**A logo for a university

Description automatically generated**

**Title: *Virtual Mouse Using Gesture Recognition with Mediapipe and OpenCV***

**By**

***Artur Walach***

***Course: BSc Computer Science***

**Project code: *PJE40 (for an engineering project, PJR40 for a research project or PJS40 for a study project***

**Supervisor: Dr *Jiacheng Tan***

**May 2024**

**Word count: 11,819**

**(should not include title page, acknowledgements, abstract, table of contents, table of tables, table of figures, reference list or appendices)**

Please tick

|  |  |
| --- | --- |
| ❏ | I give permission for my project to be published in the University library and/or be made available to other students as examples of previous work. (optional). |
| 🗹 | I confirm that I have read and understood the University Rules in respect of plagiarism and student misconduct. |
| 🗹 | I declare that this work is entirely my own. Each quotation or contribution cited from other work is fully referenced. |
|  |  |

Date: \_14/05/24\_\_\_\_\_\_\_\_

**Acknowledgements**

I would like to express my gratitude towards my supervisor Dr Jiacheng Tan for assisting me with any issues that occurred in during the process of this project. I want to thank my family for supporting me throughout the final months of this project. I also express my thanks to the University of Portsmouth staff that assisted me with problems that occurred during this project due to my health issues.

**Abstract**

Computer vision is a widely researched field, with many different applications. It is used in scenarios such as, surveillance cameras, smartphones, CT and MRI scans, and many more. The aim of this project is to explore a part of computer vision. To develop an application in the topic of gesture recognition and document the development process. I will be developing an application that will act as a virtual mouse. It will make use of the user’s camera, recognise the hand, and track the position of the gestures the user makes to perform different functions like mouse tracking, a left click, right click function and a scroll function. Each of them will be executed by doing various gestures, a pinch gesture with space in-between, a normal pinch gesture, pinch gesture with the middle finger and thumb and a pinch gesture using the left hand respectively. To develop this solution, OpenCV will be used to access the camera, Mediapipe for hand recognition and getting data about its landmarks, and the whole application will be programmed using Python.

**Table of Contents**

[Chapter 1: Introduction 7](#_Toc167040151)

[1.1 Virtual Mouse using Gesture Recognition 7](#_Toc167040152)

[1.2 Project aim and objectives 8](#_Toc167040153)

[1.3 Project constraints 8](#_Toc167040154)

[1.4 Log of risks 9](#_Toc167040155)

[1.5 Project deliverables 10](#_Toc167040156)

[1.6 Project Approach 10](#_Toc167040157)

[1.6.1 Project Management 10](#_Toc167040158)

[1.6.2 Research 10](#_Toc167040159)

[1.7 Legal, ethical, professional, social issues (mandatory) 10](#_Toc167040160)

[1.8 The report 11](#_Toc167040161)

[Chapter 2: Literature Review 12](#_Toc167040162)

[2.1 Introduction 12](#_Toc167040163)

[2.2 Research Strategy 12](#_Toc167040164)

[2.3 What is Gesture Recognition 12](#_Toc167040165)

[2.4 Gesture Recognition Structure 13](#_Toc167040166)

[2.5 How to Perform Gesture Recognition 13](#_Toc167040167)

[2.6 Existing Software 15](#_Toc167040168)

[2.7 Summary 16](#_Toc167040169)

[Chapter 3: Requirements Capture (Analysis) 17](#_Toc167040170)

[3.1 Introduction 17](#_Toc167040171)

[3.2 Functional Requirements 17](#_Toc167040172)

[3.3 Non-Functional Requirements 18](#_Toc167040173)

[3.4 Summary 18](#_Toc167040174)

[Chapter 4: Design 19](#_Toc167040175)

[4.1 Introduction 19](#_Toc167040176)

[4.2 Methodology 19](#_Toc167040177)

[4.2.1 Mediapipe 19](#_Toc167040178)

[4.2.2 OpenCV 19](#_Toc167040179)

[4.2.3 PyAutoGUI 20](#_Toc167040180)

[4.3 Gestures/Functions 20](#_Toc167040181)

[4.3.1 Cursor Movement 20](#_Toc167040182)

[4.3.2 Left Click 20](#_Toc167040183)

[4.3.3 Right Click 20](#_Toc167040184)

[4.3.4 Scrolling 20](#_Toc167040185)

[4.4 Processes Flowchart (Appendix I) 21](#_Toc167040186)

[4.5 Summary 21](#_Toc167040187)

[Chapter 5: Development 22](#_Toc167040188)

[5.1 Introduction 22](#_Toc167040189)

[5.2 Planning 22](#_Toc167040190)

[5.3 Development Cont. 23](#_Toc167040191)

[5.4 Code Description 24](#_Toc167040192)

[5.5 Encountered Problems 24](#_Toc167040193)

[5.6 Summary 26](#_Toc167040194)

[Chapter 6: Testing 27](#_Toc167040195)

[6.1 Introduction 27](#_Toc167040196)

[6.2 Test plan 27](#_Toc167040197)

[6.3 Results 28](#_Toc167040198)

[6.4 Summary 29](#_Toc167040199)

[Chapter 7: Evaluation 30](#_Toc167040200)

[7.1 What was achieved 30](#_Toc167040201)

[7.1.1 Functional 30](#_Toc167040202)

[7.1.2 Non-Functional 31](#_Toc167040203)

[Chapter 8: Conclusion 32](#_Toc167040204)

[8.1 Recommendations 32](#_Toc167040205)

[8.2 Aim & Objectives 33](#_Toc167040206)

[8.3 Risks (did any happen, was your mitigation ok?) 33](#_Toc167040207)

[8.4 Project Management 33](#_Toc167040208)

[8.5 Conclusion 33](#_Toc167040209)

[References 35](#_Toc167040210)

[Appendix A: Project Initiation Document (inc. Gantt Chart) 39](#_Toc167040211)

[Appendix B : Ethics Certificate 46](#_Toc167040225)

[Appendix C : “Main” Method Code 49](#_Toc167040227)

[Appendix D : Camera Initiation 50](#_Toc167040229)

[Appendix E : Hand Detection and Landmark Plotting Code 51](#_Toc167040231)

[Appendix F : Handedness Check 52](#_Toc167040233)

[Appendix G : Switch Case Statement for Gesture Functions 53](#_Toc167040235)

[Appendix H : Gestures Code 54](#_Toc167040237)

[Appendix I : Processes Flow Chart Diagram 55](#_Toc167040239)

**Table of Tables**

[Table 1: Log of risks for the project 9](#_Toc167041449)

[Table 2: Testing results 28](#_Toc167041450)

[Table 3: Table of whether the functional requirements were achieved. 30](#_Toc167041451)

[Table 4: Table of whether the non-functional requirements were achieved. 31](#_Toc167041452)

# Chapter 1: Introduction

## Virtual Mouse using Gesture Recognition

Computer vision can be a solution to many different problems. It allows computers to interact with the real world and vice versa. It can, for example, help people with physical impairments to interact with a computer. Computer vision is used everywhere in the outside world. It is used from surveillance cameras to modern day smartphones for face detection. It has its uses in healthcare environments such as in CT and MRI scans. Computer vision works by recognising landmarks using many (often thousands) diverse images of different situations. For example, if you were building a model to recognise different gestures, you would need hundreds or thousands of images of the desired gestures from different angles, lighting conditions and skin colours, if you want the model to be accurate. According to A. Rosenfeld (1988), to process an image in that model, it firstly needs to be converted into a digital image, which is an array of different numbers that correspond to the brightness and colour values of various pixels.

This paper will document the development of my project which is focused on a single area of computer vision, gesture recognition. We use gestures everyday when communicating with others. Around 55% of human-to-human communication is conveyed through gestures, compared to 45% verbally (D. Phutela, 2015). it can be a useful tool for solving various problems. Gesture recognition is about recognising human expressions of motion which would include hands, arms, face, head, and/or body. This project’s emphasis will be on hand related gesture recognition. For human-computer interaction to be effective, it needs a reliable and accurate gesture recognition model (S. Mitra and T. Acharya, 2007). This can act as an alternative for using a mouse and keyboard for interaction with a computer. The problem that I aim to solve is to assist people who cannot easily use a mouse. There are multiple difficulties, and challenges that come with developing an application for such a problem. Some of those include figuring out how to track the hand, i.e. what kind of method to use. Deciding which method is the most optimal when you factor in their ease of use,

Hand gestures can be recognised using different methods from turning the user’s camera feed into binary images (images where the background is black, and the foreground is white, isolating the hand from the background), colour segmentation (recognising the hand by isolating the user’s skin colour), to using a model for gesture recognition, made up of thousands of different various images. The last part is what this project will make use of. The project will use the Mediapipe framework to isolate the hand from the background and recognise its features. It can be used in environments such as web pages, Android and iOS. Furthermore, it supports languages such as C++, Python, JS, and Coral (Han et al., 2022). Mediapipe is a framework that allows for the development of solutions for work with arbitrary sensory data. It can be used for hand tracking and gesture recognition. It is a model that makes use of machine learning as a pipeline to detect and track (Lugaresi et al., 2019). After detection, Mediapipe uses a method of mapping 21 landmarks to define 3D coordinates to either display joints of the hand or/and obtain data. Its tracking features can not only be used on desktop computers, but also on mobile phones providing accessibility to a wider user range (Han et al., 2022).

As to why I chose this direction for this project it’s because, being well acquainted with hospitals in the past, there are a lot of touch screens used for things such as registering appointments. Hospitals are hot spots of different bacteria and viruses. They are often attended by immuno-compromised patients. Touch screens such as those mentioned before, are used by a lot of different people which unfortunately promotes the spread of bacteria. Although the deliverable of this project is not supposed to be used in hospitals, it is where the idea of this project came from.

## Project aim and objectives

I devised the aims and objectives for this project by looking at already existing solutions to this problem. They gave me a vague idea of what to develop and the expectations for the given timeframe. The overall aim of this project is to develop a virtual mouse application that will help the user with navigating their computer with gesture recognition by using their hands. The ultimate aim is to achieve all of the requirements set below, however, the most probable outcome is that not all of them will be completed. This is because some parts of the project could turn out to be more difficult than expected, resulting in them not being able to be completed when you factor in the given time frame for this project. Other limitations or constraints, could include factors outside of my control, such as personal problems outside of this project, or even device failure etc.

## Project constraints

Every project has its own limitations that the developer has to be aware of. Some of those would include:

* Scope of the project – this would refer to how large the project is. The changes to the scope can impact other constraints as well, such as time or cost.
* Time – this is the timeline of the project. It refers to deadlines, milestones, and the overall duration.
* Cost – this could involve any equipment a project is making use of. This project will need a webcam for the deliverable to work, however, the majority of laptops already come with one therefore there is no need to spend additional money.
* Resources – this would also refer to the equipment needed for the project. This project will need a webcam, therefore the developer and the user need one for the deliverable to operate.

## Log of risks

Table 1: Log of risks for the project

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Description | Likelihood (high, medium, low) | Impact | Mitigation/Avoidance |
| 1 | Laptop/desktop failure | Medium | Data loss and delays to the project | Backup on a USB or a cloud environment. |
| 2 | Scale of the project being too big | Medium | Would cause the project to be incomplete on deadline | Carefully consider the scale of the project and/or drop less important features as the project goes on |
| 3 | Accuracy of the gesture recognition | High | Would cause the final program to recognise gestures inaccurately and make it unintuitive to use | Collect a diverse dataset that would include pictures of intuitive gestures and fine-tune the system to improve its accuracy |
| 4 | User variation | High | The program could have issues with recognising gestures from people with different hand shapes, sizes and skin tones | Include data and perform testing on a diverse group of users |
| 5 | Environment variation | High | The program’s performance may be affected by differing lighting conditions and clutter in the background of the image reducing its accuracy | Include data and perform testing in varied lighting conditions and different environments |

## Project deliverables

The only deliverables of this project will be this report and the final application alongside the code and a manual on how to use the application. The application will be a program that uses the user’s camera to detect their hand and translate gestures into what works as a virtual mouse. It will have the functions of a normal computer mouse, such as a left click, right-click, cursor tracking, and scrolling.

## 1.6 Project Approach

This section will detail what is the approach for this project. This refers to aspects such as, how I will go about developing the application, documenting it, and what is needed to be researched.

## 1.6.1 Project Management

The plan is to first spend a few weeks on researching how to develop what is required. After that, dedicate a couple/few months or weeks for the coding part of the project. Every time a feature is implemented, it will be tested to ensure it works and does what it’s intended to do. During the coding sections, any issues will be documented. Unless a feature of the application is important, too much time will not be spent on it if it cannot be fixed easily and will be cast aside to make time for more important features.

## 1.6.2 Research

Research will be conducted to learn how to do certain things, e.g., coding using Mediapipe and OpenCV, how to code regarding gesture recognition, and how to implement the required features. Most of those will be studied through already existing documentation on the internet. Additional research that will be performed, is to look at already existing software that is similar to this project’s final deliverable. This will give inspiration on what features should or could be included in the application and will help to determine whether they are feasible or not.

## Legal, ethical, professional, social issues (mandatory)

There are no explicit professional issues in this project, however, there are some legal and ethical issues to be addressed. The final application will need to access the user’s webcam for it to work, therefore the legal and ethical issues would be if any data about the user is stored from the camera feed. However, this is not a concern, because this project will not store any information about the user from the camera feed or any other source. The only information used is from other parties such as the Mediapipe gesture recognition model. That model includes hundreds of photographs of features that aid in gesture recognition, such as hands, facial features etc., but Mediapipe is open-source and free to use by anyone for whatever reason they want to use it for.

## The report

The report will include the following:

* **Literature Review –** The purpose of it will be to research existing methods used for gesture recognition, compare them, and pick the most appropriate one for this project.
* **Requirements Capture –** this section will outline the requirements made for the application of this project.
* **Design –** This section outlines how the final application is designed.
* **Development –** This section describes the development process of the application. How it was developed and what tools were used etc.
* **Testing –** This is the part in which results from the testing performed on the application are shown and explained.
* **Conclusion –** This is the last section of the report. It will describe if the application does or does not do what was initially planned, how it could be improved in the future, and some concluding remarks.

# Chapter 2: Literature Review

## Introduction

The intent of this project is to develop a program that will assist the user with navigating a personal computer with the help of gesture recognition. The program will try to recognise the user’s different and unique hand gestures and translate them into commands that will perform certain tasks such as closing a website, window etc. The user should also be able to move the mouse cursor using their hand in front of a webcam. The purpose of this literature review is to present the general idea of what gesture recognition is, how to go about developing it and its design methods, the existing solutions and software, and the testing methodologies that will be used during development and for the final product, while referring to already existing studies and research in this field

## Research Strategy

This literature review will look at different, already existing research papers that tell what gesture recognition is and how to go about doing it and compare different methods. It will also look at topics such as testing strategies and already pre-existing solutions for the problem of this project, to gain inspiration on how the final application will be structured.

## What is Gesture Recognition

Gesture recognition refers to recognising different expressions made by a human through the use of motion involving the face, arms, hands, head, and even the whole body with the intent of conveying meaningful information, although this project will only focus on making use of the user’s hands and fingers. For gesture recognition to work, there are four main steps: image acquisition, Hand segmentation, Feature extraction, and Gesture recognition (Ahuja & Singh, 2015). Image acquisition defines a process for capturing images used for a vision-based approach. Hand segmentation refers to the process that isolates hands and their features from the background. Feature extraction is a process of retrieving features from the captured images (Suharjito et al., 2018). Classification, which refers to gesture recognition, is a crucial and final stage (Ghosh & Ari, 2016).

There are many different applications of gesture recognition such as allowing very young children to interact with different devices, helping physically impaired people with navigating a computer, reading sign language (Rosenfeld, 1988), and during medical operations (Mitra & Acharya, 2007), or just making the experience of navigating a virtual environment more convenient for the user. Although there are many uses for gesture recognition, different factors can negatively impact its performance. Those include changes in lighting, difficult backgrounds, camera movement, and user-specific changes (Fang et al., 2007).

## Gesture Recognition Structure

There are two main types of gesture recognition:

* Offline Gestures: these gestures are processed after the user’s interaction with the program (Mitra & Acharya, 2007) i.e., static gestures.
* Online Gestures: These gestures are analysed in real time i.e., dynamic gestures.

Gestures can be static or dynamic. Dynamic gestures are more complex but have more real-life applications (Sung et al., 2021). In this project, both static and dynamic gestures will be used. This is because dynamic gestures, although they are more complex, are better for the purpose of this project as they would allow the user to, for example, move the cursor of their computer using their hand. Static gestures will be used for operations such as, opening and closing something, for example opening and closing a website in a browser.

According to S. Mitra and T. Acharya (2016), recognising the meaning of a gesture can be dependent on the:

* Spatial information – where it occurs.
* Pathic information – the path it takes.
* Symbolic information – the sign it makes.
* Affective information – its emotional quality

## How to Perform Gesture Recognition

First, a set of data must be collected for gesture recognition software to work. The type of data depends on what the purpose of the application is. This could differ from facial recognition to gesture recognition, which is the aim of this project. The dataset would contain images of the required hand gestures taken using a camera such as a webcam or a normal photo camera. Captured real-life images are so noisy that they require a high amount of pre-processing to work on the other hand, existing datasets are often so noiseless that they would train the application to only recognise gestures in a perfect or near-perfect environment, which is not practical for real-life scenarios. Therefore, it is important to capture images that do not have too much noise but that are also not too clean to develop a model that can handle noisy images and produce desirable results (Sharma et al., 2020).

Hand gesture recognition normally involves steps such as image acquisition, pre-processing, feature extraction, and gesture recognition. Image acquisition involves capturing an image of the user’s hand using a webcam. The image goes through pre-processing which normally involves colour filtering, smoothing, and thresholding (Ismail et al., 2021). Colour filtering would involve turning an RGB image into grayscale. Smoothing involves removing or reducing the noise from the image and thresholding is about turning the cleaned, grayscale image into a binary image. Thresholding is used to transform the grayscale image into a binary image. This technique compares each of the pixel’s intensity values and compares them with the preset threshold. If the intensity is bigger than the threshold, the pixel in the grayscale image is replaced with a white pixel, if the intensity is smaller than the threshold, then the pixel is replaced with a black pixel, according to (Haria et al., 2017).

First, the hand of the user needs to be located in the real-time footage from a webcam. This can be difficult due to the differences in pose, orientation, location, and scale (Ismail et al., 2021). Then the segmentation process begins. It involves dividing the input image of the hand gesture into regions separated by boundaries. This process differs on the type of the input gesture, if it is dynamic, then the gesture first needs to be tracked and located before segmentation. However, if it is static, the other steps can be omitted to go straight into segmentation. (Khan & Ibraheem, 2012) One of the ways for the application to detect the user’s hand would be to use colour segmentation to detect the skin of a hand. (Oudah et al., 2020) This works by comparing colours of every pixel with the colour of the neighbouring pixels or a preset classifier to divide an image into colour regions of the desired colour (Sharma et al., 2012). The image could also be normalised. Normalisation is the process of changing the intensity of an image’s pixels. This is done to negate the effects of glares, poor contrast of an image that would make it difficult to detect a gesture. It is also often necessary to magnify the area around the user’s hand to make recognition easier if the user is far away from the camera (Chandra et al., 2015).

Gesture recognition is often developed using MATLAB or Python. MATLAB would be a more straightforward choice for developing a gesture recognition system as it has already built-in features that help with the development. They include functions that can convert an image into greyscale, binary. Functions that reduce the noise in an image and clean it up to isolate the object or objects of interest in the image. The boundaries, centers, etc. can be detected as well. All of these will be useful in this project. However, python has many resources available and is a more general language, meaning it can be used for many different projects and purposes, although it can be more complex.

When using MATLAB, Shwetha K., Yewale et al. (2011) mentioned that you could first, use histogram equalisation for image pre-processing, detect the edges of the hand using canny edge detection, enlarge the edges by using dilation in order to achieve a continuous edge, filling the inside of the boundary, store the boundary of the hand in an array, perform a vectorization operation on every pixel of the hand, detect it’s fingertips and finally track the detected fingertips to understand the motion. The method shows that after histogram equalisation, the hand is much more recognisable than before. Before applying it, the spread of the intensity of the pixels in the graph was very low, meaning that the hand blended into the background. After equalisation, the spread was much wider therefore the contrast between the hand and the background was much higher, which made the edges of the hand easier to detect.

Ahmed K. H. AlSaedi and Abbas H. AlAsadi (2020) describe their technique for gesture recognition using Python. First, from the captured, only the important area is chosen (a region of interest), then it is blurred using gaussian blur to reduce the amount of background objects. The hand is then separated from the background using background subtraction by first taking the image of the background then putting the hand in front of the background and calculating the absolute difference between the two images. Next the output from that subtraction will be thresholded so that the hand is white and the rest of the image (the background) is black. Next step would be to detect the boundary of the hand. Lastly, to determine the number of fingers that can help to recognise a gesture, a convex hull is used to determine extreme points. This is a group of points surrounding the hand. Draw a circle in the fingers, centre point of the hand and seventy percent of the length of the maximum euclidean distance between the centre and the extreme points which would determine the radius. Using the AND operation between the circle and the threshold image can help calculate the number of fingers. Each of the images would be processed in this way. The results of final testing of their method show that the accuracy is quite high. A fist gesture, one, two, three, four and five fingers up gestures were implemented and recognised using this method. The testing showed that every gesture was very accurate except for the fist and the five finger gestures which had a 90% accuracy, however the overall accuracy was 96.6%. Each gesture was tested ten times.

Another way of recognising the hand and gesture would be to use a mediapipe library. Mediapipe is a framework developed by Google that allows it to process different data such as video and audio. Those include but are not limited to posture estimation and face recognition (Kavana & Suma, 2022). For this project only the hand tracking feature will be used. This is possible due to the thousands of different images that make up the gesture recognition model of the framework. First step would be to initiate the webcam using OpenCV.  After that is to recognise the hand of the user using the mediapipe library, plot the points of interest on the hand, also known as landmarks, for ease of use. These points are plotted on all of the joints and tips of fingers, and on the wrist. Each of those points has its own ID which can be used to identify them and can be used in code to get the location of those points, which then further can be used for functions such as translating the location of a certain point in the camera feed and translating it on to a position on the user’s screen. Gestures can be used for different functions, for example, raising two fingers would initiate a mouse, and lowering one of them would correspond to a left or right mouse click. The user’s other free hand could be used to perform the remaining functions of a mouse such as scrolling. Pinching your index finger and thumb could be used for dragging a website. Also, other gestures could be used as shortcuts for e.g., opening a web browser or an email app. This could be done with the user’s other hand that is not in use.

A method similar to the one mentioned above was once done by Patil V. et al. (2023). They first captured the live feed of a webcam and used the Mediapipe gesture recognition model to recognise the hand and different gestures. In the beginning, the webcam is accessed, the image is processed, then the image is send to a gesture recognition system provided by Mediapipe. After that the unique gesture is recognised and one of the several functions are performed; play/pause, forward, rewind, volume up or down, or no action is performed. In their results they presented that all of the gestures were executed and processed correctly. This compared to the other methods mentioned in this literature review shows that this method is more accurate. Furthermore, from the looks of it, this method is simpler to perform than all of the previous methods, because the hand detection system does not have to be made from scratch.

## Existing Software

There are many already existing, advanced gesture recognition software. They range from recognising hand gestures, for e.g., sign language, to gesture recognition that helps with human-to-computer interaction, to just making a virtual mouse, which uses the real-life position of your hand captured through a webcam and translates it to a position on your screen. They often use other gestures in combination with that to recognise scrolling, left clicks and right clicks. Some existing software requires you to raise your first two fingers to initiate the mouse, moving your hand in that position moves the cursor on screen. Bringing down your middle and index finger works as a left and right mouse click respectively. Some software uses your other free hand to perform gestures that function as clicking, scrolling etc. Most of the pre-existing applications make use of Mediapipe, for gesture recognition, and OpenCV for accessing the camera of the user’s device.

There are also additional features that are included in the existing software that could be included in the final deliverable of this project. Some of them have features such as raising or lowering volume by using gestures, or a gesture for taking screenshots.

## Summary

There are many ways in which gesture recognition can be conducted. From using binarization to separate the hand from the background, to using pre-existing machine learning models such as Mediapipe. After conducting research for this literature review, I have decided to choose the method involving OpenCV and Mediapipe. This is because, it is the simplest method and the most accurate one, giving me more time to focus on the other aspects of the application. It is also easily integrated with Python, which is the language used for this project’s application, and readily accessible for anyone.

# Chapter 3: Requirements Capture (Analysis)

## Introduction

Setting requirements is a very fundamental step for managing a project. It helps with determining the scope and provides an outline of what features to focus on. This section will state the requirements of this project, both functional and non-functional, and analyse their importance to the overall project. Not every requirement will be equally important, for example, for a basic calculator it would be more important to implement addition than something like logarithms. This project will also have requirements that are more important than the others, and not necessarily all of them will be completed. The necessary requirements are the ones that need to be implemented for a virtual mouse to be a virtual mouse. This means functions such as cursor tracking, left-click, right-click and scrolling. The requirements mentioned below, were decided on by analysing and taking inspiration from some already existing software. I have decided to go about it this way because there are already many pre-existing virtual mouse programs on the internet. Majority of solutions to any problems have already been discovered, therefore, it is difficult to be original. This is one of the better ways as there is no explicit client for this project.

## Functional Requirements

* Detect the user’s hand through their camera (Must).
* Detect the important landmarks of the user’s hand (Must).
* Implement a function for cursor movement (Must).
  + Code a function to recognise when the user brings their index finger and thumb tips close together, without touching each other using their right hand (approximately between 50 to 100 pixels from each other or around 3-4cms).
* Implement a function for a left mouse click (Must).
  + Code a function to recognise when the user touches their index finger and thumb tip together using their right hand.
* Code a feature to recognise which of the user’s hands is left or right (Must)
* Implement a scrolling function (Must)
  + When the user touches their index and thumb fingertips together, and moves their hand in this position up and down, the program should scroll accordingly with the movement.
* Code a front-end GUI for the application (Could)
  + Implement a GUI for the application that could allow the user to choose aspects such as their dominant hand.
* Code additional gestures that can be used as shortcuts to open certain programs such as a browser or emails (Could).

Above are the requirements that describe features that are planned to be implemented in the final application. However, not all of them are equally important. The requirements with the “Must” annotation are features that have to be implemented, and the ones with “Could” are less important features that are not necessary for the functioning of the application, therefore they do not need to be implemented.

## Non-Functional Requirements

* The final application needs to respond to the user’s gestures in real-time or with menial delay (Must).
* The application should operate in high enough frame rate so that it can be used smoothly (Must).
* It should have high accuracy in recognising the user’s gestures (Must).
* The program should have minimal or no number of errors so that it is acceptable for everyday use (Must).
* If errors do exist, it should recover from them (Must).
* The system should work with existing digital environments such as a browser and the user’s desktop, and anything that doesn’t have its own mouse input (Must).
* The application should work on other systems apart from Widnows computers, such as, MacOS, mobile devices, etc. (Could)

## Summary

The purpose of this section was to show the different requirements made for the final application of this project. Also to help determine the scope of the final solution, which would aid in deciding on what aspects of the project to prioritise.

# Chapter 4: Design

## Introduction

This section outlines the application design part of this project. Desing is an important part of every project; it aids in deciding how the application will look like and/or function. As there is no GUI in the final deliverable, this section will mostly outline the design of the back end/code. It will also explain the methodology behind the application.

## Methodology

For this project, python programming language will be used. It is an easy to understand and concise language that supports I/O, numerics, images and plotting (Solem, 2012, p. ix). Python is one of the most widely used languages and has many various resources and documentation for different libraries and frameworks. OpenCV will be used for camera-related operations such as opening the camera so that the input from it can be used by other parts of the code. To recognise the hand and its landmarks, the input from the camera will be processed by a Mediapipe framework.

## Mediapipe

Human-computer interaction using gesture recognition, specifically hand recognition, can be extremely difficult to implement as it needs a reliable and robust model to work accurately. For this reason, the Mediapipe framework is used in this project. Mediapipe is an open-source framework made by Google for public use, of which the purpose is to recognise different aspects of gesture recognition. It can be used for object detection, hand recognition, facial recognition and many other different scenarios (Lugaresi et al., 2019). The Mediapipe model contains thousands of images in varying positions, lighting and skin colours for those purposes. It makes use of those images to recognise different landmarks of the hand (for this purpose). It recognises the joints/points of the hand and each one of them is assigned their own ID number (fig. 1) so that they can be easily accessed and manipulated. This can be used for operations, such as calculating the distance between different points, which then later can be manipulated to perform different functions. For example, touching your index fingertip and thumb tip can be translated into a left mouse click, or bringing your middle fingertip and thumb tip together could operate as a right mouse click, by calculating the distance between those points, and by writing code to execute those said functions if the distance is small enough. Therefore, it is much more accurate and feasible than developing a completely new model for this project. To make my own hand recognition model, hundreds, if not thousands, of images of different gestures in varying positions, angles, lighting conditions, etc. would be needed, which would not be very feasible in the given timeframe.

## OpenCV

OpenCV is an open-source library available for public use that aids with computer vision, machine learning, and image processing-related operations. It has a lot of different modules that aid with various computer vision problems. It can be used for cropping images, enhancing them by modifying brightness, sharpness and contrast, object detection etc. (S. Brahmbhatt, 2013, p. 3-4) It can be seamlessly integrated into python to apply it into a program. For this project, it was used to operate the user’s camera as Mediapipe, or any other library in use, doesn’t have this sort of functionality.

## PyAutoGUI

PyAutoGUI allows python scripts to make use of the mouse and keyboard which lets you automate interactions with different computer applications (Sweigart, 2021). In this project it will be used to enable the different functions triggered by the gestures. First, the position of the index finger and thumb tips will be tracked by storing the coordinates of the Mediapipe landmarks in the camera feed, then they will be translated into x and y coordinates on the screen to move the mouse cursor in accordance with the position of the hand landmarks.

## Gestures/Functions

This section describes the gestures and functions that will be included in the final program. It is assumed that the user will only have one hand present at a time in the camera feed. Functions outlined in sub-sections 4.3.1 to 4.3.3 are performed using your right hand and the function in sub-section 4.3.4 is performed using the left hand.

## Cursor Movement

In order to move the cursor, the user needs to make a pinch gesture with their index finger and thumb using their right hand. The program will detect if the user has moved the index and thumb fingertips together, in a way such that they do not touch but are close to each other (space of approx. 2 to 4 cm in-between). The cursor will then move on the screen in accordance with the position of the user's hand.

## Left Click

To do a left click function, the user needs to touch their index and thumb fingertips together in a similar position which is used for moving the mouse cursor. The application will detect that and perform a left click. The program will freeze (sleep) for a second so that a single click is detected instead of multiple in a row.

## Right Click

To perform a left mouse click, the user needs to touch their middle (ID of 12) and thumb (ID of 4) fingertips together. The application will detect this and do a left click function. As earlier, the program will sleep for a second so that a single click is detected instead of multiple in a row. The user is expected to hold their hand facing forward, palm towards the camera, as the mediapipe model can sometimes confuse the middle finger with the index finger if the hand is positioned in any other way. This would make the program think that the user is instead trying to move the cursor or do a left mouse click.

## Scrolling

To scroll (for example, a web page), the user needs to bring up their left hand and do a pinch gesture, the same one that is used to perform a left mouse click. The program will detect whether the hand is left or right using the handedness feature from the mediapipe pipeline. After a pinch gesture is detected, the user can move their hand (still in the same gesture) up and down to scroll accordingly.

## Processes Flowchart (Appendix I)

First, after the application is started, the program gains access to the user’s webcam using OpenCV. The hand and its landmarks are then recognised using the Mediapipe framework. Each landmark is plotted on the hand, on the live feed from the webcam.

Every important point (i.e. joints, knuckles, ends of fingers) is plotted using points and connected with lines. Small circles are also drawn around the tips of the index finger and the thumb. This is done to make it easier for the user to use the cursor movement feature of the program. Afterwards, the code detects if there is a gesture being performed or not. Depending on the gesture, the following functions are executed: moving the desktop cursor, a left mouse click, a right mouse click, and a scroll function. These are done by using the PyAutoGUI library which allows python scripts to make use of the mouse and keyboard to automate interactions with computer applications.

## Summary

This section outlined the important process in every project, design. It has described the methodology behind this project, i.e. how and using what tools the project will be developed. Code will be written using the Python programming language, gesture recognition will be done with the use of Google’s Mediapipe. OpenCV will be used to gain access to the user’s camera and PyAutoGUI will be used to transfer the real-life coordinates of the user’s gestures to screen coordinates.

It also described the different functions that will be implemented and what kind of gestures will be used. The described functions were, mouse tracking, left click and right click function, and a scrolling function.

The last thing that was outlined, was the flowchart of how the process of the application works. The flowchart provides an easier way to understand the whole process.

# Chapter 5: Development

## Introduction

The development of the application took the bulk time of the schedule. It was developed using python while making use of OpenCV for operating the camera, and Mediapipe for gesture recognition related operations. As to why python was chosen as the programming language, is because there is a lot of widely available documentation for this programming language, and the language is very commonly used in various applications for different scenarios. In fact, from what I’ve encountered in my research of already existing applications in the computer vision field, a huge chunk of them, if not most, are developed using python. It is one of the most commonly used programming languages. Additionally, one of the language’s attributes is its ease of use. Python code looks like plain English, therefore it lets the developer focus on the design of the application and not just on coding. It does, however, have worse performance compared to languages such as C++.

## Planning

Planning is an extremely important aspect of development. It is an imperative step in order to develop a successful solution. At the very beginning of the project, an early-stage Gantt chart was made that outlined the time frames of different stages of development (Appendix A.). I have tried my hardest to stick to it, however, as it often happens, not everything goes as planned. At first, I researched information on the Literature Review. This included reading academic papers, mostly on how gesture recognition can be performed and what methods can be used and after that, the literature review was written. More research after that was performed on already existing solutions for the problem area of this project. This gave me inspiration on what to include in my final application, what functions to implement, and how to implement them.

After that, I moved on to the design stage of the project. This included working out how the application could be structured, i.e. what gestures could be used etc. At first, this stage was mostly included because I was planning to develop a front-end GUI for this application. However, as it is later mentioned, this idea had to be scrapped due to time constraints.

Later on, the backend development of the application was started. This took a huge bulk of the time given for this project. It was delayed and done later than planned due to personal reasons such as health issues, but in the end, most of the features planned were implemented.

Throughout the whole project, notes were taken on different aspects of it, so that they would be included in the final report. The final report stage took most of the time given for this project. At first, a draft was planned, and a later final report was supposed to be rewritten taking into account the parts of the draft that went wrong. This idea was discarded as I decided it would be more time-efficient to just write the whole report, proofread it, and edit parts of it or/and add to it. Also, throughout it, I contacted my supervisor through email and made appointments whenever an issue arose.

Overall, a plan was made, most of it wass followed, however some parts had to be delayed or postponed due to issues out of my control.

## 5.3 Development Cont.

In this project OpenCV was only used in the code for accessing the user’s camera, however, it also has features that allow for object detection, gesture recognition, etc. meaning that it has most of the features that mediapipe has. The library is coded using C/C++ which means it can be imported to other languages (in this case python), and still make use of the features that C/C++ has, such as its improved performance. It has a large community of developers who contribute to the library providing help and resources making them easier to find. Also, it is free to use as opposed to programs like MATLAB which can also be used for computer vision tasks. As mentioned before, OpenCV can also be used for computer vision tasks such as gesture recognition just like mediapipe. This means I could’ve used just OpenCV for implementing the computer vision part of this project, however I have decided against that. This is because, Mediapipe is overall easier to use, having a simpler API and already pre-built functions, making it easier to program basic solutions. This made developing my program faster which was beneficial with the given timeframe. Although using just OpenCV could have made the application perform faster and it overall allows for more detailed control over what is being programmed.

A development process can be very complex, and a lot of major changes will probably be made. Some ideas that seem feasible before the coding stage, had to be completely discarded. At the very beginning of the development, I tried to recognise gestures by first capturing the image using the camera, selecting a ROI (region of interest) from the input, and transforming the camera frames into binary images from an HSV, in order to separate the hand from the background. This was extremely ineffective as the lighting conditions of the area that the user was in, had to be very specific which wasn’t realistic and any changes in skin colouration would make the program inaccurate or completely unusable. In addition, even when it did work the mouse tracking feature that was implemented was not working properly in this model. The cursor would be very jittery and inaccurate.

Starting all over again, I implemented the Mediapipe framework into my program. This made hand tracking much more accurate. It works in varying light conditions and should in theory work with different skin colours as it is a model trained by Google developers with thousands of diverse images.

First, the camera is initialised using OpenCV inside of a “while” loop and will be running until the user presses the escape key. The hand’s landmarks that are stored in the Mediapipe framework are then plotted on the live camera feed of the user’s hand. Those landmarks help identify important points on the hand. Later on, a function in the code checks if the hand used is left or right. This is done by using an inbuilt function in Mediapipe which returns the “handedness” of a hand, that determines whether the hand is left or right. If the code detects that the input hand is right, it constantly checks if the index finger and the thumb fingertips are closer than around 100 pixels to each other, but further than 50 pixels. If they are, the method that implements the code for tracking the mouse cursor movement is called upon. When the user moves their hand in that position, the x and y coordinates of the fingertips in the camera feed are translated to x and y coordinates of the screen. The resolution of the camera input and the screen are not predetermined, therefore this, in theory, should work with any camera and computer screen. The code then constantly looks for whether a gesture is performed. The gestures are the same as outlined before in section 4.3. If the user’s input hand is left, the code checks if the user’s index and thumb fingertips touch together. When the user moves their hand in that position up and down, the program scrolls accordingly.

## Code Description

Overall, in the code at this moment, there are 3 files, “main”, “camera\_operations” and “gestures”. The “main” is where the application is initiated, and all the other parts of the code are called (Appendix C). A new instance of the “camera\_operations” file is made, and that code is called.

The “camera\_operations”, as the name suggests, is used for accessing the camera and recognising the hand. The camera is first opened using OpenCV (Appendix D), the hand is recognised, and landmarks are plotted on the hand using the Mediapipe library (Appendix F).

The code for hand recognition and different function gestures is initiated or called in a while loop, which makes it so that code is only executed when the camera is opened (Appendix D). At first in this while loop, the landmarks of the hand are drawn on the camera feed (Appendix F). This helps debug the system and recognise which part of the hand responds to which landmark. The code after that checks if the hand in front of the camera is left or right using a switch case statement. This is because there are different functions that the program can execute depending on which hand is shown and the condition for their execution is a Boolean statement (Appendix G). The code later calculates the distance between the user’s index finger or middle fingertip, and thumb tip. If the distance between the index finger and thumb tip is between 50 and 100 pixels, the mouse cursor is moved to the position of the index finger. If the distance between the index finger and thumb tip is less than 20 pixels a left click is performed. If the distance between the middle finger and thumb tip is less than 20 pixels a right click is performed. Depending on these factors, different functions are called from the “gestures” file. If the user’s input hand is left, the code checks if the user’s index and thumb fingertips touch together. When the user moves their hand in that position up and down, the program scrolls accordingly. They are all done using if statements (Appendix H).

The gesture file includes the code for five different functions. The cursor movement, left click, right click, determining whether the hand is left or right, and a scroll function. They are all done in a different file and separate methods so that they are easier to access and more organised during development.

## Encountered Problems

Rarely do projects go as planned. They often go through many iterations before arriving at the final, complete stage. This is due to different factors, such as time constraints, changes in requirements, feasibility of the planned requirements, etc. There were multiple times in my project where features had to be scaled down or removed as a result of those factors.

One of the problems encountered was figuring out which method of gesture recognition that were mentioned in the Literature Review, to use for this project. I had to factor in which method would give the best results, while being feasible to develop in the given timeframe. This resulted in me, at first, choosing a method of gesture recognition that uses binarization to separate the hand from the background, calculating the center of the user’s finger, and using its location for cursor movement. This was the first iteration of this application.

As mentioned before, another problem occurred when the first iteration of my project had to be completely discarded. In the prototype, at the beginning, each frame of the camera feed was first into an HSV colour scheme of the RGB model. This made it then easier to transfer the image into a binary image by thresholding the colour that corresponded to the colour of the hand. The first iteration made use of binarization (converting images into binary images), to separate the foreground (the hand) from the background. After all of that, the location of the index finger was calculated, and the x and y coordinates were transferred to the desired cursor position on the screen. This prototype was rejected because the results were unsatisfactory, i.e. the hand tracking was extremely inaccurate, worked only under very specific conditions and only with the same skin colour as mine. Also, the prototype would detect some background items as the foreground which made the accuracy inadequate. To combat this issue, I used the previously mentioned Mediapipe framework. This made hand tracking much more reliable under various lighting conditions and with different skin colours. Additionally, to fix the issue of the model confusing background items as foreground, a ROI (region of interest) could be used so that only the hand would be in view.

Another issue was working out which gestures worked and were intuitive for the required functions of the application. At first, the mouse was moved by only raising your index finger and the mouse moved in accordance with the x and y position of the index fingertip. This worked well in terms of mouse tracking, however, it made it difficult to implement other functions, such as a left and right mouse click, because the mouse would move quite a lot if the user wanted to perform any other function. As an alternative, I made it so that if the index finger and thumb fingertips are close together, only then is the mouse tracked, and if the user touches those fingers together, a left mouse click is executed. Then if the user wants to perform a right mouse click, they will need to touch their middle finger and thumb fingertips together. However, in hindsight, the same thing could have been done for the first iteration of gestures. The distance from the base to the tip of the index finger could have been calculated, and a percentage threshold could have been set so that the cursor is not moved if the fingertip is below or above it.

At first a front-end was planned for this project. The GUI would display the user’s webcam and let them choose their dominant hand, however, I underestimated the time constraint on this project and the other features took more time than expected. As a result, this feature had to be discarded. To combat this for the future, I would have to manage my time more efficiently and not only research how to code certain features, but also think more about how feasible they are with the given time constraints. In this GUI, another feature was to be implemented at one point. This was the aforementioned feature of allowing the user to choose their dominant hand. The user would use their dominant hand as the virtual mouse and the other would be free to use for gestures that would be programmed to perform shortcuts. It is now assumed that the user will use their right hand as the mouse, and the other for shortcut gestures.

Another problem encountered was working out how to implement the scrolling feature of the virtual mouse. At first, the plan was to make it so that the user could move their left hand up and down in the pinch gesture to scroll up and down. This way of performing this function had to be discarded. This type of gesture would feel more natural, however, many issues were encountered, where either the function would not work at all and not do anything, or the overall application would crash whenever the pinch gesture was performed. As a compromise, it was decided upon to try and change it so that when a pinch gesture is performed with an index or middle finger, the function would scroll up or down respectively. However, the issue was resolved in the end, and the original idea worked properly. Now when the user does a pinch gesture with their index and thumb fingers, and moves their hand up or down in this position, the program will scroll up or down accordingly.

Lastly, one more problem was encountered during development that remains in the deployed solution. The problem is that when the user tries to make a gesture, the application slows down quite considerably. The cursor movement is slower than desired. One way to combat this would be for the user to move their hand carefully and slowly so that the cursor can be kept track of and isn’t all over the place. However, an actual solution could be to use more OOP (object oriented programing) for accessing different attributes, variables, and functions of the program, instead of implementing gestures nested for loops. As mentioned before in this report, using just OpenCV without including Mediapipe could be considered a solution as well. OpenCV is made using C/C++ which is more optimised than python.

## Summary

This section outlined the development process of the application. It described the plan for, not only the development part of the project, but the whole project overall. It also explained what was used in order to make the application; the kind of tools that were used, such as, OpenCV for camera operation, Mediapipe for gesture recognition etc. Explanations were given as to why I would choose to use these tools. Furthermore, the code of the application was explained with the references to small snippets of the code showing the important parts.

# Chapter 6: Testing

## Introduction

Applications need testing methodologies to fully examine their robustness so that they meet their specified requirements. It is performed to check if everything works as intended or not, and if there are any possible errors. Testing methodologies are different strategies and approaches to testing a specific product (*Intro Guide To Software Testing Methodologies | Inflectra |*, n.d.). There are many different ones, such as: the waterfall model, agile model, iterative model, verification and validation model, spiral model, extreme programming model, etc. (*What Is Test Methodology?*, n.d.)

In this section, the testing results of the final application will be outlined. Tests will be repeated multiple times in a row to see if they are consistent. Testing will be documented in a table in the following format with these headings:

1. Testing scenario - this column will state what is being tested.
2. Description - this section will briefly explain the testing scenario and what is being tested in more detail.
3. Input - this will show what is being done to test this scenario, meaning it will show the type of input and what is being input for the test.
4. Desired outputs - shows what the result of the test is supposed to be.
5. Actual outputs - this section will show what the actual result of the test is.
6. Reliability - tells if the results of the test are consistent. It will range from mostly reliable (actual result is the same as the desired output most of the time) to reliable (actual result is always the same as the desired result).

## Test plan

For this project, an own methodology was used, as the ones mentioned before involve a client whereas this project does not have a set client. Also, concerning the testing for this application, tests will be performed after reaching a good stopping point and while adding significant features, for e.g., adding a function to the code and ensuring it works. The tests and their results will be documented. I will adapt the application and the requirements in accordance with the results of those tests.

The testing layout will look like this: first, there will be a column describing the testing scenario (what was tested), next, there will be the input for the test (such as strings or a gesture), the desired output of the test, the actual output and evidence of the test (screenshot of the actual output).

## Results

Table 2: Testing results

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Testing Scenario** | **Description** | **Input** | **Desired Outputs** | **Actual Outputs** | **Reliability** |
| Test camera feed | To test if the camera turns on and works appropriately | N/A (execute the code) | Executing the code would show the camera feed in a window | Executing the code shows the live feed from the user’s camera in a separate window | Reliable |
| Test mouse cursor movement | To test if the cursor moves with accordance to the movement of the user’s hand | Holding the tip of the index finger and the thumb around 1 to 2 cm from each other | Performing the gesture will allow the user to move the desktop’s cursor by moving their hand in this gesture. If the user moves their fingers further away than ~2 cm then the cursor should stop | User can move the cursor using their hand in the appropriate gesture, and if the user moves their fingers further away, the cursor stops | Reliable |
| Test left mouse click function | This tests if the user can perform a left click function by touching their index finger | Performing a “pinch” gesture - touching the tips of the index finger and the thumb together | Performing this gesture would allow the user to perform a left click function of a mouse | The user can perform the left click function of a mouse by doing the appropriate gesture | Reliable |
| Test the right click function | This tests if the user can perform a right click function of a mouse by touching the tip of their middle finger and thumb | Performing a gesture in which the user touches the tip of their middle finger, and the thumb together | Making this gesture would allow the user to perform a right click function of a mouse | The user can perform the right click function of a mouse by doing the appropriate gesture | Mostly reliable to reliable\* |
| Test the scrolling function | This tests if the user can perform the scrolling function of a mouse by using their non-dominant hand to make a pinch function and move their hand up and down to scroll accordingly | Performing a pinch gesture with your non-dominant hand, and move the hand up and down to scroll accordingly | Making this gesture would allow the user to scroll up and down on, for e.g., a website |  |  |

\*The output is mostly reliable if the user’s hand is facing sideways. If the user’s hand is facing forward while performing the gesture, the output is reliable. This is because, if the hand is facing sideways, the model sometimes confuses the middle finger for the index finger and starts moving the cursor and doing the left click function instead.

## Summary

This section outlined the testing process of the final application explaining the testing plan and showcasing the results. The tests were performed multiple times, and each feature was tested as it was developed and at the very end as well. The results of the tests show that the application functions mostly as intended, although the “right-click” function is only mostly reliable. It works very well if the user’s hand is facing straight towards the camera.

# Chapter 7: Evaluation

## 7.1 What was achieved

This section will outline which parts of the deliverable of this project were achieved or not, based on the functional and non-functional requirements, and supported by testing performed in the previous chapter. Not all of the requirements were achieved as planned. This due to multiple factors such as time constraints, difficulty of implementation, and certain health related personal issues which further reduced the available time frame.

## 7.1.1 Functional

Table 3: Table of whether the functional requirements were achieved.

|  |  |  |
| --- | --- | --- |
| Requirement | Achieved or not | Description |
| Implement a function for cursor movement | Achieved | Mouse tracking has been implemented and works as intended. |
| Implement a function for a left mouse click | Achieved | The function for a left click has been implemented and works as intended. |
| Implement a right click function | Achieved | The function for a right click has been implemented and works as intended. |
| Code a feature to recognise which hand is left or right | Achieved | Application recognises which hand is left or right |
| Implement a scroll down function | Achieved | A scroll down function has been implemented and works as intended. |
| Implement a scroll up function | Achieved | A scroll down function has been implemented and works as intended. |
| Develop a front-end GUI | Not achieved | Unable to develop due to time constraints, as other features and aspects of the project required more focus. |
| Implement additional gestures that can be used as shortcuts | Not achieved | Unable to implement due to time constraints, as other features and aspects of the project required more focus. |

## 7.1.2 Non-Functional

Table 4: Table of whether the non-functional requirements were achieved.

|  |  |  |
| --- | --- | --- |
| Requirement | Achieved or not | Reason |
| The final application needs to respond to the user’s gestures in real-time or with menial delay. | Achieved - the response time to the user’s gestures is minimal. | N/A |
| The application should operate in high enough frame rate so that it can be used smoothly. | Not achieved | Code for gesture recognition is implemented using many for loops therefore making it not as smooth as it can be, although still usable. The user needs to be careful when moving their hand and performing different gestures |
| It should have high accuracy in recognising the user’s gestures. | Mostly achieved | It is accurate if the user moves their hand carefully and their hand is facing towards the camera, although still usable if the hand is at an angle. |
| The program should have minimal or no amount of errors so that it is acceptable for everyday use. | Achieved | There were no explicit errors encountered during testing that would disturb the performance of the application. |
| The system should work with existing digital environments such as a browser and the user’s desktop, and anything that doesn’t have its own mouse input. | Achieved | The system works in environments such as a browser, emails, desktop etc. Environments that use the device’s cursor and not its own independent one. |
| The application could work on systems other than Windows computers, such as, MacOS, mobile, etc. | Not achieved | There was no time left to check if the developed application works on other systems, and cater it towards those systems. |

# Chapter 8: Conclusion

## Recommendations

The application as it is at this stage works perfectly fine, however, there are many improvements that can be implemented in order to better the performance of the gesture recognition and introduce new features.

As it currently stands, the application, to an extent, is not very fluid. What is meant by that, is that the program’s gesture recognition model sometimes confuses one hand landmarks for another, for example, on occasion the index finger is confused with the middle finger and vice versa. This causes the model to recognise a left click gesture as a cursor movement gesture. This could be due to different conditions such as the environment you’re in, the quality of the camera, or lighting conditions. Probably, the only way to fix that would be to make a more accurate gesture tracking model than Google mediapipe, which is not feasible by one person within the given timeframe. Although, one way in which I could improve it, is by changing how the gestures are performed, to change the already coded gestures into more intuitive ones. This could include changing the already existing gestures so that when the user does a peace sign, the cursor is moved and when the user puts down their index finger or middle finger a left click or a right click, respectively, is performed. The fingers to do this would be reversed if the user was left-handed. This would certainly make the application more intuitive and the overall camera performance smoother as Google’s Mediapipe already has an inbuilt peace sign gesture which means that the program would not need to look through every hand landmark every time whenever the user wants to move the cursor.

Going back to the performance of the program, another issue with it is that the camera frame rate noticeably reduces when the user makes a gesture. When the user just puts their hand up in front of the camera without doing any gesture, the fps (frames per second) is normal, however as soon as the user performs a gesture, the fps drops. This could be due to some poor choices of code, such as using nested “for loops” for the part of the code that recognises what gesture the user does. What could have been done instead, would be to use more OOP (object-oriented programming) to make a class, fill it with different functions, attributes, variables and use it to recognise gestures. Already existing programs on the internet use this kind of programming and they are much more fluent and reliable.

In the future, a front-end GUI could be added to the application. This was originally planned in the beginning, however due to time constraints it was not implemented. The GUI could be used for things, such as, asking the user whether they are right or left-handed to determine which hand would act as the virtual mouse.

Another improvement that I would make in a future project is to better manage my time. This is not an improvement strictly related to the application itself, however I did learn this while doing this project in general. Improving time management would include stuff such as starting sections of the project in advance so that more features can be included in the final product and the application can be of higher quality.

## Aim & Objectives

The final product slightly changed from what was intended at the beginning, which is often the norm during the development process. Most of those changes happened due to not being feasible for me to develop in the given time frame. They include missing features that were planned to be added in the later development stages, and some non-functional requirements that were not fulfilled such as having a system for gesture recognition that is high enough framerate for it to work smoothly.

## Risks (did any happen, was your mitigation ok?)

Overall, there weren’t many risks that occurred during this project that were outlined before such as, device failure, the scale of the project being too large etc. However, there was one risk that did occur, it happened with the first iteration of the application. As outlined in sub-section 5.5, the first iteration was very inaccurate and only worked under very limited lighting conditions rendering it unrealistic for everyday use.

Although most risks did not appear, there were delays due to personal issues out of my control, which caused some features to be cut from the final application.

## Project Management

As outlined in the planning section of chapter 5 in this report, there was a plan in place with a made Gantt Chart. It was mostly followed, the Literature review, research, and design process were finished on time, however, some parts had to be delayed due to personal health issues that were out of my control, therefore parts such as the development and writing of this report were not finished or started exactly on time.

## Conclusion

In this project, the aim was to develop a virtual mouse application that would help the user to navigate a computer through the use of gestures performed using their hands. The requirements were to implement mouse tracking, left click, right click, scrolling function, and a possible front-end GUI. All of those but one were completed. The GUI was ultimately left out due to time constraints. Even though in the end, most of the requirements were fulfilled, it was unexpectedly difficult to achieve them. The first iteration or prototype of the application, which took a few weeks to develop, failed completely. This was because it was extremely inaccurate and only worked in very specific circumstances, therefore it had to be discarded. After the first iteration, a different approach was found. I learned that Mediapipe could be used for recognising objects and human features such as hands. It was then implemented in this project and used for gesture recognition. This has helped immensely with the development, as the Mediapipe model is made from thousands of different images that make its gesture recognition extremely accurate and, in the end, allowed for the development of this project.

However, this project has its limitations. Even though the Mediapipe model is extremely in-depth, its accuracy can still be negatively affected. It all depends on the user’s light conditions and even on the quality of the user’s webcam. In addition, the number of gestures that can be performed is somewhat limited, allowing for more future additions to the application. There are many aspects that could be further improved and researched, such as the fluidity of the final application as mentioned before in the report.

Overall, this project presented that gesture-based virtual mouse technology is very much feasible and promising. This project has taught me that gesture recognition, and computer vision in general, is a very widespread topic with a lot of already existing research. It is used in many various scenarios already and can be used as a solution for many different issues. I learned how to develop a basic solution based on gesture recognition and have acquired some experience on how to work with gesture recognition tools such as OpenCV and Mediapipe.

# References

Harvard APA v7

Ahuja, M. K., & Singh, A. (2015). Static vision based Hand Gesture recognition using principal component analysis. *2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE)*, 402–406. <https://doi.org/10.1109/MITE.2015.7375353>

AlSaedi, A. K. H., & AlAsadi, A. H. H. (2020). A new hand gestures recognition system. *Indonesian Journal of Electrical Engineering and Computer Science*, *18*(1), Article 1. <https://doi.org/10.11591/ijeecs.v18.i1.pp49-55>

Brahmbhatt, S. (2013). *Practical OpenCV*. Apress. <https://books.google.co.uk/books?hl=pl&lr=&id=_5sQAwAAQBAJ&oi=fnd&pg=PA2&dq=opencv&ots=8VkR9WM9c1&sig=iGyIOF0eyLHIDdN8If0UjT7a7Ow&redir_esc=y#v=onepage&q=opencv&f=true>

Chandra, N. S., Venu, T., & Srikanth, P. (2015). *A Real Time Static & Dynamic Hand Gesture Recognition System*. *4*(12). <https://www.ijeijournal.com/papers/Vol.4-Iss.12/P04129398.pdf>

Fang, Y., Wang, K., Cheng, J., & Lu, H. (2007a). A Real-Time Hand Gesture Recognition Method. *2007 IEEE International Conference on Multimedia and Expo*, 995–998. <https://doi.org/10.1109/ICME.2007.4284820>

Fang, Y., Wang, K., Cheng, J., & Lu, H. (2007b). A Real-Time Hand Gesture Recognition Method. *2007 IEEE International Conference on Multimedia and Expo*, 995–998. <https://doi.org/10.1109/ICME.2007.4284820>

Ghosh, D. K., & Ari, S. (2016). On an algorithm for Vision-based hand gesture recognition. *Signal, Image and Video Processing*, *10*(4), 655–662. <https://doi.org/10.1007/s11760-015-0790-4>

Han, J.-S., Lee, C.-I., Youn, Y.-H., & Kim, S.-J. (2022). A Study on Real-time Hand Gesture Recognition Technology by Machine Learning-based MediaPipe. *Journal of System and Management Sciences*. <https://doi.org/10.33168/jsms.2022.0225>

Haria, A., Subramanian, A., Asokkumar, N., Poddar, S., & Nayak, J. S. (2017). Hand Gesture Recognition for Human Computer Interaction. *Procedia Computer Science*, *115*, 367–374. <https://doi.org/10.1016/j.procs.2017.09.092>

Ibraheem, N., Khan, R. Z., & Hasan, M. (2013). Comparative Study of Skin Color based Segmentation Techniques. *International Journal of Applied Information Systems*, *5*, 24–39. <https://doi.org/10.5120/ijais13-450985>

*Intro Guide To Software Testing Methodologies | Inflectra |*. (n.d.). Retrieved 13 May 2024, from <https://www.inflectra.com/Ideas/Topic/Testing-Methodologies.aspx#:~:text=Testing%20methodologies%20are%20the%20strategies,it%20is%20fit%20for%20purpose.>

Ismail, A. P., Aziz, F. A. A., Kasim, N. M., & Daud, K. (2021). Hand gesture recognition on python and opencv. *IOP Conference Series: Materials Science and Engineering*, *1045*(1), 012043. <https://doi.org/10.1088/1757-899X/1045/1/012043>

Kavana, K., & Suma, N. (n.d.). Recognization of Hand Gestures Using Mediapipe Hands. *International Research Journal of Modernization in Engineering Technology and Science*. <https://www.irjmets.com/uploadedfiles/paper/issue_6_june_2022/27011/final/fin_irjmets1656344520.pdf>

Khan, R. Z., & Ibraheem, N. (2012). Hand Gesture Recognition: A Literature Review. *International Journal of Artificial Intelligence & Applications (IJAIA)*, *3*, 161–174. <https://doi.org/10.5121/ijaia.2012.3412>

Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M. G., Lee, J., Chang, W.-T., Hua, W., Georg, M., & Grundmann, M. (2019). *MediaPipe: A Framework for Building Perception Pipelines* (arXiv:1906.08172). arXiv. <http://arxiv.org/abs/1906.08172>

Mitra, S., & Acharya, T. (2007). Gesture Recognition: A Survey. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, *37*(3), 311–324. <https://doi.org/10.1109/TSMCC.2007.893280>

Oudah, M., Al-Naji, A., & Chahl, J. (2020). Hand Gesture Recognition Based on Computer Vision: A Review of Techniques. *Journal of Imaging*, *6*(8), 73. <https://doi.org/10.3390/jimaging6080073>

Patil, V., Sutar, Dr. S., Ghadge, S., & Palkar, S. (2023). Gesture Recognition for Media Interaction: A Streamlit Implementation with OpenCV and MediaPipe. *International Journal for Research in Applied Science and Engineering Technology*, *11*(9), 1039–1046. <https://doi.org/10.22214/ijraset.2023.55775>

Phutela, D. (2015). The Importance of Non-Verbal Communication: IUP Journal of Soft Skills. *IUP Journal of Soft Skills*, *9*(4), 43–49.

Rosenfeld, A. (1988). Computer vision: Basic principles. *Proceedings of the IEEE*, *76*(8), 863–868. <https://doi.org/10.1109/5.5961>

Sharma, A., Mittal, A., Singh, S., & Awatramani, V. (2020). Hand Gesture Recognition using Image Processing and Feature Extraction Techniques. *Procedia Computer Science*, *173*, 181–190. <https://doi.org/10.1016/j.procs.2020.06.022>

Sharma, N., Mishra, M., & Shrivastava, M. (2012). *COLOUR IMAGE SEGMENTATION TECHNIQUES AND ISSUES: AN APPROACH*. *1*(4). <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=8379d497eb4ada7c69fad7e23e159719e5e1fc6f>

Solem, J. (2012). *Programming Computer Vision with Python: Tools and Algorithms for Analyzing Images*. O’Reilly Media, Inc. <https://books.google.co.uk/books?hl=pl&lr=&id=gf2LxrmMzXAC&oi=fnd&pg=PP2&dq=python+for+computer+vision&ots=YPKeBLtTer&sig=XX-3KMJRT8Q4hcmSGrwnLKV1p-M#v=onepage&q=python%20for%20computer%20vision&f=true>

Suharjito, Ariesta, M. C., Wiryana, F., & Kusuma, G. P. (2018). A Survey of Hand Gesture Recognition Methods in Sign Language Recognition. *Pertanika Journal of Science & Technology*, *26*(4), 1659–1675.

Sung, G., Sokal, K., Uboweja, E., Bazarevsky, V., Baccash, J., Bazavan, E. G., Chang, C.-L., & Grundmann, M. (2021). *On-device Real-time Hand Gesture Recognition* (arXiv:2111.00038). arXiv. <https://doi.org/10.48550/arXiv.2111.00038>

Sweigart, A. (2021). *PyAutoGUI Documentation*. <https://readthedocs.org/projects/pyautogui/downloads/pdf/latest/>

Yewale, Shweta. K., & Bharne, Pankaj. K. (2011). Hand gesture recognition using different algorithms based on artificial neural network. *2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*, 287–292. <https://doi.org/10.1109/ETNCC.2011.6255906>

# Appendix A: Project Initiation Document (inc. Gantt Chart)

****

**School of Computing**

**Final Year {Engineering / Research / Study} Project**

**Project Initiation Document**

**Artur Walach**

**Computer Science**

**Hand Gesture Recognition**

**for Navigation**

# Basic details

|  |  |
| --- | --- |
| Student name: | Artur Walach |
| Draft project title: | Hand Gesture Recognition for Computer Navigation |
| Course and year: | Bsc 3rd Year Computer Science |
| Project supervisor: | Jiacheng Tan |
| Client organisation: |  |
| Client contact name: |  |

**In each of the following sections, please write your text below ours in regular (non-italic) font and then delete all blue text.**

# Degree suitability

My project satisfies the criteria because it involves concepts included in a module that I'm learning, computer vision, such as image processing, object recognition and deep learning.The project would further my knowledge on the topics covered in the aforementioned module. It would also test my programming skills and improve my skills to work on a project in a possible future programming job.

# The project environment and problem to be solved

My project will focus on developing a program that can recognise hand gestures and translate them into commands that can be used to interact with a computer. It would involve training a deep learning model to recognise different hand gestures in real time through the use of a webcam. This project would potentially help people who struggle with navigating a computer using a mouse and keyboard. It would possibly make devices easier to use for people with disabilities. Another potential client could be healthcare providers. Gesture recognition could be used for touchless interfaces in cases where hygiene is important, such as hospitals etc. This would make the hospital environment safer for people by reducing the spread of germs which are commonly found on devices in hospitals.

# Project aim and objectives

**Aim:** Program a hand gesture recognition software that allows a user to interact with a computer using hand gestures/movements.

**Objectives:**

* Research the topic thoroughly.
* Code a robust gesture recognition system
* Collect a dataset of different hand gesture images ensuring that a wide range of gestures and hand positions are covered or research already existing datasets
* Integrate a real-time video feed capture so that the gestures can be used.
* Improve accessibility for people who struggle to use traditional ways of operating a computer or a phone.

# Project constraints

* Hardware limitations - the user needs to have a computer/phone and a webcam for the program to work.
* Project timeline - the project may be limited due to the available time frame for the completion. Some potential features may be omitted due to the time constraints.
* Budget - if the program is not developed on a laptop, an external webcam will need to be bought.

# Facilities and resources

* Laptop or desktop
* Webcam, if not using a laptop
* IDE software for coding - free

# Log of risks

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Description** | **Likelihood**  **(high, medium, low)** | **Impact** | **Mitigation/Avoidance** |
| 1 | Laptop/desktop failure | Medium | Data loss and delays to the project | Backup on a USB or a cloud environment. |
| 2 | Scale of the project being too big | Medium | Would cause the project to be incomplete on deadline | Carefully consider the scale of the project and/or drop less important features as the project goes on |
| 3 | Accuracy of the gesture recognition | High | Would cause the final program to recognise gestures inaccurately and make it unintuitive to use | Collect a diverse dataset that would include pictures of intuitive gestures and fine-tune the system to improve its accuracy |
| 4 | User variation | High | The program could have issues with recognising gestures from people with different hand shapes, sizes and skin tones | Include data and perform testing on a diverse group of users |
| 5 | Environment variation | High | The program’s performance may be affected by differing lighting conditions and clutter in the background of the image reducing its accuracy | Include data and perform testing in varied lighting conditions and different environments |

# Project deliverables

* App prototype
* Final app
* Project report
* Requirements specification
* UML diagram
* Test reports/logs
* User guide
* Datasets

# Project approach

To manage my project I will use an agile development methodology, I will dedicate a couple/few months to code the project. Every time I implement a feature, I will test it to ensure it works and does what it’s intended to do. If it does not, then the issue will be documented and I will try to fix it. Unless the implemented feature is important, I will not spend too much time on it if it can’t be fixed easily and move on to more necessary features.

I will need to conduct secondary research in order to learn how to do some things, for e.g., coding some of the features that will be required in my software etc. This will possibly include looking through books, research papers using Google Scholar. I will also conduct primary research, such as looking at already existing software that is similar to what I intend to do, and taking inspiration for what kind of features my project should have, i.e., making a list of requirements.

# Project tasks and timescales

.

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Stage** | **Dates** | **Main Tasks** |
| 1 | Research topic | 25/09/23 - 18/12/23 |  |
| 2 | Project Initiation Document | 25/09/23 - 16/10/23 | * Write up PID * Gantt Chart |
| 3 | Literature & Ethics Review | 16/10/23 - 08/12/23 | * Research for Lit. Review * Write Literature Review |
| 4 | Further Research | 27/11/23 - 11/12/23 | * Research existing solutions |
| 5 | Design | 11/12/23 - 29/01/24 | * User Interaction Diagram (UID) * (possible) Frontend design * Complete presentation |
| 6 | Program Development | 11/12/23 - 29/04/24 | * Backend * Possible Frontend |
| 7 | Final Report | 06/11/23 - 13/05/24 | * Take notes along the way * Draft report * Finalise report * Proof-reading * Complete final presentation |

# Supervisor meetings

Meetings aren’t fully concrete yet, but I will contact my supervisor through email and meet face-to-face if possible every week or two, or when I’m stuck/need some advice on a certain part of my project.

# Legal, ethical, professional, social issues

**Legal:**

* If there is data stored about the user, the user needs to give consent and the data needs to be very secure so that a breach of privacy is avoided.
* If the project will collect data about the user, it cannot collect any other data aside from the one that the user consented to.
* The software cannot make use of any other part of the user’s computer other than the ones intended and the user consents to.

**Ethical:**

* Collecting data about the users, such as movement and behaviour.
* Possibility of the program collecting other data than just gestures, such as face, their environment etc. since the software will make use of the user’s webcam.
* Access rights - the software will need access to the end user’s webcam which would raise concerns about the user’s privacy.

# Permission

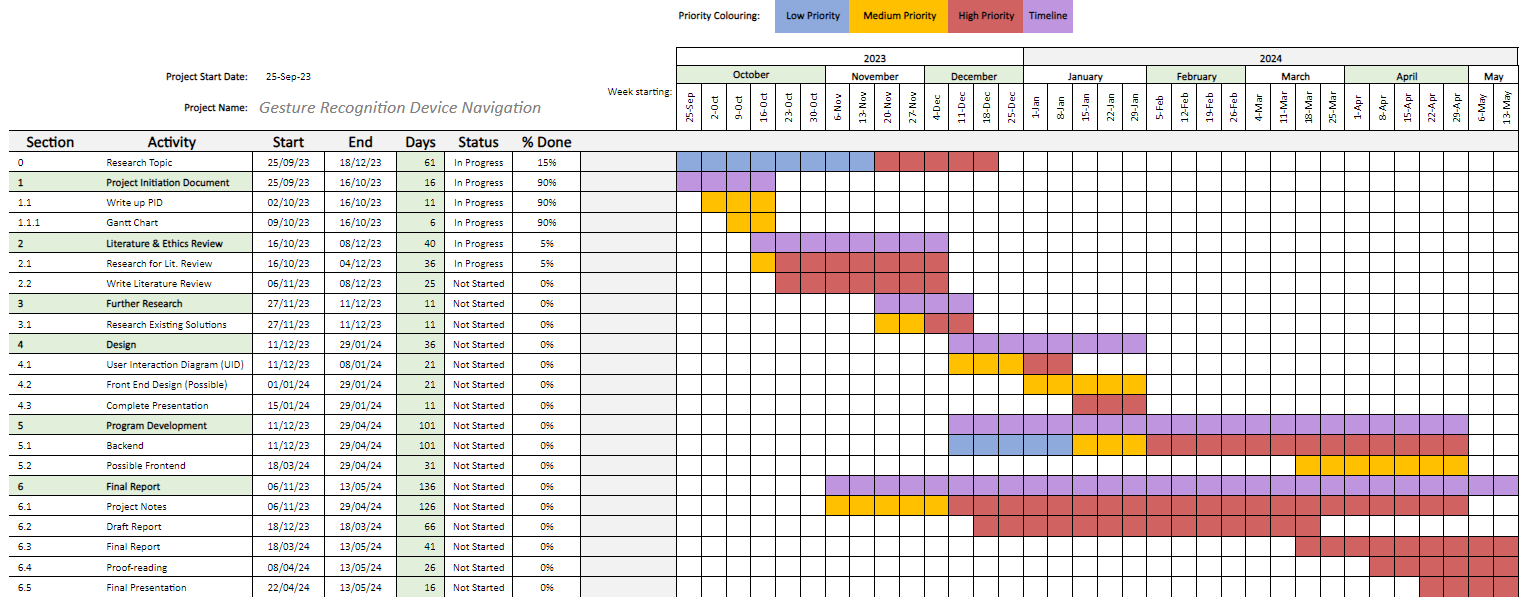
Please tick

|  |  |
| --- | --- |
| ❏ | I give permission for my PID to be made available to other students as examples of previous work. |
| ❏ | I do not give permission for my PID to be made available to other students as examples of previous work. |
|  |  |

Date: 15/10/23

**Appendix A: Gantt chart**

*Place your detailed project plan here.*



# Appendix B : Ethics Certificate



Certificate of Ethics Review

**Project title:** Hand Gesture Recognition for Computer Navigation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name**: | Artur Walach | **User ID**: | 2019085 | **Application date**: | 07/12/2023  18:37:21 | **ER Number**: | TETHIC-2023-107115 |

You must download your referral certificate, print a copy and keep it as a record of this review.

The FEC representative(s) for the **School of Computing** is/are [**Elisavet**](https://www.port.ac.uk/about-us/structure-and-governance/our-people/our-staff/elisavet-andrikopoulou)[**Andrikopoulou**](https://www.port.ac.uk/about-us/structure-and-governance/our-people/our-staff/elisavet-andrikopoulou)**,** [**Kirsten**](https://www.port.ac.uk/about-us/structure-and-governance/our-people/our-staff/)[**Smith**](https://www.port.ac.uk/about-us/structure-and-governance/our-people/our-staff/)

It is your responsibility to follow the University Code of Practice on Ethical Standards and any Department/School or professional guidelines in the conduct of your study including relevant guidelines regarding health and safety of researchers including the following:

* [University](https://staff.port.ac.uk/departments/services/corporategovernance/healthandsafety/healthandsafetypolicy/) [Policy](https://staff.port.ac.uk/departments/services/corporategovernance/healthandsafety/healthandsafetypolicy/)
* [Safety](http://www.southampton.ac.uk/~imw/safety.htm) [on](http://www.southampton.ac.uk/~imw/safety.htm) [Geological](http://www.southampton.ac.uk/~imw/safety.htm) [Fieldwork](http://www.southampton.ac.uk/~imw/safety.htm)

It is also your responsibility to follow University guidance on Data Protection Policy:

* [General](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr) [guidance](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr) [for](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr) [all](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr) [data](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr) [protection](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr) [issues](https://www.port.ac.uk/about-us/structure-and-governance/legal/data-protection-and-gdpr)
* [University](http://policies.docstore.port.ac.uk/policy-022.pdf) [Data](http://policies.docstore.port.ac.uk/policy-022.pdf) [Protection](http://policies.docstore.port.ac.uk/policy-022.pdf) [Policy](http://policies.docstore.port.ac.uk/policy-022.pdf)

Which school/department do you belong to?: **School of Computing**

What is your primary role at the University?: **Undergraduate Student**

What is the name of the member of staff who is responsible for supervising your project?: **Jiacheng Tan**

Is the study likely to involve human subjects (observation) or participants?: Yes

Will you gather data about people (e.g. socio-economic, clinical, psychological, biological)?: Yes

Will your data collection be strictly limited to gathering anonymous insights about a particular artefact or research question (e.g. opinions, feedback)?: Yes

Confirm whether and explain how you will use participant information sheets and apply informed consent.: I will use a consent form for whether the user wants to give feedback on the final product and I will also use another consent form for the participants that I will chose to collect hand gestures from. I will ask the participants whether they agree for their feedback to be collect and photos of their hands preforming different gestures.

Confirm whether and explain how you will maintain participant anonymity and confidentiality of data collected: The feedback provided will be anonymous, meaning that none of the participants' details will be mentioned. The collected photos of other participants' hands, that will be used for making up the final dataset, will not show their faces and only their hands preforming different gestures. Will the study involve National Health Service patients or staff?: No

Do human participants/subjects take part in studies without their knowledge/consent at the time, or will deception of any sort be involved? (e.g. covert observation of people, especially if in a non-public place): No

Will you collect or analyse personally identifiable information about anyone or monitor their communications or on-line activities without their explicit consent?: No

Does the study involve participants who are unable to give informed consent or are in a dependent position (e.g.

children, people with learning disabilities, unconscious patients, Portsmouth University students)?: No Are drugs, placebos or other substances (e.g. food substances, vitamins) to be administered to the study participants?: No

Will blood or tissue samples be obtained from participants?: No

Is pain or more than mild discomfort likely to result from the study?: No

Could the study induce psychological stress or anxiety in participants or third parties?: No

Will the study involve prolonged or repetitive testing?: No

Will financial inducements (other than reasonable expenses and compensation for time) be offered to participants?: No

Are there risks of significant damage to physical and/or ecological environmental features?: No Are there risks of significant damage to features of historical or cultural heritage (e.g. impacts of study techniques, taking of samples)?: No

Does the project involve animals in any way?: No

Could the research outputs potentially be harmful to third parties?: No

Could your research/artefact be adapted and be misused?: No

Will your project or project deliverables be relevant to defence, the military, police or other security organisations and/or in addition, could it be used by others to threaten UK security?: No

Please read and confirm that you agree with the following statements: I confirm that I have considered the implications for data collection and use, taking into consideration legal requirements (UK GDPR, Data Protection Act 2018 etc.), I confirm that I have considered the impact of this work and and taken any reasonable action to mitigate potential misuse of the project outputs, I confirm that I will act ethically and honestly throughout this project

# Supervisor Review

As supervisor, I will ensure that this work will be conducted in an ethical manner in line with the University Ethics Policy.

Supervisor comments:

Supervisor’s Digital Signature**: jiacheng.tan@port.ac.uk** Date**: 07/05/2024**

|  |  |  |
| --- | --- | --- |
| Faculty Ethics Committee Review  Ethics Rep comments:  Faculty Ethics Committee Member’s Digital Signature(s)**:** | **elisavet.andrikopoulou@port.ac.uk** | Date**: 13/05/2024** |

# Appendix C : “Main” Method Code

# A screen shot of a computer Description automatically generated

# Appendix D : Camera Initiation

# A screen shot of a computer program Description automatically generated

# Appendix E : Hand Detection and Landmark Plotting Code

# 

# A screen shot of a computer code Description automatically generated

# Appendix F : Handedness Check

# A screen shot of a computer code Description automatically generated

# Appendix G : Switch Case Statement for Gesture Functions

# A screenshot of a computer code Description automatically generated

# Appendix H : Gestures Code

# A computer screen shot of a program code Description automatically generated

# Appendix I : Processes Flow Chart Diagram

# A flowchart of a computer Description automatically generated