Glovesy

BY

Alan Devine - 17412402

Sean Moloney - 17477122

A functional specification document

As a requirement for CA400

Last Revision: 04/12/2020

Dublin City University (DCU)

PLAGIARISM DECLARATION

I/We declare that this material, which I/We now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work. I/We understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I/We engage in plagiarism, collusion, or copying. I/We have read and understood the Assignment Regulations. I/We have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study.

I/We have read and understood the referencing guidelines found at
`https://www.dcu.ie/info/regulations/plagiarism.shtml`,
`https://www4.dcu.ie/students/az/plagiarism`,
and/or recommended in the assignment guidelines.

Project Title	Glovesy
By	Alan Devine - 17412402
	Sean Moloney - 17477122
Field of Study	Computer Science
Project Advisor	David Sinclair
Academic Years	2020/2021

ABSTRACT

Glovesy is a wearable computer interfacing device in the form of a glove which will allow the user to interface with their computer by using custom macros or use the device for hand-tracking in VR or AR applications.
Keywords: : Wearables, human-computer interfacing, VR, AR, Arduino

GLOSSARY

1. VR Headset A head-mounted device for use in consuming Virtual Reality Content.
2. AR Headset Similarly to a VR Headset, an AR Headset is a head-mounted device which aims to provide a User with content that is built around their surroundings rather than replace them in the case of a VR Headset.
3. Arduino An Open-Source electronic prototyping platform.
4. ISR Interrupt Service Routine is a software process invoked by an interrupt request from a hardware device.

TABLE OF CONTENTS

LIST OF FIGURES

Introduction

1.1 Overview

Glovesy is an Arduino based wearable device which will allow the user to interface with their computer, either by using user-defined macros, which will be set up using our program, the Glovesy Configuration Suite, which will allow several gestures do be defined to certain actions within the PC or by allowing the user accurate hand and finger tracking for use in Virtual and Augmented Reality.

Gestures will be defined by a user and mapped to some action on their PC. Actions could include entering a combination of keystrokes, opening an application, raising/ lowering system volume, etc. They will be executed upon the user repeating their chosen gesture.

General Description

2.1 Product / System Functions

Glovesy is a wearable human/computer interfacing device, which will allow users to interact with their PC in several different ways, for different scopes. The primary focus of the device will be to allow the user to set up macros, or certain movements or gestures, which the computer will recognise as a specific command, thereby allowing ease of use. Another function of the device will be to track user hand and finger movements for increased accuracy and control in VR applications, since the device is so low profile, as opposed to current VR controllers which tend to be bulky, handheld devices.

2.2 User Characteristics and Objectives

Glovesy is designed for users who engage in VR/AR applications. This includes, but is not limited to, gaming, product prototyping across remote teams, creating virtual art, etc. Glovesy is designed to be a standard input device, we also envision it being used in environments such as, a presenter navigating a stage, users who for whatever reason may not feel comfortable/can use traditional input devices, etc. We hope to make Glovesy as accessible as possible to the vast majority of users, that being said, here is our idea of the "Ideal User".

- Experience with gesture-based input devices
  - e.g. Nintendo Wii, or an Air mouse.
- Experience with defining Macros

– Keyboard and Mouse Combinations that are present with devices with extra additional keys/ buttons.

2.3 Operational Scenarios

Table 1: Gaming in VR

Use Case ID:	1
Use Case Name:	Gaming in a VR Setting
Actors	User
Description:	The User would see a virtual representation of their hand (this is entirely dependent on the game itself). The user would be able to interact with their surroundings using Gloves rather than a traditional controller.
Preconditions:	1. The User has Connected Glovesy to their PC. 2. The User has opened a VR game.
Postconditions:	The User can use Glovesy as a stand-in for a traditional controller.
Trigger:	The User Opens a VR Game.
Normal flow:	1. The User loads a level; 2. The User interacts with their surroundings using Glovesy;
Exceptions:	In the case that the User opens a VR title which does not track individual fingers, the User has mapped a keypress to an individual finger.

Table 2: Use case detail

Use Case ID:	2
Use Case Name:	Mouse Emulation
Actors	User Host Operating System
Description:	The User controls their mouse cursor by moving their hand in a way similar to the WiiMote.
Preconditions:	1. The User has Connected Glovesy to their PC. 2. The User has enabled Mouse Emulation mode. 3. The User has defined a threshold for click actuation.
Postconditions:	The User can control the Mouse Cursor using their hand.
Trigger:	The User Enables Mouse Emulation Mode.
Normal flow:	1. The User Moves their hand along the YZ plane; 2. The User positions Mouse Cursor relevant item; 3. The User curls their index finger past the predefined threshold to initiate a click;

Table 3: Recording a Macro

Use Case ID:	3
Use Case Name:	Record Gesture Macro
Actors	User Glovesy Configuration Suite Host Operating System

Table 3 – Continued on next page

Table 3 – Continued from previous page

Description:	The User records a sequence of hand movements and maps them to take a certain action.
Preconditions:	1. The User has Connected Glovesy to their PC.
Postconditions:	Macro is added to Macro Database.
Trigger:	The User Clicks ”Record” in Glovesy Configuration Suite.
Normal flow:	1. The User performs several distinct actions while recording is active; 2. Relevant data is extracted from the Input Stream and places them into an input queue; 3. A MongoDB ”Collection” is created; 4. A MongoDB ”Document” is created for each of the items in the queue; 5. Each of the MongoDB ”Documents” are added to the Collection; 6. The User is notified that their Macro has been recorded.

Table 4: Recognise Gesture

Use Case ID:	4
Use Case Name:	Recognise Gesture
Actors	User Recogniser Daemon Host Operating System

Table 4 – Continued on next page

Table 4 – Continued from previous page

Description:	A sequence of movements is parsed and an attempt to match is made.
Preconditions:	1. The User has Connected Glovesy to their PC. 2. The User has recorded a Macro.
Postconditions:	An Action defined by the User is Executed.
Trigger:	The User has enabled Macros.
Normal flow:	1. The User performs some sequence of Movements; 2. Glovesy sends finger and positional data to the Host Machine; 3. The Recogniser Daemon extracts Relevant data from the Input Stream and places them into an input queue; 4. The first item in the input queue will be searched for in the first level of the Macro Tree; 5. The subsequent actions in the input queue will be searched for in the first level following the tree, with the Head being redefined upon every success. 6. Upon reaching a terminal leaf, the action defined by the user in the macro will be executed.

Table 4 – Continued on next page

Table 4 – Continued from previous page

Alternative flows:	4.a If no match is found on the first item of the input queue, it will be removed from the list and Step 4 is repeated using the next item as the Head. 5.a If no match is found in the subsequent searches, the Head item in the input queue is removed from the queue and Step 4 is repeated using the next item as the Head.
Notes and issues:	Relevant Data means any change in state, such as a rapid change in Acceleration, a change in direction, a change in finger position, etc.

2.4 Constraints

There are several constraints that we foresee will have some impact on the development process of this project.

- Distinguishing Gestures: It may be challenging to distinguish hand movements between general movement and purposeful gestures.
- Application Support: It could be difficult to set up programs to use the device, as, particularly in games, there may be different controls that are pre-defined.
- Cross-platform: a different driver will need to be made for different operating systems, which could be challenging.

Functional Requirements

3.1 Data Transfer

The device must be able to transfer data to the PC reliably.

- Criticality This Aspect is critical, since, even if the rest of the data is accurately measured, if the data isn't transferred reliably, the device is useless, meaning the entire functionality of the device is dependent on this.
- Technical Issues: One of the issues with this will be getting the Bluetooth module on the board we are using to be reliable, otherwise we can use the device with

The device must be able to track the overall hand's orientation and position in 3D space.

3.2 Hand Tracking

- Criticality: This aspect is critical to the overall accuracy of the device, especially for tracking the hand for VR and Gesture Recognition, as well as for general use, as it can be very frustrating to move a mouse cursor with a device that doesn't accurately read user inputs.
- Technical Issues: A significant issue with this is that the accuracy is almost entirely dependent on the hardware, meaning if the IMU device we use is inaccurate, we won't be able to do anything to improve the tracking.

3.3 Finger Tracking

The device must be able to track the fingers' joints accurately.

- Criticality: This aspect is also critical, as it is one of the main functions of the device.
- Technical Issues: One of the main issues with this is that the flex sensors that we are using are made by us, so there will need to be some calibration when first using the device.

3.4 Gesture Recognition

The system must be able to distinguish between random movements and intentional gestures.

- Criticality: This requirement is important for a specific use-case scenario which is not focused on gaming or VR, but rather on workflow.
- Technical Issues: Due to the nature of the device, we only have a certain number of inputs, and thereby limited on the amount of information we get from the device, meaning that we will have to focus on some way to distinguish between gestures and non-gestures through pattern recognition.

System Architecture

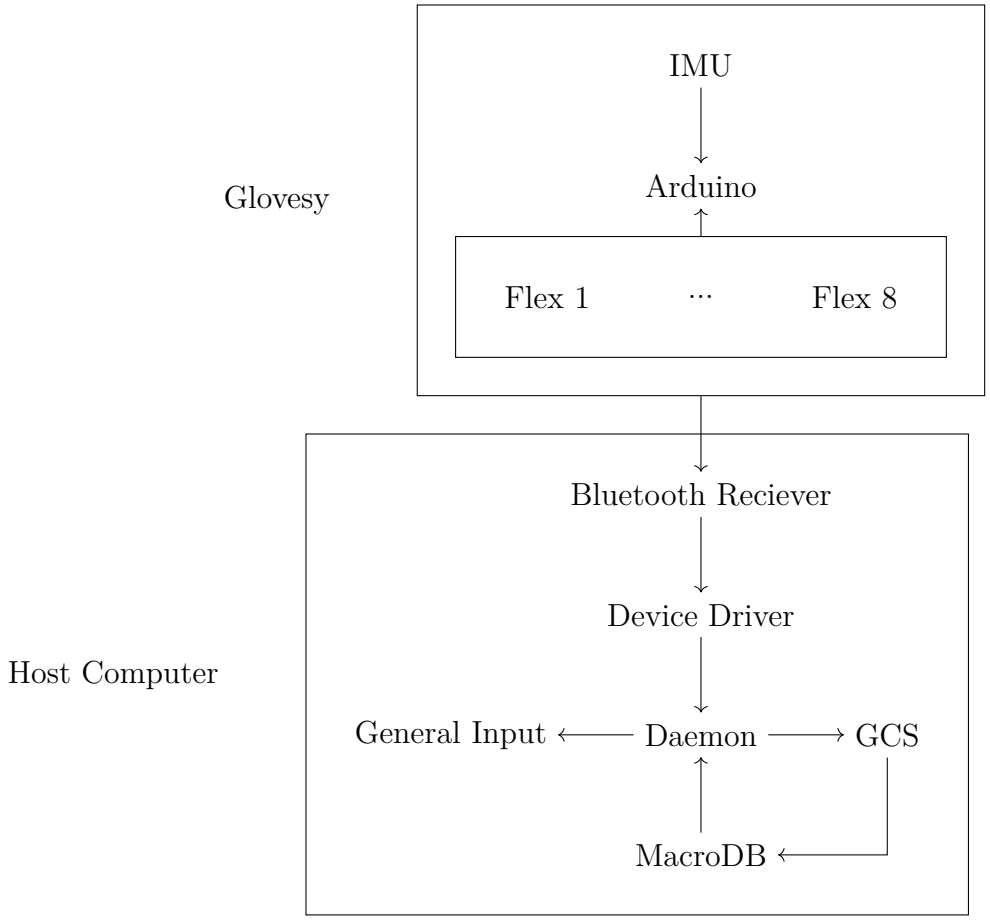


Figure 1: High-Level System Architecture Diagram

4.1 Arduino

The Arduino in use is an *Adafruit Feather M0 Bluefruit*. This has an integrated Bluetooth controller which allows us to transfer data to and from the host computer. This Arduino has 10 analogue pins which we can use to measure the resistance on our flex sensors.

4.2 Flex sensor Array

A flex sensor is a variable resistor that changes resistance based on how much the sensor is bent. For this project, we will be making use of 8 flex sensors which will measure the position of each knuckle as well as each finger joint. Except for the Thumb and pinky fingers, which will only have one flex sensor each. To help mitigate the costs associated with this project, we will be constructing our flex sensors using several layers of velostat and conductive thread which will act as our positive and negative terminals.

4.3 IMU

The IMU, or Inertial measurement unit is the device which we will use to measure the device in 3D space. Our device in question measures Acceleration and Gyroscopic data on the X, Y and Z Axises in metres per second and radians respectively. Our IMU also measures temperature, however, that will not be used for this project.

4.4 Macro Database

User-defined macros will be stored in a database implemented in MongoDB. We chose MongoDB as conceptually, it makes more sense to group these actions as an object rather than in a relational database.

High-Level Design

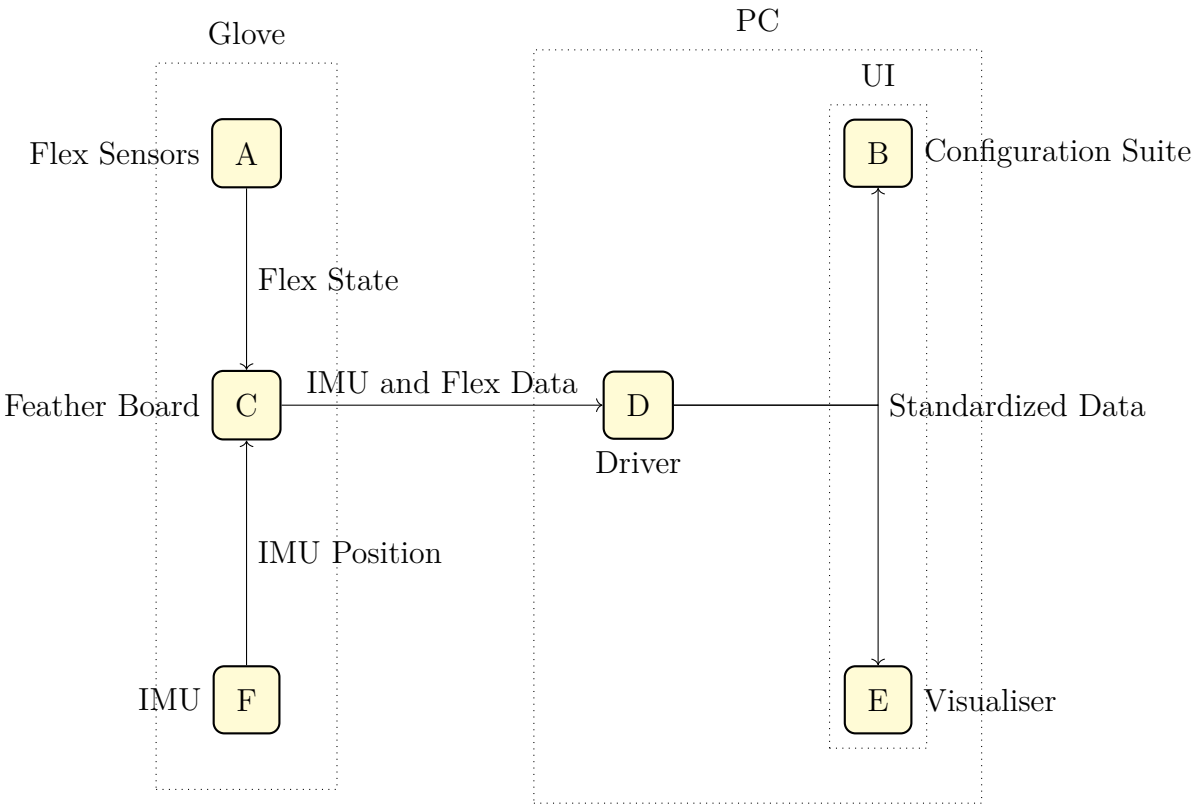


Figure 2: Data Flow Diagram

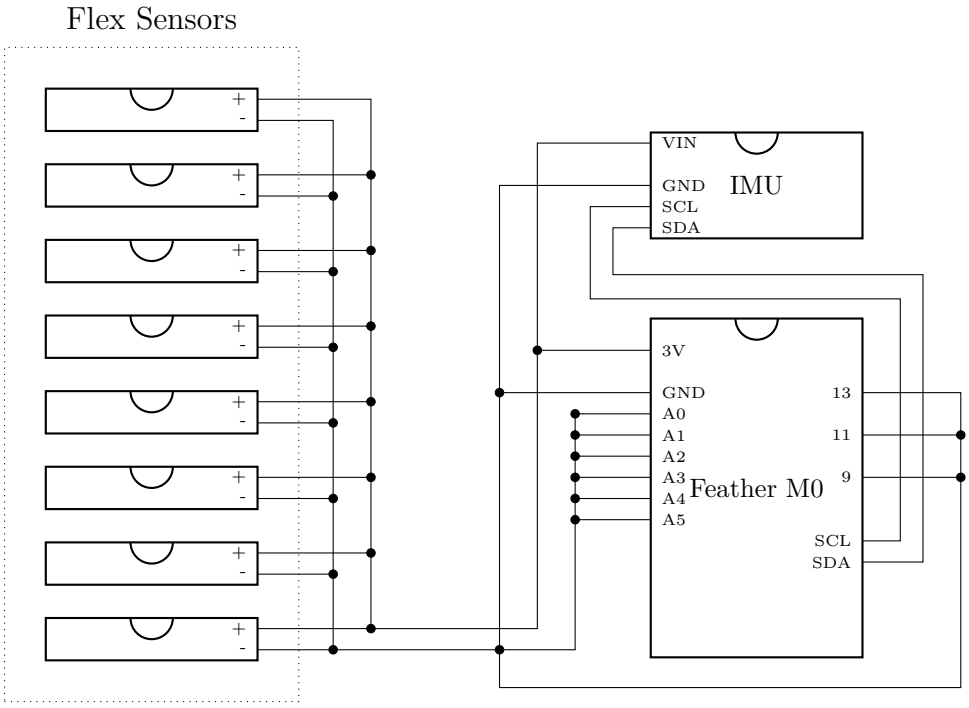


Figure 3: Circuit Diagram

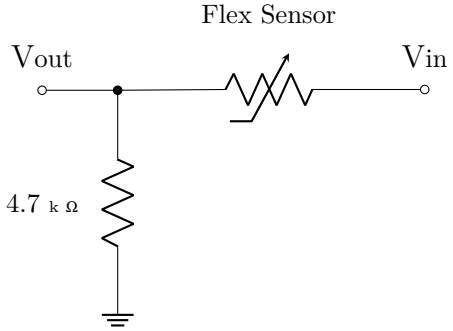


Figure 4: Flex Sensor Circuit

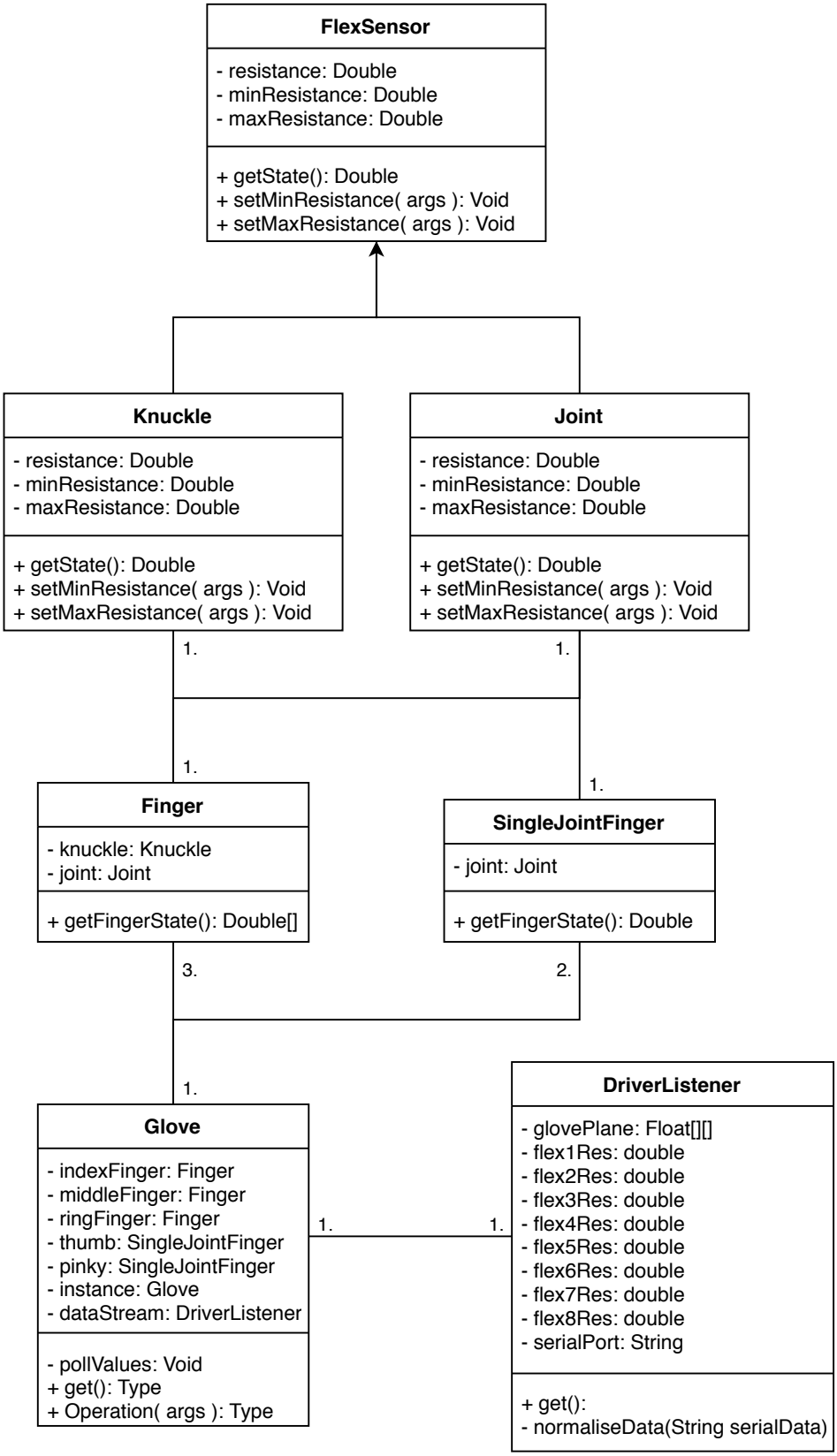


Figure 5: UML Class Diagram

The task calls the device driver via a Kernel device call, the driver receives the message from the queue and returns the data to the task and the buffer to the memory partition. When new data arrives, the ISR[0] allocates a buffer from the memory partition, puts the data in the buffer, and puts the pointer in a message which is then sent to the queue.

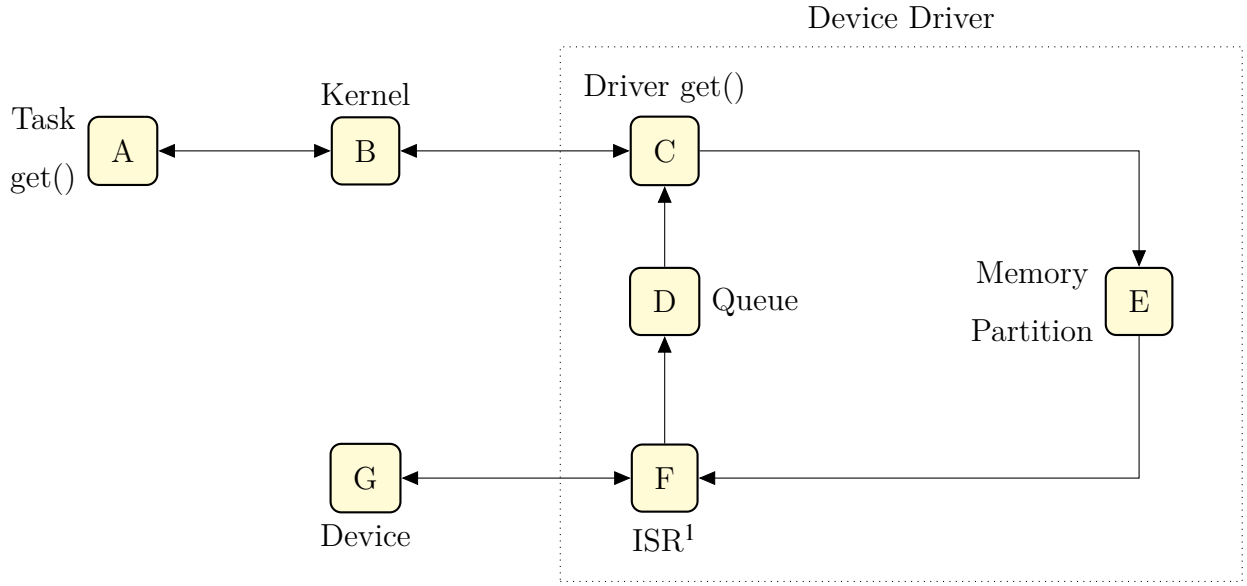


Figure 6: Device Driver

This is an example of how Gesture Recognition will be accomplished using the example of recognising a ”Hand Wave” Gesture.

A ”Hand Wave” is defined with the following steps.

1. Acceleration on the Z axis with a 90°rotation on the Y axis.
2. Decrease in Acceleration on the Z-Axis.
3. (Acceleration | Deceleration) on Y-Axis with slight deceleration on the Z-Axis.
4. Opposed (Acceleration | Deceleration) to step 3 on Y-axis with slight acceleration on Z-Axis
5. (Go to step 3 | Deceleration on Z-Axis)

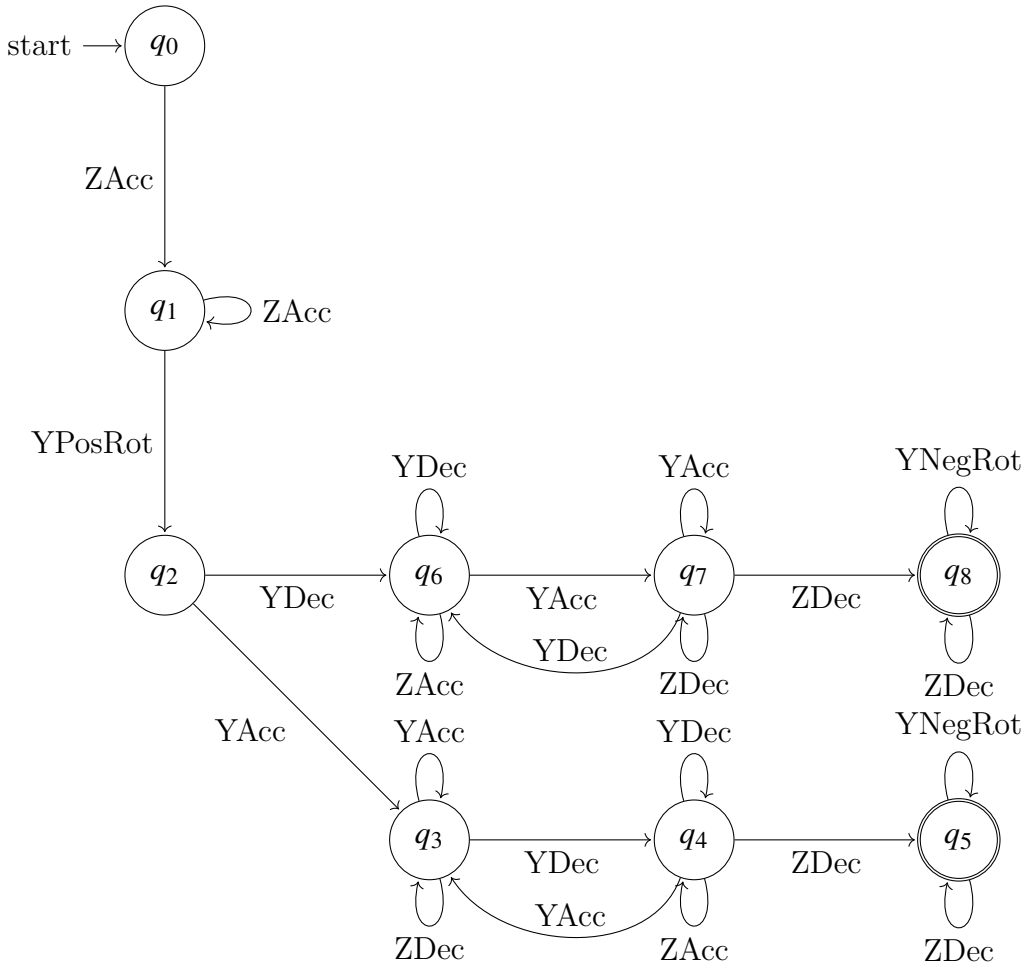


Figure 7: Example of Gesture Recognition

Preliminary Schedule

6.0 Major Tasks

The Major tasks of this project are listed below. They are ordered from most critical to least and their Dependencies on other tasks are also provided.

Task ID	Task	Dependency
1	Construct Glove	n/a
2	Develop Linux Driver	1
3	Develop OpenGL Environment	n/a
4	Implement Gesture Recognition	1, 2
5	Application Handler	n/a

6.1 Hardware Requirements

See below a table of the required hardware to complete this project.

Name	Description
Adafruit Feather M0 Bluefruit LE	Arduino board with built in Bluetooth
Adafruit LSM6DS33 + LIS3MDL	9-DoF IMU
Pressure-Sensitive Conductive Sheet	Velostat for Flex Sensors
Stainless Thin Conductive Yarn	30ft of conductive thread
Adafruit Flex Perma-Proto	Flexable Bread board

The above hardware was ordered upon receiving project approval.

6.2 Software/ Wetware Requirements

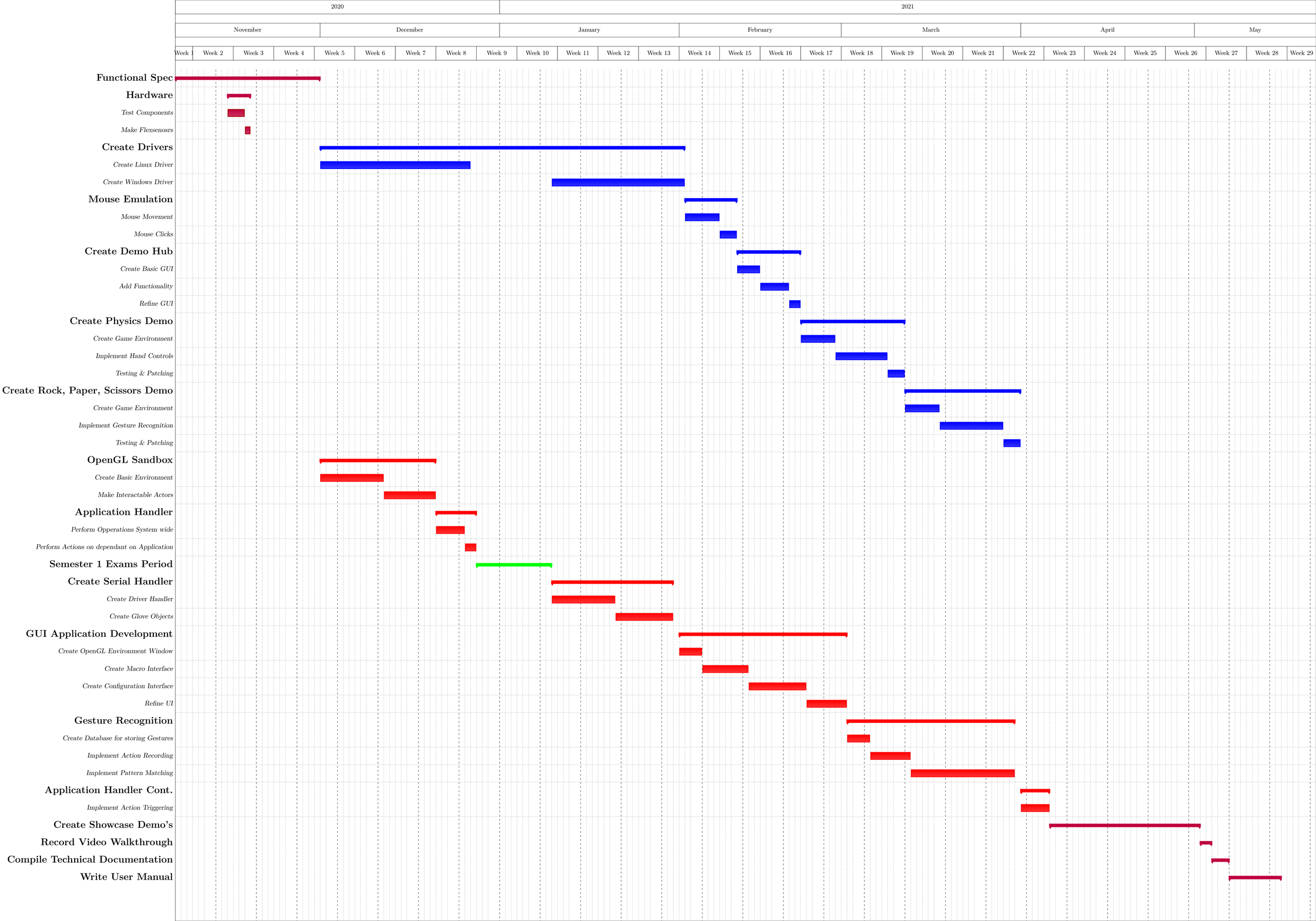
There are no specific Software/ Wetware requirements to complete this project.

6.3 Gantt Chart

In this section we will outline our provisional Gantt chart depicting how work will be completed through the duration of this project. The document is laid out as follows:

- Items which are Blue will be completed by Sean Moloney.
- Items which are Red will be completed by Alan Devine.
- Items which are purple will be completed by both of us.

Note: Time to write unit tests is factored in with each development task presented below.


2020
2021
November
December
January
February
March
April
May
Week 1
Week 2
Week 3
Week 4
Week 5
Week 6
Week 7
Week 8
Week 9
Week 10
Week 11
Week 12
Week 13
Week 14
Week 15
Week 16
Week 17
Week 18
Week 19
Week 20
Week 21
Week 22
Week 23
Week 24
Week 25
Week 26
Week 27
Week 28
Week 29
Functional Spec
Hardware
Test Components
Make Flexsenosrs
Create Drivers
Create Linux Driver
Create Windows Driver
Mouse Emulation
Mouse Movement
Mouse Clicks
Create Demo Hub
Create Basic GUI
Add Functionality
Refine GUI
Create Physics Demo
Create Game Environment
Implement Hand Controls
Testing & Patching
Create Rock, Paper, Scissors Demo
Create Game Environment
Implement Gesture Recognition
Testing & Patching
OpenGL Sandbox
Create Basic Environment
Make Interactable Actors
Application Handler
Perform Opperations System wide
Perform Actions on dependant on Application
Semester 1 Exams Period
Create Serial Handler
Create Driver Handler
Create Glove Objects
GUI Application Development
Create OpenGL Environment Window
Create Macro Interface
Create Configuration Interface
Refine UI
Gesture Recognition
Create Database for storing Gestures
Implement Action Recording
Implement Pattern Matching
Application Handler Cont.
Implement Action Triggering
Create Showcase Demo's
Record Video Walkthrough
Compile Technical Documentation
Write User Manual