**Gesture Interpreter**

Andrew Ladlow

Contents

1. [Introduction 3](#_Toc432508450)

[1.1 Motivation 3](#_Toc432508451)

[1.2 Objectives 3](#_Toc432508452)

[1.3 Report Overview 3](#_Toc432508453)

1. [Background 4](#_Toc432508454)

[2.1 Research 4](#_Toc432508455)

[2.2 Related Work 4](#_Toc432508456)

[2.3 Similar Applications 4](#_Toc432508457)

1. [Design 5](#_Toc432508458)

[3.1 A 5](#_Toc432508459)

1. [Implementation 6](#_Toc432508460)

[4.1 A 6](#_Toc432508461)

1. [The System in Operation 7](#_Toc432508462)

[5.1 A 7](#_Toc432508463)

1. [Testing and Evaluation 8](#_Toc432508464)

[6.1 A 8](#_Toc432508465)

1. [Conclusions 9](#_Toc432508466)

[7.1 A 9](#_Toc432508467)

[References 10](#_Toc432508468)

**Chapter 1**

# Introduction

## 1.1 Motivation

Human-computer interaction has, for many years, been limited to mouse pointing and typing based devices. With the more recent development of smartphones we have seen numerous breakthroughs with touch based devices. These methods are all inherently unnatural when compared with real world interaction, however. In all of these cases users are forced to interact through some kind of interface as opposed to naturally communicating via hand signals or body language.

Currently, there are several publically available devices which aid in monitoring, recording and displaying these more natural forms of communication. The gesture interpreter aims to utilise one of these devices, the Leap Motion, in order to facilitate natural gesture communication – particularly gestures found in the British Sign Language (BSL). The use of this software will aid users who are deaf or hearing impaired in communicating as they would in real world scenarios via the BSL.

## 1.2 Objectives

The application should fulfil the following main objectives in order to achieve the goal described above:

* Record gestures for BSL letters in order to account for each user’s own individual communication style.
* Accurately recognise the formation of these gestures and distinguish between them without false positives.
* Allow for recording and recognition of custom, user defined gestures to improve the scope of the system.
* Transform the recognised gesture into speech output.

## 1.3 Report Overview

The structure of the report is as follows:

Chapter 1 discusses the problem and the proposed solution, including its required properties. Chapter 2 analyses background research and related work as well as past applications that bear similarities to this project. Chapter 3 … Chapter 4… Chapter 5… Chapter 6… Finally, chapter 7…

**Chapter 2**

# Background

## 2.1 Research

As a key aspect of the project, it’s important to investigate the existing methods used to provide gesture recognition, whether just as a proof of concept or as part of a complete application.

The $P recognizer, as discussed by Wobbrock, et al. (2012) [1] is one such example of a recognition algorithm. As described in the paper, the $P is “a 2-D gesture recognizer designed for rapid prototyping of gesture-based user interfaces”. The $P aims to overcome the technically complex task of matching user gestures by instead treating them as groups or “clouds” of points and evaluating each one in turn. Even the simplest of gestures could be created in many different ways depending on the properties of its strokes e.g. start and end points, order or time, and direction. The use of a point cloud helps remove any ambiguity from the gesture which simplifies comparison and recognition. According to the paper, the algorithm requires only “70 lines of code” to function and delivered over 99% accuracy in user-dependant testing.

…

## 2.2 Related Work

Despite being a relatively new device, Leap Motion has seen numerous projects developed for it.

One such example is ‘LeapTrainer.js’, created by O’Leary in 2013 [2]. As described in the readme file on its GitHub repository, LeapTrainer is “A gesture and pose learning and recognition framework for the Leap Motion.” Developed in JavaScript, LeapTrainer allows users to create and store gestures. The software recognises a gesture using a custom template matching algorithm. The software supports both gestures and ‘poses’ – the difference being that a gesture is the movement of one or both hands at or above a specific velocity, whereas a pose is the stationary position of one or both hands over a period of time.

…

## 2.3 Similar Applications

A commercial application, UNI, is currently in development and is scheduled to be released in summer 2016 by MotionSavvy [3]. UNI utilises the Leap Motion in order to translate gestures into spoken text. As shown on the current site, there will be two models available to order – a base version including only the Leap Motion, and a ‘PRO’ version including the Leap Motion built into a custom designed tablet. The software is closed-source and based on a subscription model of $20 / month as well as an initial purchase price. The UNI appears to be similar to the idea expressed in the opening chapter – its closed nature and hence limited expandability may hinder its success, however. With no proof of concept or other form of demo available it is difficult to gauge how effective the final product will be.

….

**Chapter 3**

# Design

## 3.1 A

A

**Chapter 4**

# Implementation

## 4.1 A

A

**Chapter 5**

# The System in Operation

## 5.1 A

A

**Chapter 6**

# Testing and Evaluation

## 6.1 A

A

**Chapter 7**

# Conclusions

## 7.1 A

A

### **References**

[1] Wobbrock J. O., Anthony L. and Vatavu R.D. (2012) Gestures as Point Clouds: A $P Recognizer for User Interface Prototypes. On line publication, University of Washington, <http://faculty.washington.edu/wobbrock/pubs/icmi-12.pdf>. Last accessed 13 October 2015.

[2] O’Leary R. (2013) LeapTrainer.js. GitHub repository, <https://github.com/roboleary/LeapTrainer.js>. Last accessed 9 October 2015.

[3] MotionSavvy. (2015) UNI. Website, <http://www.motionsavvy.com/>. Last accessed 9 October 2015.