hand to be used from each sensor. In this case, out of the 10 tests, Sensor 1 chose the left hand and Sensor 2 chose the right hand a total of 5 times. This is the ideal scenario as both sensors chose the hand which corresponded to their optimal field of view.

6.3.2 Config: West - East

The West-East combination was chosen as it allows the gesture to be viewed from both sides. This is useful if the user is left or right-handed allowing visuals of both.

Method	Similarity Ranking
Sensor 1	2
Sensor 2	4
Data Fusion	16
Feature Fusion	17
Decision Fusion	2

Table 6.4 Results for West - East, Letter V

In contrast to the previous combination, these results show that feature fusion performed the worst out of all the different methods - including single sensor recognition. This could be caused due to a lack of accurate tracking data since the feature fusion method is the only method that does not make use of temporal fusion - which aims to reduce the effect of outliers on the data. Furthermore, the decision fusion method performed much better than last time, tying with Sensor 1 as the optimal method.

6.3.3 Config: South - North

The North-South combination was chosen as it gives two completely opposite views of the gesture which is especially useful in BSL as one hand often occludes the other.

Method	Similarity Ranking
Sensor 1	25
Sensor 2	24
Data Fusion	25
Feature Fusion	2
Decision Fusion	25

Table 6.5 Results for South - North, Letter V

The results show, again, that feature fusion outperformed the other methods by a large margin.

6.3.4 Config: South - Northeast

The South-Northeast combination was chosen as the northeast sensor is aligned so that it is more likely to be parallel to the gesture - allowing for maximum coverage.

Method	Similarity Ranking
Sensor 1	26
Sensor 2	20
Data Fusion	18
Feature Fusion	3
Decision Fusion	24

Table 6.6 Results for South - Northeast, Letter V

The feature fusion method proved better once again. The results for Sensor 1 highlight the need for a second sensor further as it guessed that the letter V was the least similar despite being the gesture that was performed.

6.4 Summary of Results

In summary, the single sensor testing proves the need for a multisensor environment when attempting to detect BSL signs. The feature fusion method performed very well compared to the other fusion methods. This is to be expected as it only takes the best features from the data whereas the other fusion methods use all of the data to create a more average result. Therefore feature fusion has the power to detect gestures with a much higher accuracy but can also be spectacularly useless if the input data is not authentic.

The multisensor system proposed and implemented has proven to be successful in meeting its initial aims and objectives.

7 Conclusion

7.1 Review of Aims

- The use of Leap Motion sensors to provide input - Whilst a simple goal, it proved useful to understand the type of data that the Leap sensors require and how it interprets that data. The system performs well at receiving input from two sensors simultaneously.
- Gestures can be created and saved - Saving a gesture is integral to the testing phase. Gestures can be easily saved to a file and loaded in by the system. This could be improved as the gesture is simply stored in an easily modifiable format, allowing for potential errors if it is changed.
- Gestures can be compared by similarity to already known gestures - The gesture recognition system works well although not perfectly. Its methodology is very simple and could benefit from using more complex methods.
- Gestures can be created using a single sensor - The gesture creation system was executed poorly as it forces the user to define the inputs, however, it proved to work to some degree of accuracy.
- Gestures can be created using multiple sensors at different angles - The system allows for 8 different angles.
- Multisensor gestures can be created using different fusion techniques - Gestures can be created using a total of 4 different fusion techniques, however, the methods of implementation for these techniques are very simple and could benefit from more complex algorithms.
- The system should not be overly slow or inefficient - The results of testing appear almost immediately.
- The results of testing should be valid and allow interpretation - The results allow some interpretation although more data could have been presented to the user to allow more comparisons between performance, efficiency and accuracy.

Overall, all of the aims have been met to some degree however, there are many areas which could benefit from further improvement.

Additionally, this project has deviated slightly from its original proposal with the introduction of data fusion. Multiple Leap Motion sensors were discussed but were not seen as the main focus of the project. However, the project evolved to further explore the use and benefits of multisensor fusion when detecting sign language. The original aims have still been met, only with an extra

7.2 Future Work

The gesture detection algorithm uses a distance-calculation to measure how close two gestures are together. This is perhaps the simplest algorithm possible for gesture recognition. Further research into Hidden Markov Models would have proved to be enlightening and allowed for a much more developed system.

Additionally, the methods for fusion were very simple. The fusion methodology was explored in much greater detail than the actual techniques used to fuse data. This led to a simple implementation of the fusion methods.

There is a distinct lack of responses to the user regarding the behaviour of the system. Results are printed to a file rather than being presented to the user. To remedy this, improvements to the GUI would be welcoming such as data being immediately presented to the user through the use of graphs and tables. An internal visualizer would also have been useful to allow the user to see exactly how the system was interpreting the gestures.

7.3 Self Reflection

This project has proved to be one of the most challenging and interesting processes of my academic studies. It has taught me more about myself and how I approach problems. My time-management has historically been a downfall, however, I have certainly improved in this aspect. Furthermore, I have always doubted my own ability and this project has provided me with the confidence to go forward. Finally, I am thoroughly pleased with the progress I have made throughout my years at university. This project is a culmination of all of the skills I have developed academically and personally and I believe it to be a successful endeavour. I look forward to the future.

7.4 Closing Statement

To conclude, I believe this project to be a success as the initial aims have been met. Gesture recognition systems are always improving and the room for future projects is vast. This project has highlighted some of the problems that multisensor systems have and I have been able to express my own methods and ideas to overcome these problems.

References

[1] British Deaf Association (Date unavailable) ”BSL FINGERSPELLING ALPHABET” [On line publication].
https://bda.org.uk/help-resources/#statistics
Last accessed 22 March 2019.

[2] Samara Lynn (14 January 2019) ”KENYAN ENGINEER CREATES HIGH-TECH GLOVES THAT TURN SIGN LANGUAGE INTO SPEECH” [On line publication].
https://www.blackenterprise.com/kenyan-engineer-creates-high-tech-gloves-sign-language-into-speech/
Last accessed 22 March 2019.

[3] Michael Erard (9 November 2017) ”Why Sign-Language Gloves Don’t Help Deaf People” [On line publication].
https://www.theatlantic.com/technology/archive/2017/11/why-sign-language-gloves-dont-help-deaf-people/545441/
Last accessed 22 March 2019.

[4] Eleanor Busby (31 May 2018) ”Government open to new GCSE in British Sign Language following campaign” [On line publication].
https://www.independent.co.uk/news/education/education-news/british-sign-language-gcse-government-nick-gibb-campaign-school-a8377466.html
Last accessed 22 March 2019.

[5] Zion Market Research (Date unavailable) ”Global Disabled and Elderly Assistive Technology Market” [On line publication].
https://www.zionmarketresearch.com/report/disabled-elderly-assistive-technology-market
Last accessed 22 March 2019.

[6] Oxford Dictionaries (Date unavailable) ”Fusion Definition” [On line publication].
https://en.oxforddictionaries.com/definition/fusion
Last accessed 22 March 2019.

[7] F. E. White. (October 1991) ”DATA FUSION LEXICON”, The DATA FUSION SUBPANEL of the Joint Directors of Laboratories, Technical Panel for C3.

[8] R. C. Luo, M. G. Kay. (October 1989) "Multisensor Integration and Fusion in Intelligent Systems", IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, VOL. 19. NO. 5.

[9] P. K. Varshney. (2000) "Multisensor Data Fusion", Logananthara R., Palm G., Ali M. (eds) Intelligent Problem Solving. Methodologies and Approaches. IEA/AIE 2000. Lecture Notes in Computer Science, vol 1821.

[10] C. A. Fowler (1979) "Comments on the Cost and Performance of Military Systems", IEEE Transactions on Aerospace and Electronic Systems, AES-15(1), 2–10.

[11] J. Llinas, D. L. Hall (Date Unavailable) "AN INTRODUCTION TO MULTI-SENSOR DATA", ISCAS '98. Proceedings of the 1998 IEEE International Symposium on Circuits and Systems (Cat. No.98CH36187), Monterey, CA, 1998, pp. 537-540 vol.6.

[12] E. Azimirad, J. Haddadnia (January 2015) "The Comprehensive Review on JDL Model in Data Fusion Networks: Techniques and Methods", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 13, No. 1.

[13] H. F. Durrant-Whyte (1988) "Sensor Models and Multisensor Integration", Durrant-Whyte, H. F. (1988). Sensor Models and Multisensor Integration. The International Journal of Robotics Research, 7(6), 97–113.

[14] B. V. Dasarathy (January 1997) "Sensor Fusion Potential Exploitation—Innovative Architectures and Illustrative Applications", PROCEEDINGS OF THE IEEE, VOL. 85, NO. 1.

[15] B. Khalegi, A.Khamis, F. Karray (29 September 2015) "Multisensor Data Fusion a Data Centric Review of the State-of-the-Art and Overview of Emerging Trends", Taylor & Francis Group, LLC

[16] Leap Motion (Date Unavailable) "API Overview" [On line publication]. `https://developer-archive.leapmotion.com/documentation/java/devguide/Leap_Overview.html`
Last accessed 22 March 2019.

[17] Leap Motion (20 December 2018) "Experimental Release #2: Multiple Device Support" [On line publication].

http://blog.leapmotion.com/multiple-devices/
Last accessed 22 March 2019.

[18] Rosalind (Date Unavailable) "Euclidean distance" [On line publication]. http://rosalind.info/glossary/euclidean-distance/
Last accessed 22 March 2019.

[19] Oxford Dictionaries (Date Unavailable) "Markov model definition" [On line publication].
https://en.oxforddictionaries.com/definition/markov_model
Last accessed 22 March 2019.

[20] R. E. Kalman (Date Unavailable) "A New Approach to Linear Filtering and Prediction Problems", Research Institute for Advanced Study.

[21] K. Fok, N. Ganganath, C. Cheng, C. K. Tse (2015) "A Real-Time ASL Recognition System Using Leap Motion Sensors", Department of Electronic and Information Engineering

[22] M. Aeberhard, N. Kaempchen (2011) "High-Level Sensor Data Fusion Architecture for Vehicle Surround Environment Perception", BMW Group Research and Technology

[23] Jolanta Lapiak (Date unavailable) "American Sign Language Dictionary" [On line publication].
https://www.handspeak.com/word/
Last accessed 22 March 2019.

[24] M. Mohandes, S. Aliyu, M. Deriche (2015) "Prototype Arabic Sign Language Recognition using Multi-Sensor Data Fusion of Two Leap Motion Controllers", 12th International Multi-Conference on Systems, Signals & Devices.

[25] D. Ruta, B. Gabrys (2000) "An Overview of Classifier Fusion Methods", Computing and Information Systems, 7 (2000) p.1-10.

[26] Jason Brownlee (6 April 2016) "Linear Discriminant Analysis for Machine Learning" [On line publication].
https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/
Last accessed 22 March 2019.

[27] H. Jin, Q. Chen, Z. Chen, Y. Hu, J. Zhang (January 2016) "Multi-LeapMotion sensor based demonstration for robotic refine tabletop object manipulation task", CAAI Transactions on Intelligence Technology, Volume 1, Issue 1, January 2016, Pages 104-113.

[28] G. Marin, F. Dominio, P. Zanuttigh (2014) "Hand gesture recognition with leap motion and kinect devices," 2014 IEEE International Conference on Image Processing (ICIP), Paris, 2014, pp. 1565-1569.

[29] Leap Motion (Date unavailable) "Leap Motion Documentation" [On line publication].
`https://developer.leapmotion.com/documentation/`
Last accessed 22 March 2019.

[30] Oracle (Date unavailable) "Java Documentation" [On line publication].
`https://docs.oracle.com/javase/tutorial/networking/sockets/index.html`
Last accessed 22 March 2019.

[31] British Deaf Association (Date unavailable) "WHAT IS BSL?" [On line publication].
`https://bda.org.uk/help-resources/#statistics`
Last accessed 22 March 2019.

[32] British Sign (Date unavailable) "Fingerspelling Alphabet" [On line publication].
`https://www.british-sign.co.uk/fingerspelling-alphabet-charts/`
Last accessed 22 March 2019.

Appendices

Appendix A - The Project Proposal

Abstract

The following proposed project aims to one or more Leap Motion Controllers to reliably interpret the British Sign Language when signed by the user. Previous projects involving sign language interpretation have been met with many difficulties. This project focuses on maintaining a high degree of accuracy when interacting with the application. Testing the application will determine its success and viability against previous projects.

Introduction

The Leap Motion controller offers unique control over software using the hands suspended in the air. It uses three infrared light emitters that send signals out from the device and onto the users' hands above the controller. The infrared light then bounces back off the hands and into one of the two infrared cameras. These cameras interpret the receiving signal to accurately measure the hands size and position, along with the fingers, in 3D space. This technology allows us to create solutions to problems simply by waving our hands in the air.

British Sign Language (BSL) is spoken by around 125,000 people in the United Kingdom and a staggering 50,000 of those claim BSL as their first language [1]. BSL seems to be on the rise as a second language in the UK and a knowledge, at least at a basic level, is widely considered amicable and useful in today's society. Despite this, those who rely on BSL or any other sign language to communicate are still faced with some difficulties that those who are not hearing-impaired do not face. Efforts have been made to accommodate the problems that arise for the hard of hearing in recent years however there is still much more that could be done. Recently, the University of Manchester announced that its students had voted to replace 'clapping' with 'jazz hands' as to be more inclusive and this proved to be very controversial as some were for the idea and others were widely against the move, including US politicians such as Jeb Bush and television personalities like Piers Morgan [2]. Clearly,issues arise with these sorts of stories and perhaps the most effective way to move forward is to use existing technology along with education and tolerance to create a society that functions for all its members equally.

The proposed project will seek to create a piece of software that accurately and reliability translates hand movements performed over the Leap Motion controller into BSL that are displayed on a computer screen in front of the user in real time. The aim is to create a tool that is useful in the learning, teaching and understanding of BSL with a very small degree of difficulty for the user. The project will be guided using existing data about the Leap Motion controller; including its capabilities and limitations. Additionally, qualitative data will besought from speakers of BSL and possibly other sign languages to gain a deeper insight into using the hands and fingers as primary communication. To measure the effectiveness of the system, evaluative data will be collected. This will include quantitative data regarding the input and output of the Leap Motion controller and the software. As well as this, it will be necessary to garner feedback from the users of the product in the form of qualitative data to understand their experiences.

Background

The first device to be able to read signs from the hand was conceived by the Southwest Research Institute in 1977 [3]. The device, which took the form of a mechanical hand, allowed deaf-blind people to be able to communicate with anyone who is able to use a standard keyboard – before this, a deaf-blind person would only be able to communicate with someone who knew the same sign language as them [3]. The keyboard would change the position of the mechanical hand depending on the input [3]. The deaf-blind user would then feel the mechanical hand and be able to interpret the meaning from the position it is in [3]. This was the first time a deaf-blind person could receive succinct information from practically anyone else in the world [3]. Despite this, the device had numerous problems. The mechanical hand was unable to perform all the different letters of the alphabet to an acceptable degree of accuracy due to its rigidity and the device often performed much slower than a human interpreter [3].

Beginning in 2012, researchers at the Chinese Academy of Sciences have been working on a solution to the sign language translation problem that occurs when using mechanical hands like Dexter, Dexter-II and RALPH as well as using gloves as input – which may be too expensive to popularise [4]. This project used Microsoft's Kinect as their 3D camera to record the input of the user and can translate single words in the Chinese Sign Language and are working on accurately translating full sentences [5]. They are also currently working with Microsoft to allow the implementation of the American Sign Language into the device [5]. This project is similar to the proposed project. However, the proposed project will translate the British Sign Language which none of the previous solutions do and it will use the Leap Motion controller which is significantly cheaper than the Kinect.

An article by the Telegraph published in 2012 showed how researchers at Aberdeen University are using a normal camera to translate BSL into text that is displayed on an app [6]. This project is exciting as it can use the camera found on the user's phone meaning that the cost for such a product more likely will be the cost of just the app. Unfortunately, the project has not lived up to its hype and the product is not publicly available.

An article on the use of multiple Leap Motion Controllers was published in 2016. This article suggests that two Leap Motion Controllers are better than one [7]. Furthermore, it states that the optimal positioning of the devices be as follows:

- One Leap Motion Controller at the bottom, facing upwards towards the area of movement[7].
- The seconds controller to the side, facing down and sideways towards the area of movement at angle of 120°[7].

The article also gives details of the optimal distances between objects suggesting the diameter of the most efficient tracking area is 235mm if the second sideways-facing controller is a distance of 200mm away [7]. Furthermore, the article shows that using two controllers simultaneously increases the recognition rate significantly as opposed to using just one. For a simple gesture of pinching the ring finger with the thumb, a lone controller at the bottom of the hand has a recognition rate of 6.7% whereas by using two controllers, the recognition rate increases drastically to 73.3% [7].

The Proposed Project

Aims and Objectives

The aim of the project is to create design and implement the software required to be able to interpret the British Sign Language using a Leap Motion Controller to a high degree of accuracy and reliability. The completed product shall be easy to use for any user whether they suffer from hearing loss or not. The data required for this project will be sought after and obtained before and through the development phase.

The Application

One or more Leap Motion Controllers will be connected to a computer via USB. The application will appear on the users computer. The Leap Motion Controllers will be the users' input to interact with the application on the screen. Mouse and keyboard inputs will also be available to provide accessibility and additional options. The system will be able to read letters through the use of fingerspelling as well as full words. If the system detects that a letter is being signed, it will display the letter as it is gestured with no space after it. If a word is detected, the word will be displayed with a space at the end. Signs for the different punctuation marks will be created in order to allow the user more flexibility in their writing. The output will involve a text interface on the screen displaying the words and letters that the user signs above the Leap Motion Controllers. The output will be in real-time so the user can quickly check whether the application has provided the correct letter, word or punctuation. The GUI will be simple - featuring just an editable text box which displays the output and an options menu. The user will also have the option to store the text that has been signed into a document format.

Potential Obstacles

There are several challenges and obstacles that are expected during the proposed project, these include:

- The inaccuracies of using one leap motion controller as some hand gestures may be hidden from the device's cameras. To mitigate this issue, a second leap motion controller may be used.

- Using and translating data from two leap motion devices simultaneously.

- Some signs require stationary positioning of the hands and fingers while other signs are a combination of positioning and movement so the distinction of the different types of signs must be acknowledged.

- There are no BSL signs for punctuation. There are two options to fix this: the application can suggest the most likely piece of punctuation and automatically send it to the output for the user to verify; or signs could be created to represent the various punctuation marks.

Methodology

The project will be guided using the Waterfall method as the goals, resources and end result are unlikely to change throughout the project. The project has a sequential style in which the Waterfall method suits the needs.

Evaluation

During the evaluation, two types of data will be collected. The users experience with the software is instrumental to the success of the project. To test this, we will collect qualitative data from the test users. There will be three groups of test users:

- Deaf users who speak BSL.

- Non-deaf users who speak BSL.

- Users who do not speak BSL.

The data will be collected with a questionnaire which will evaluate whether the users' needs with the system have been adequately met. With this we will know if the system is a viable solution to the sign language interpretation problem. As well as this, we can record certain quantitative data to allow the software to work as efficiently as possible. This is important to reduce the inaccuracies that many other similar projects have experience. Such inaccuracies may cause the user to become frustrated with the system and ultimately make it much less useful. The following data will be collected:

- Time taken to read signs.
- Number of signs that are readable.
- Loading time of the software.
- Resources taken up by the software.

Furthermore, as previous projects have indicated, the proposed project is unlikely to be perfect – there will be inaccuracies. A feedback system will be implemented into the software which allows the user to verify whether the translation is correct. Additionally, this will give us access to further sets of data:

- Accuracy rate of words.
- Accuracy rate of letters.

This data will allow us to compare our implementation to other projects.

Programme of Work

The project will take approximately 20 weeks to complete starting from the 8th October 2018 and finalising on the 22nd March 2019. The stages of the project:

1. Background Research – Collating relevant research on BSL and Leap Motion as well as other projects that attempted to interpret sign language using computers.
2. Analysis of Research – This will involve reading and understanding the current market for the product. It will also allow for a detailed insight into the viability of such technology.
3. Requirements – Mapping the wants and needs of the user and the system.
4. Design – During this stage, the software will be planned thoroughly with the use of UML diagrams.
5. Development and Implementation – The project will be created in line with the design that will have been set out for it.

6. Testing – A working version of the software will be used by various users to validate whether the program is working as intended and meets the requirements that it is supposed to.

7. Improvements and Fixes – Any problems outlined during the testing phase will be fixed and any improvements to the software will be made.

8. Study – Once a finalised piece of software is made, a comprehensive study of the project will be produced.

9. Evaluation – The suitability and effectiveness of the project will be critically evaluated to give a final view on its success.

10. Finalise Project – During this stage, the project will be read and checked. Any outstanding work will be written up.

The following Gantt chart summaries the order of events:

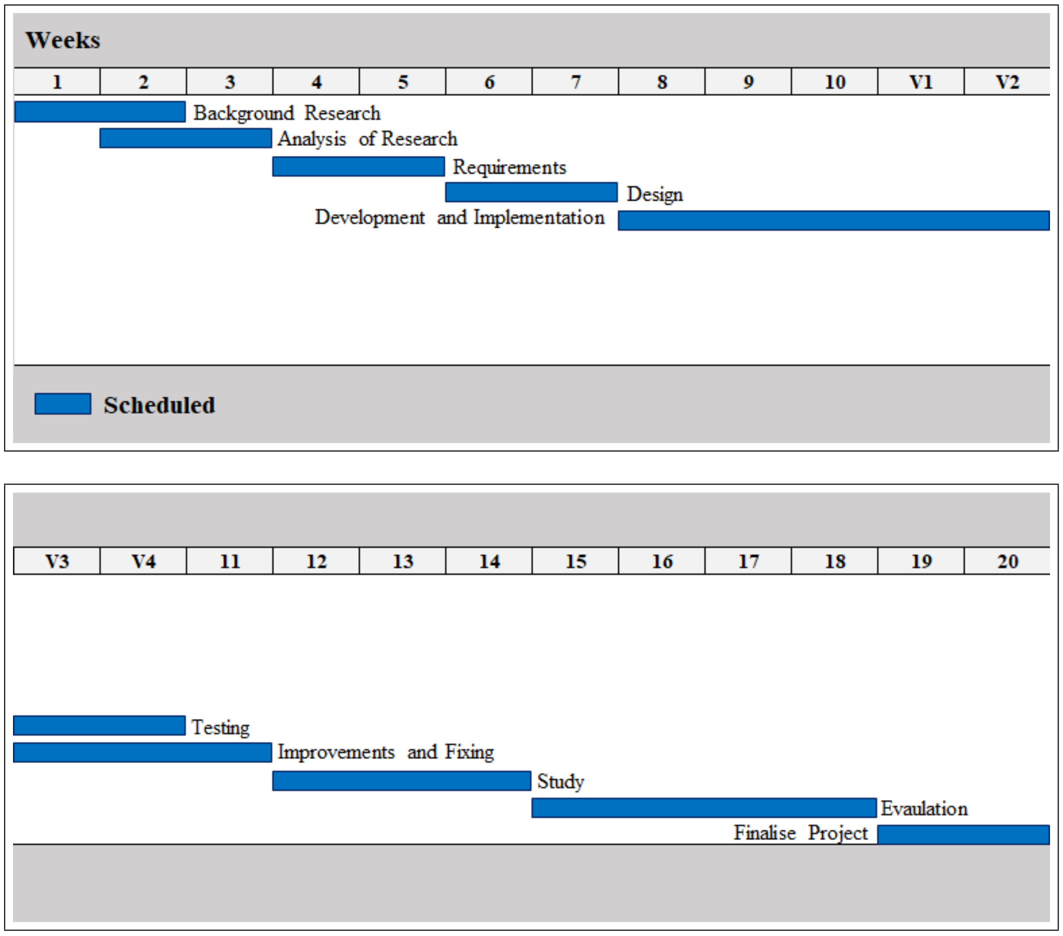


Figure 1: Gantt Chart for the programme of work

Required Resources

The project will require at least one Leap Motion Controller however more will drastically improve the scope of the project. Using two Leap Motion Controllers allows for a much higher degree of accuracy when identifying hand gestures as well as being able to manage the challenges faced with hidden gestures and problematic camera angles.

References

[1] UK Statistics and Facts (Disability Resource Centre, Date Unavailable)
`https://www.disability.co.uk/sites/default/files/resources/UKStatistics%26Facts.pdf`
Accessed 4th October 2018

[2] Jazz hands at Manchester University: the calm behind the storm (The Guardian, 5th October 2018)
`https://www.theguardian.com/society/2018/oct/05/jazz-hands-at-manchester-university-the-calm-behind-the-storm`
Accessed 5th October 2018

[3] Evolution of mechanical fingerspelling hands for people who are deaf-blind (David L. Jaffe, August 1994)
`https://www.rehab.research.va.gov/jour/94/31/3/pdf/jaffe.pdf`
Accessed 4th October 2018

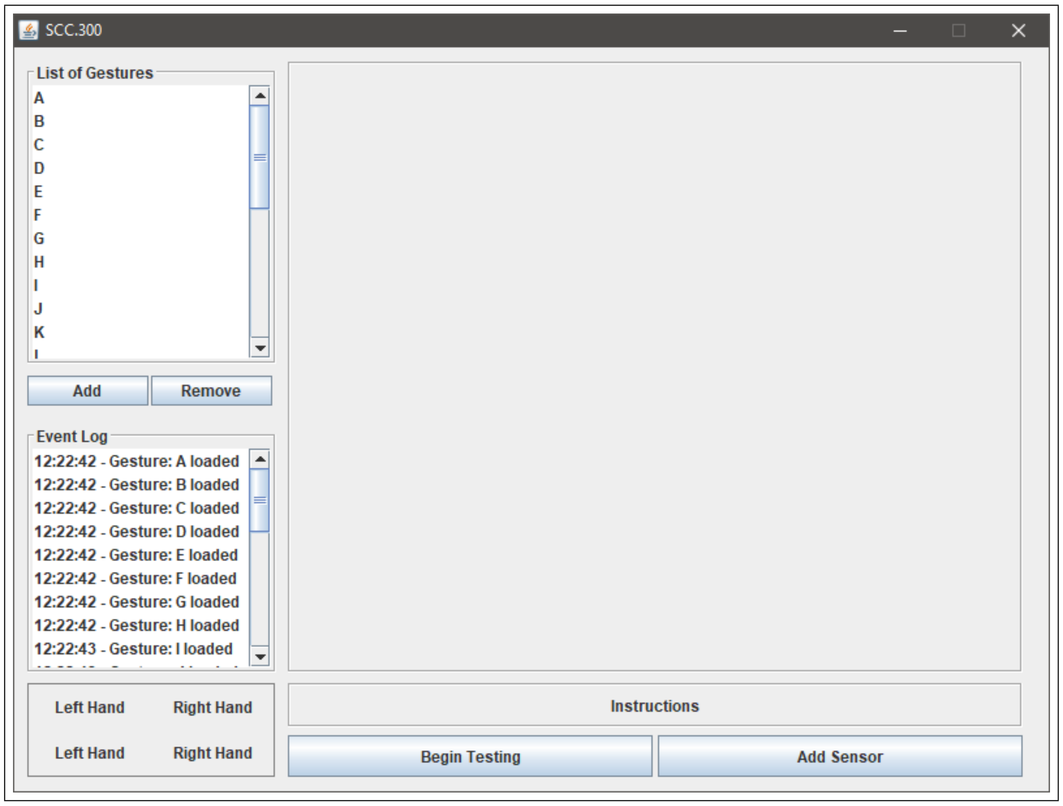
[4] Sign Language Recognition and Translation with Kinect (Xiujuan Chai, Guang Li, Yushun Lin, Zhihao Xu, Yili Tang, Xilin Chen, April 22nd-26th 2013)
`http://iip.ict.ac.cn/sites/default/files/publication/2013_FG_xjchai_Sign%20Language%20Recognition%20and%20Translation%20with%20Kinect.pdf`
Accessed 4th October 2018

[5] Kinect Sign Language Translator – part 1 (Microsoft blog editor, October 29 2013)
`https://www.microsoft.com/en-us/research/blog/kinect-sign-language-translator-part-1/`
Accessed 5th October 2018

[6] Sign language program converts hand movements into text (The Telegraph, 12th March 2012)
`https://www.telegraph.co.uk/news/science/science-news/9134827/Sign-language-program-converts-hand-movements-into-text.html`
Accessed 6th October 2018

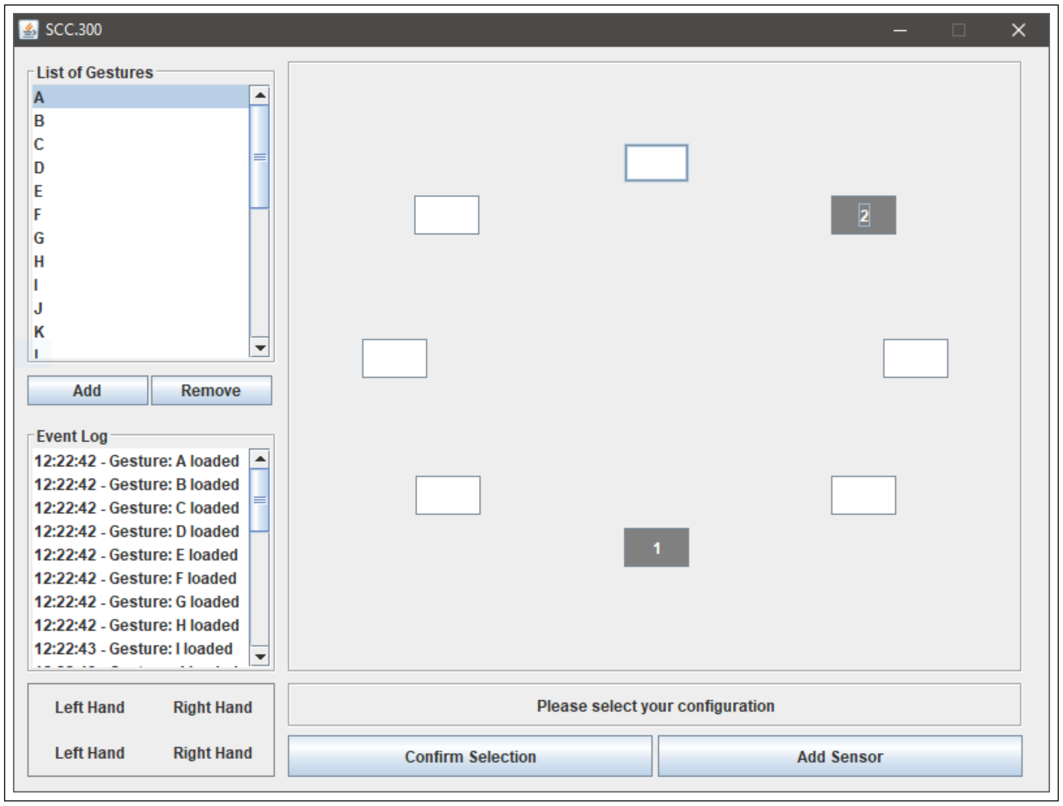
[7] Multi-LeapMotion sensor based demonstration for robotic refine tabletop object manipulation task (Haiyang Jin, Qing Chen, Zhixian Chen, Ying Hu, Jianwei Zhang, 2nd June 2016)
`https://www.sciencedirect.com/science/article/pii/S2468232216000111`
Accessed 6th October 2018

Appendix B - GUI



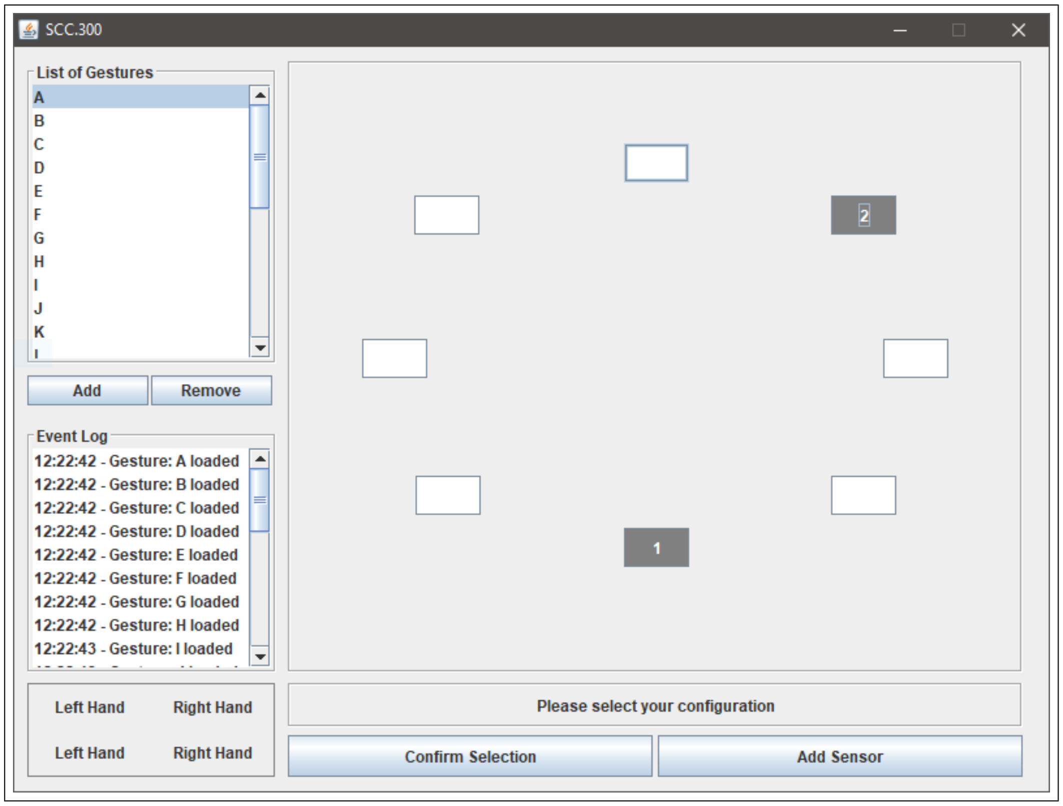
```
12:22:42 - Gesture: A loaded
12:22:42 - Gesture: B loaded
12:22:42 - Gesture: C loaded
12:22:42 - Gesture: D loaded
12:22:42 - Gesture: E loaded
12:22:42 - Gesture: F loaded
12:22:42 - Gesture: G loaded
12:22:42 - Gesture: H loaded
12:22:43 - Gesture: I loaded
```

Figure B.1 GUI



```
12:22:42 - Gesture: A loaded
12:22:42 - Gesture: B loaded
12:22:42 - Gesture: C loaded
12:22:42 - Gesture: D loaded
12:22:42 - Gesture: E loaded
12:22:42 - Gesture: F loaded
12:22:42 - Gesture: G loaded
12:22:42 - Gesture: H loaded
12:22:43 - Gesture: I loaded
```

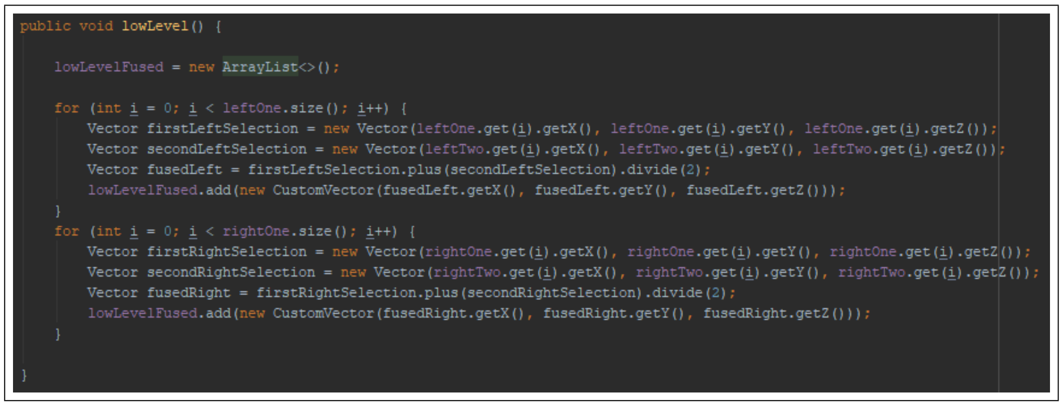
Figure B.2 Selecting Angles



```
12:22:42 - Gesture: A loaded
12:22:42 - Gesture: B loaded
12:22:42 - Gesture: C loaded
12:22:42 - Gesture: D loaded
12:22:42 - Gesture: E loaded
12:22:42 - Gesture: F loaded
12:22:42 - Gesture: G loaded
12:22:42 - Gesture: H loaded
12:22:43 - Gesture: I loaded
```

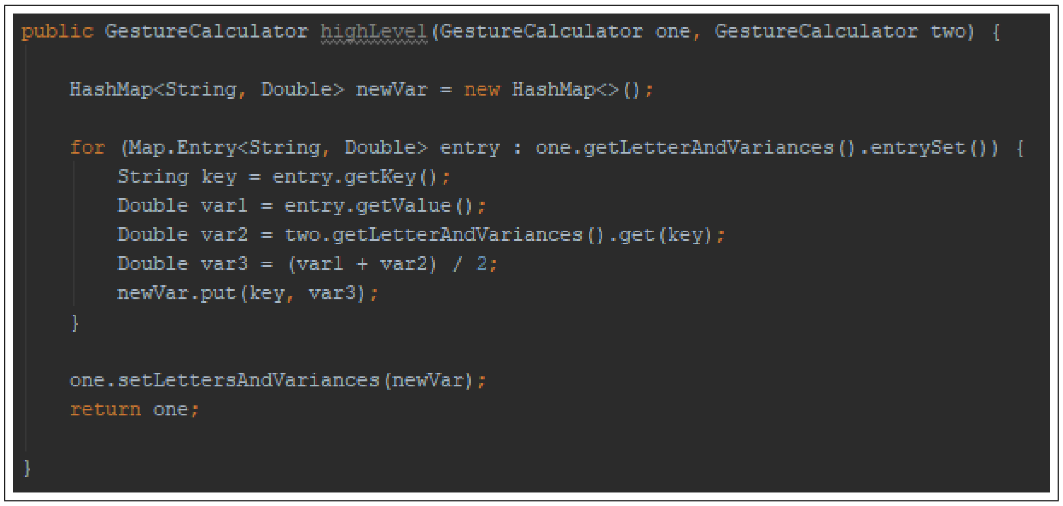
Figure B.3 Adding Gesture 'A'

Appendix C - Fusion Implementation



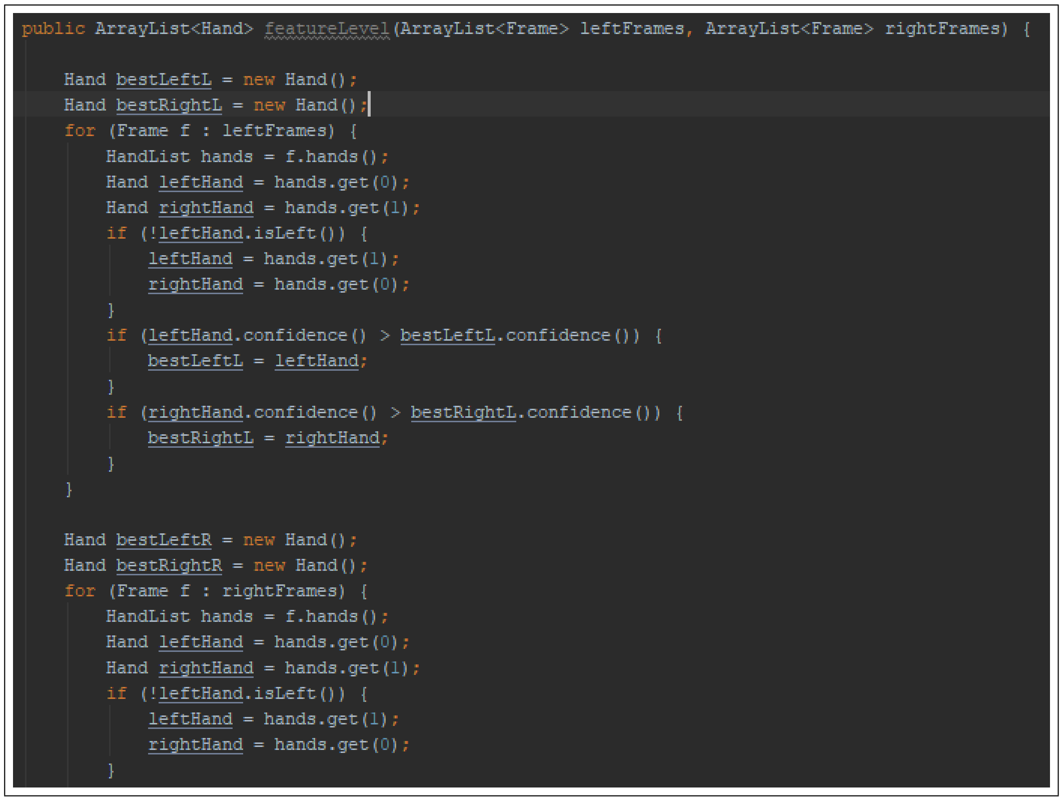
```java
public void lowLevel() {

    lowLevelFused = new ArrayList<>();

    for (int i = 0; i < leftOne.size(); i++) {
        Vector firstLeftSelection = new Vector(leftOne.get(i).getX(), leftOne.get(i).getY(), leftOne.get(i).getZ());
        Vector secondLeftSelection = new Vector(leftTwo.get(i).getX(), leftTwo.get(i).getY(), leftTwo.get(i).getZ());
        Vector fusedLeft = firstLeftSelection.plus(secondLeftSelection).divide(2);
        lowLevelFused.add(new CustomVector(fusedLeft.getX(), fusedLeft.getY(), fusedLeft.getZ()));
    }
    for (int i = 0; i < rightOne.size(); i++) {
        Vector firstRightSelection = new Vector(rightOne.get(i).getX(), rightOne.get(i).getY(), rightOne.get(i).getZ());
        Vector secondRightSelection = new Vector(rightTwo.get(i).getX(), rightTwo.get(i).getY(), rightTwo.get(i).getZ());
        Vector fusedRight = firstRightSelection.plus(secondRightSelection).divide(2);
        lowLevelFused.add(new CustomVector(fusedRight.getX(), fusedRight.getY(), fusedRight.getZ()));
    }

}
```

Figure C.1 Data Fusion Implementation



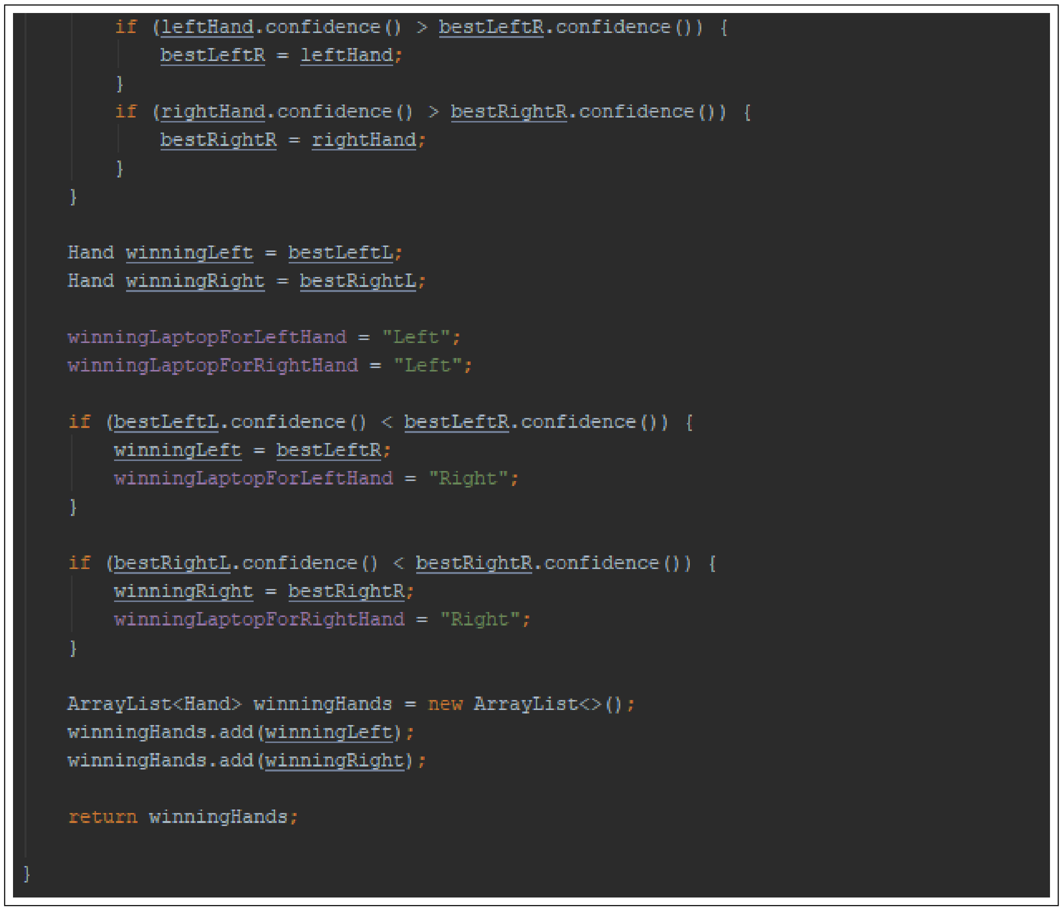
```java
public GestureCalculator highLevel(GestureCalculator one, GestureCalculator two) {

    HashMap<String, Double> newVar = new HashMap<>();

    for (Map.Entry<String, Double> entry : one.getLetterAndVariances().entrySet()) {
        String key = entry.getKey();
        Double var1 = entry.getValue();
        Double var2 = two.getLetterAndVariances().get(key);
        Double var3 = (var1 + var2) / 2;
        newVar.put(key, var3);
    }

    one.setLettersAndVariances(newVar);
    return one;

}
```

Figure C.2 Decision Fusion Implementation



```java
public ArrayList<Hand> featureLevel(ArrayList<Frame> leftFrames, ArrayList<Frame> rightFrames) {

    Hand bestLeftL = new Hand();
    Hand bestRightL = new Hand();
    for (Frame f : leftFrames) {
        HandList hands = f.hands();
        Hand leftHand = hands.get(0);
        Hand rightHand = hands.get(1);
        if (!leftHand.isLeft()) {
            leftHand = hands.get(1);
            rightHand = hands.get(0);
        }
        if (leftHand.confidence() > bestLeftL.confidence()) {
            bestLeftL = leftHand;
        }
        if (rightHand.confidence() > bestRightL.confidence()) {
            bestRightL = rightHand;
        }
    }

    Hand bestLeftR = new Hand();
    Hand bestRightR = new Hand();
    for (Frame f : rightFrames) {
        HandList hands = f.hands();
        Hand leftHand = hands.get(0);
        Hand rightHand = hands.get(1);
        if (!leftHand.isLeft()) {
            leftHand = hands.get(1);
            rightHand = hands.get(0);
        }
```

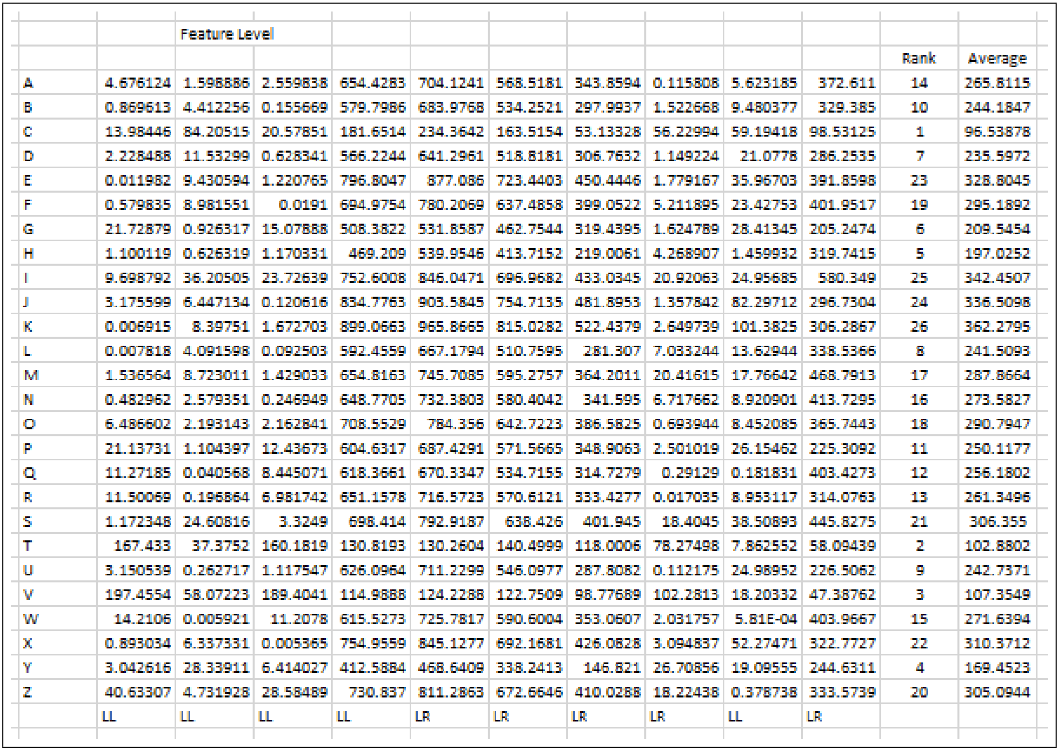
Figure C.3 Feature Fusion Implementation 1



```
            if (leftHand.confidence() > bestLeftR.confidence()) {
                bestLeftR = leftHand;
            }
            if (rightHand.confidence() > bestRightR.confidence()) {
                bestRightR = rightHand;
            }
        }

        Hand winningLeft = bestLeftL;
        Hand winningRight = bestRightL;

        winningLaptopForLeftHand = "Left";
        winningLaptopForRightHand = "Left";

        if (bestLeftL.confidence() < bestLeftR.confidence()) {
            winningLeft = bestLeftR;
            winningLaptopForLeftHand = "Right";
        }

        if (bestRightL.confidence() < bestRightR.confidence()) {
            winningRight = bestRightR;
            winningLaptopForRightHand = "Right";
        }

        ArrayList<Hand> winningHands = new ArrayList<>();
        winningHands.add(winningLeft);
        winningHands.add(winningRight);

        return winningHands;

}
```

Figure C.4 Feature Fusion Implementation 2

Appendix D - Example Resultset



		Feature Level										
											Rank	Average
A	4.676124	1.598886	2.559838	654.4283	704.1241	568.5181	343.8594	0.115808	5.623185	372.611	14	265.8115
B	0.869613	4.412256	0.155669	579.7986	683.9768	534.2521	297.9937	1.522668	9.480377	329.385	10	244.1847
C	13.98446	84.20515	20.57851	181.6514	234.3642	163.5154	53.13328	56.22994	59.19418	98.53125	1	96.53878
D	2.228488	11.53299	0.628341	566.2244	641.2961	518.8181	306.7632	1.149224	21.0778	286.2535	7	235.5972
E	0.011982	9.430594	1.220765	796.8047	877.086	723.4403	450.4446	1.779167	35.96703	391.8598	23	328.8045
F	0.579835	8.981551	0.0191	694.9754	780.2069	637.4858	399.0522	5.211895	23.42753	401.9517	19	295.1892
G	21.72879	0.926317	15.07888	508.3822	531.8587	462.7544	319.4395	1.624789	28.41345	205.2474	6	209.5454
H	1.100119	0.626319	1.170331	469.209	539.9546	413.7152	219.0061	4.268907	1.459932	319.7415	5	197.0252
I	9.698792	36.20505	23.72639	752.6008	846.0471	696.9682	433.0345	20.92063	24.95685	580.349	25	342.4507
J	3.175599	6.447134	0.120616	834.7763	903.5845	754.7135	481.8953	1.357842	82.29712	296.7304	24	336.5098
K	0.006915	8.39751	1.672703	899.0663	965.8665	815.0282	522.4379	2.649739	101.3825	306.2867	26	362.2795
L	0.007818	4.091598	0.092503	592.4559	667.1794	510.7595	281.307	7.033244	13.62944	338.5366	8	241.5093
M	1.536564	8.723011	1.429033	654.8163	745.7085	595.2757	364.2011	20.41615	17.76642	468.7913	17	287.8664
N	0.482962	2.579351	0.246949	648.7705	732.3803	580.4042	341.595	6.717662	8.920901	413.7295	16	273.5827
O	6.486602	2.193143	2.162841	708.5529	784.356	642.7223	386.5825	0.693944	8.452085	365.7443	18	290.7947
P	21.13731	1.104397	12.43673	604.6317	687.4291	571.5665	348.9063	2.501019	26.15462	225.3092	11	250.1177
Q	11.27185	0.040568	8.445071	618.3661	670.3347	534.7155	314.7279	0.29129	0.181831	403.4273	12	256.1802
R	11.50069	0.196864	6.981742	651.1578	716.5723	570.6121	333.4277	0.017035	8.953117	314.0763	13	261.3496
S	1.172348	24.60816	3.3249	698.414	792.9187	638.426	401.945	18.4045	38.50893	445.8275	21	306.355
T	167.433	37.3752	160.1819	130.8193	130.2604	140.4999	118.0006	78.27498	7.862552	58.09439	2	102.8802
U	3.150539	0.262717	1.117547	626.0964	711.2299	546.0977	287.8082	0.112175	24.98952	226.5062	9	242.7371
V	197.4554	58.07223	189.4041	114.9888	124.2288	122.7509	98.77689	102.2813	18.20332	47.38762	3	107.3549
W	14.2106	0.005921	11.2078	615.5273	725.7817	590.6004	353.0607	2.031757	5.81E-04	403.9667	15	271.6394
X	0.893034	6.337331	0.005365	754.9559	845.1277	692.1681	426.0828	3.094837	52.27471	322.7727	22	310.3712
Y	3.042616	28.33911	6.414027	412.5884	468.6409	338.2413	146.821	26.70856	19.09555	244.6311	4	169.4523
Z	40.63307	4.731928	28.58489	730.837	811.2863	672.6646	410.0288	18.22438	0.378738	333.5739	20	305.0944
	LL	LL	LL	LL	LR	LR	LR	LR	LL	LR		

Figure D.1 Example Feature Fusion Result Data for Letter V in South-Northeast Combination