Virtual Mouse using skeleton recognition and ML

Yangan Yagol Xu Chen, 001186509

Software Engineering

University of Greenwich

Contact Information:
University of Greenwich
Old Royal Naval College, Park Row, London SE10 9LS

Email: yagolxuchen@gmail.com

Abstract

This paper shows the investigation, design and development of a Virtual Mouse to improve the interaction between humans and computers. Handy, the application, enables users to control their computer using hand gestures, offering a more intuitive and natural way to interact with the device. This project is developed as current models do not use natural gestures, making hard the human- computer interaction experience. The proposed model in this study begins with hand landmark extraction to determine the hand coordinates of the user's hand. These coordinates are then pushed to our ML model to predict the type of gesture performed. This allows the user to control their computer using hand movements in real-time. The results of our experiments demonstrate high accuracy in gesture recognition, and the Virtual Mouse performs well in real time. By using natural hand gestures Handy has the potential to improve the way users interact with their computers.

Introduction

Users have been always forced to use hardware devices that are not natural for human gestures. With gesture recognition the interaction between human and computers can be improved. More specifically, with hand gesture recognition users can control a device using hand gestures, providing a more natural way to manipulate and interact with computers.

This project analyses different hand recognition methods to implement a Virtual Mouse. By using lightweight recognition models, the interaction between humans and computers is improved. Hand gesture recognition has already been applied in various applications, such as virtual reality (LI et al., 2019), robot manipulation (Li, 2020), and sign language interpretation (Al- Hammadi et al., 2020). However, current implementations of virtual mouses are not popular as the gestures implemented are not natural for humans.

To capture gestures, many hardware tools can be used. In recent years, advancements have been made, including gloves with sensors (Dong, Liu and Yan, 2021), infrared sensors (Nogales and Benalcázar, 2021) and thermal sensors (Vandersteegen et al., 2020). By using these tools high accuracy can be achieved. However, these approaches are not suitable for end users due to the cost of the devices. Therefore, this project focuses on analysing webcam-based approaches. For webcam recognition, research papers and implementations for hand recognition using cameras are analysed. With many applications improving daily, there are still many challenges that need to be addressed, such as lighting changes, background, obstructions, skin colour, motion, skeleton and lag.

Our proposed solution follows a systematic process where the input hand is first found, and the landmarks are extracted. Once extracted, the hand landmark coordinates are processed using our ML algorithm to predict which gesture the user is making. Then, the gesture is passed to the application to execute gesture.

Project Objectives

1. The Virtual Mouse should accurately interpret and respond to hand gestures. This ensures reliable cursor movement and actions.
2. The system should be intuitive and easy to use, allowing users to adapt to the gestures.
3. The Virtual Mouse should have common mouse actions. This includes: pointer movement, left click, right click, scroll up, and scroll down.
4. The Virtual Mouse should have minimal lag to provide smooth movements.

Design of the application

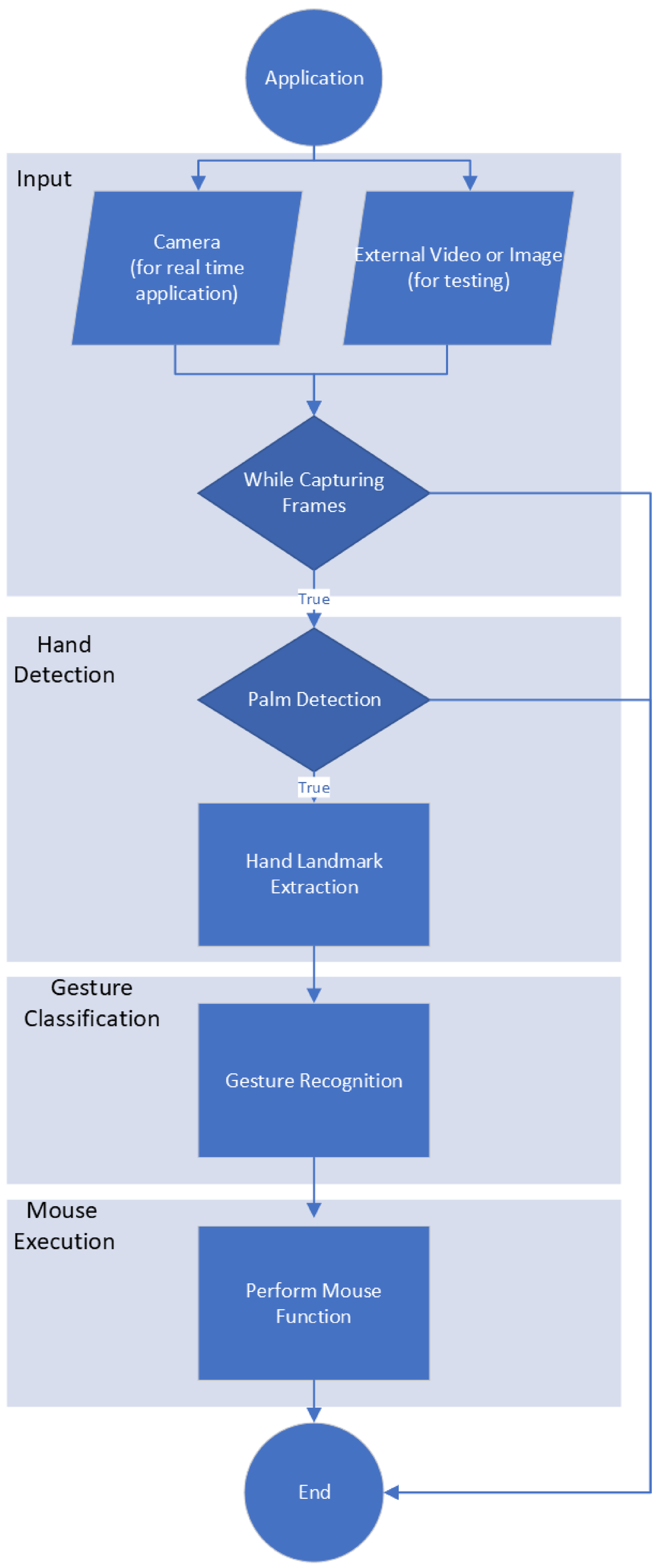

Figure 1: Design and implementation of Handy

Input

The first step consists of initializing the program with a webcam connected to the computer. Then, frames will be taken using OpenCV and passed to the hand detection algorithm. The implementation consists of an infinite loop that will be capturing every frame till the program is terminated.

Hand detection

In the second part of the algorithm, hand detection is performed with the frame obtained. The objective of this step is to identify the hand and extract the landmarks. If a hand is found, the algorithm extracts the hand landmark coordinates and pre-processes the data by normalizing the data and storing it in a vector.

Gesture classification

The Gesture Classification predicts the gesture performed. Six models were analysed, three Neural Networks, a decision tree, a random forest, and a SVM. Due to the principle of "no free lunch", there is no single algorithm that can optimally predict hand gestures. Therefore, different algorithms are analysed and at the end, the model with most accuracy is selected. With the best performing model gestures in real time are predicted. On the real time application, once the landmark is extracted, its vector is passed to the prediction algorithm to calculate the probability of the possible gestures.

Mouse Execution

After the gesture classification, the execution of the command is performed. It consists of six gestures: move pointer, left click, right click, scroll up, scroll down and nothing. For the Move Pointer command, a Dynamic Pointer (DP) is implemented. This feature allows the pointer to be controlled with a wrist movement rather than using the entire arm to move the pointer.

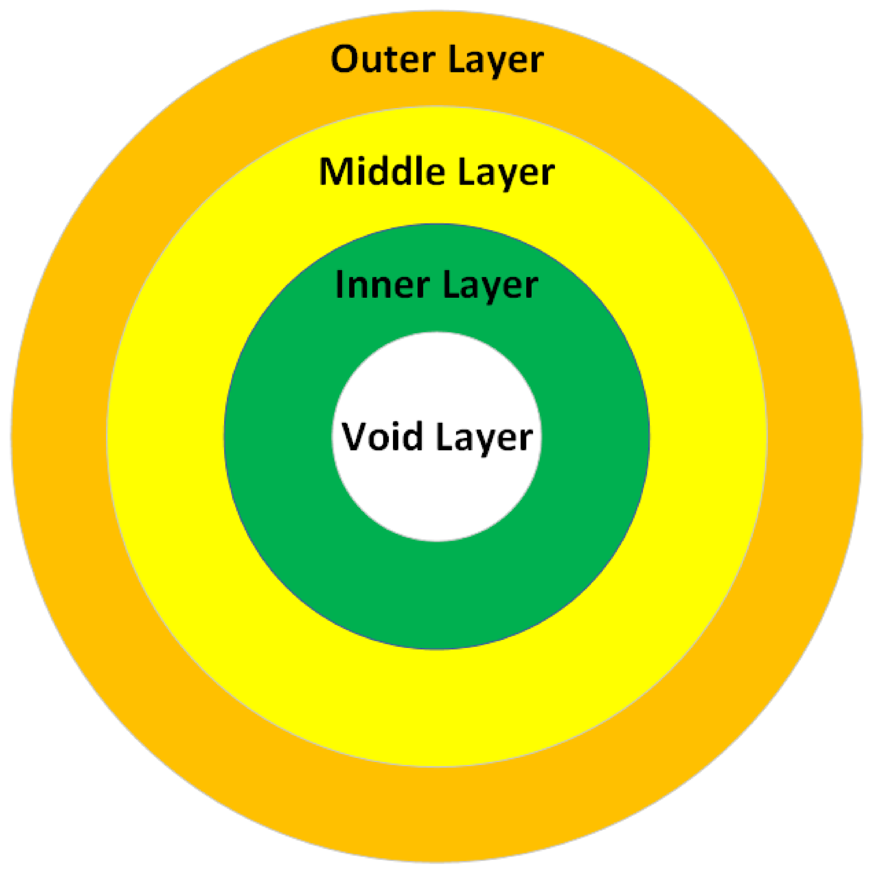

Figure 2: Dynamic Pointer for Pointer movement

Results

Six videos of 15 seconds were created for the real time testing. The NNs were chosen based on their performance on real time as it showed the best results due their capability of generalization. The results for NN1 are less compared to the ones obtained during testing due to the gestures complexity leading to a decrease in accuracy for most of the gestures. A similar trend is observed for NN2 and NN3. NN2 shows a slightly better accuracy than NN1, with an improvement of 8%. NN3 performs slightly better for the Pointer gesture but has a 5-point decrease in accuracy for the Left Click gesture. The Scroll Down and Scroll Up gestures achieve a 100% accuracy as the gestures are not correlated, resulting in perfect recognition. The performance differences between the testing and real-time results are due to the increased complexity of gestures in the real-time application. The models struggle to generalize leading to a decrease in accuracy. Despite this, the NNs demonstrate good performance.

Gesture	NN 1	NN 2	NN 3
Pointer	0.76	0.79	0.89
Left Click	0.84	0.87	0.82
Right Click	0.87	0.89	0.87
Scroll Down	1.00	1.00	1.00
Scroll Up	1.00	1.00	1.00
Nothing	0.91	0.91	0.93

Table 1: Accuracy of the NNs on real time

Conclusions

- Analysed different gesture recognition algorithms and combined different methods to create a new methodology using skeleton recognition coordinates.
- Implemented a more flexible model to control the mouse compared to other research studies and implemented the DP for mouse pointer movement.
- The hand gesture recognition model used in this project showed increased accuracy compared to other research.
- With the proposed methodology, Handy has shown good accuracy and smoothness in the real-time implementation.

Limitations and Further Work

To improve the performance and usability of Handy several areas can be explored. Firstly, conducting questionnaires and surveys on user preferences of the gestures would provide more information on intuitive gestures. Additionally, implementing additional functionalities in the mouse execution, such as shortcuts could improve the usability. Finally, analyzing the correlations between each finger and hand, and applying feature engineering techniques in the hand gesture recognition process could lead to more accurate and responsive mouse control.