ERGONOMIC VIRTUAL KEYBOARD

A Project Report

submitted to the APJ Abdul Kalam Technological University

in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology

in

Computer Science and Engineering

by

ABHISHEK U K (STM21CS006)

ALBIN BINU (STM21CS015)

AYISHA ZOOMI (STM21CS022)

NASLA SAFIYA K (STM21CS045)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ST. THOMAS COLLEGE OF ENGINEERING AND TECHNOLOGY

MATTANNUR

November 2024

DEPT. OF COMPUTER SCIENCE AND ENGINEERING

ST. THOMAS COLLEGE OF ENGINEERING AND TECHNOLOGY

MATTANNUR

2024 - 2025

CERTIFICATE

This is to certify that the report entitled **ERGONOMIC VIRTUAL KEYBOARD** submitted by **ABHISHEK UK** (STM21CS006) to the APJ Abdul Kalam Technological University in partial fulfillment of the B.Tech. degree in Computer Science and Engineering is a bonafide record of the project work carried out by him under our guidance and supervision. This report in any form has not been submitted to any other university or institute for any purpose.

Dr. SHINU MATHEW JOHN
(Project Guide)
Professor
Dept. of CSE
St. Thomas College of Engineering and Technology
Mattannur

Mr. JITHIN S
(Project Coordinator)
Assistant Professor
Dept. of CSE
St. Thomas College of Engineering and Technology
Mattannur

Dr. AMITHA I C
Head of Department
Dept. of CSE
St. Thomas College of Engineering and Technology
Mattannur

DECLARATION

I hereby declare that the project report **ERGONOMIC VIRTUAL KEYBOARD**, submitted for partial fulfillment of the requirements for the award of the degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of **Dr. SHINU MATHEW JOHN**

This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the sources.

I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data, idea, fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the university and can also evoke penal action from the sources that have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma, or similar title of any other university.

Mattannur

01-11-2024

ABHISHEK UK

ACKNOWLEDGEMENT

I take this opportunity to express my deepest sense of gratitude and sincere thanks to everyone who helped me complete this work successfully. I am extremely thankful to Principal **Dr. SHINU MATHEW JOHN** for giving me his consent for this project.

I express my sincere thanks to **Dr. AMITHA I C**, Head of the Department, Department of Computer Science and Engineering for providing me with all the necessary facilities and support.

I would like to express my sincere gratitude to the Project Coordinator, **Mr. JITHIN S**, Asst. Professor, Department of Computer Science and Engineering for the support and cooperation.

I would like to place on record my sincere gratitude to my project guide **Dr. SHINU MATHEW JOHN**, Professor, Department of Computer Science and Engineering for the guidance and mentorship throughout this work.

I sincerely appreciate my college for providing the **R&D** lab facilities that made my project possible.

Finally I thank my family and friends who contributed to the successful fulfillment of this project work.

ABHISHEK UK

Abstract

This project proposes a virtual keyboard system that offers a contactless, gesture-based typing solution by tracking and interpreting hand movements on a flat surface. The system is designed to enhance hygiene, accessibility, and convenience, especially in shared or mixed-reality environments where traditional keyboards may be impractical. Utilizing a standard camera, the system captures the user's finger movements and employs Mediapipe's hand-tracking module to detect precise finger landmarks. These landmarks provide x, y, and z coordinates for each fingertip, enabling the calculation of finger angles and distances, essential features for accurate gesture recognition.

The core of the system's functionality lies in its machine learning model, a neural network trained using the Multi-Layer Perceptron (MLP) algorithm. With an input feature set of over 75 values, derived from the angles and distances of fingertip landmarks, the model is capable of classifying gestures into 27 distinct key classes, covering each letter of the English alphabet. The dataset used to train this model was created specifically for this project, capturing a diverse range of hand gestures through the same image-processing pipeline. This data, organized into a CSV file with over 80 columns, ensures robust model performance and gesture classification accuracy.

Once the neural network predicts a keystroke based on the captured gestures, the system uses pyAutoGUI.press() to register the keystroke, providing real-time feedback by displaying the typed content on-screen. The proposed virtual keyboard system not only promotes a hygienic alternative to physical keyboards but also supports adaptable, portable input across multiple environments. Its potential applications extend to mixed-reality and augmented-reality systems, providing an efficient, touch-free interface for digital interactions.

List of Figures

Contents

Chapter 1

Introduction

The evolution of technology, especially in the fields of computer vision and machine learning, has paved the way for more sophisticated methods of interacting with computers and other digital devices. The Ergonomic Virtual Keyboard project capitalizes on these advancements to create an innovative typing system that redefines traditional input methods. Unlike physical or on-screen keyboards, which require physical interaction or occupy valuable screen space, this virtual keyboard utilizes a camera or sensor to capture real time hand and finger movements from both hands, translating them into text input seamlessly. This approach allows users to type by simply mimicking traditional typing gestures in mid air, without the need for any physical or touch based interaction. The system relies on advanced gesture recognition and machine learning algorithms to track the trajectories and patterns of the fingertips, predicting intended keystrokes with high precision. The core idea behind this project is to provide users with a more efficient, flexible, and ergonomic typing experience, especially in environments where traditional keyboards are either impractical or inefficient. This virtual keyboard is designed to enhance the user experience by offering a seamless and natural way to interact with devices, overcoming the limitations associated with physical and on screen keyboards. By leveraging the power of computer vision, the project aims to enable hands free, contactless typing that is suitable for various contexts, including mixed reality applications, mobile devices, and public spaces where physical keyboards may be inconvenient or unusable. The Ergonomic Virtual Keyboard represents a forward thinking solution in the realm of human-computer interaction, positioning itself as a modern alternative to traditional input devices while addressing both usability and ergonomic concerns.

1.1 Motivation

The motivation behind the Ergonomic Virtual Keyboard stems from the growing need for more flexible, efficient, and ergonomic input methods in an increasingly digital world. As technology continues to evolve, the reliance on traditional physical keyboards and touch-based input systems presents several challenges, particularly in mobile and mixed reality environments. One of the primary motivations for developing this project is the inefficiency of current input methods in modern computing environments. Physical keyboards, although effective, are cumbersome, occupying a large amount of space and often limiting user mobility. This is especially true in mobile and compact device settings where space is at a premium, and carrying an external keyboard is impractical. On-screen keyboards, while providing a solution, suffer from issues like reduced typing speed, lack of tactile feedback, and the consumption of valuable screen space, which limits productivity. These challenges highlight the need for an alternative that allows for efficient typing without the drawbacks of existing methods. Another significant motivation for this project is the desire to improve ergonomics. Long-term use of physical keyboards is associated with repetitive strain injuries (RSIs) and discomfort, particularly in professional environments where prolonged typing is required. By introducing a system that allows users to type through natural hand and finger movements, the Ergonomic Virtual Keyboard aims to reduce the strain on the hands and wrists, offering a more comfortable and natural typing experience. Additionally, the increasing popularity of mixed reality (MR) and virtual reality (VR) applications has created a demand for more intuitive and immersive interaction methods. In these environments, traditional input devices like keyboards and mice can feel out of place or even be unusable, prompting the need for gesture-based systems that align more naturally with the immersive experience. The virtual keyboard also addresses the growing trend of contactless interactions, particularly in the post-pandemic world, where minimizing physical contact with shared devices is a priority for hygiene and safety reasons. In public spaces, a contactless virtual keyboard can offer a cleaner, more hygienic alternative to shared physical keyboards. In summary, the motivation for this project lies in addressing the inefficiencies, ergonomic challenges, and contextual limitations of existing input methods while leveraging cutting-edge technology to create a more adaptable, comfortable, and future-proof solution.

1.2 Problem Definition

The **Ergonomic Virtual Keyboard** project aims to address several key problems associated with current input methods, particularly traditional physical keyboards and on-screen virtual keyboards. Traditional keyboards, while reliable, are not well-suited for all environments and use cases. They occupy considerable physical space, which makes them impractical for mobile devices or compact environments where portability and space efficiency are critical. Carrying a physical keyboard for mobile devices can be cumbersome and often defeats the purpose of having a portable device. Furthermore, physical keyboards are prone to wear and tear, and they lack adaptability in more modern contexts, such as mixed reality (MR) or virtual reality (VR) environments, where the need for a physical input device disrupts the immersive experience. Another significant problem with traditional keyboards is the ergonomic strain they impose on users. Prolonged use of physical keyboards can lead to repetitive strain injuries (RSIs), carpal tunnel syndrome, and other health issues due to the unnatural typing posture required. These problems are particularly pronounced in professional settings, where users may spend long hours typing, leading to discomfort and a decrease in productivity over time. The lack of flexibility in input posture and device setup is a major limitation in terms of user comfort and health.

On-screen keyboards, on the other hand, offer a partial solution by eliminating the need for a physical device, but they introduce a new set of problems. One of the main drawbacks is the waste of screen space. On-screen keyboards take up a large portion of the screen, reducing the available workspace for other tasks, which is particularly problematic on smaller devices like tablets and smartphones. Additionally, on-screen keyboards tend to be slower and less accurate than physical keyboards. Users often find it difficult to achieve the same typing speed and precision due to the lack of tactile feedback and the cramped nature of touch-based interfaces. This results in a slower, less efficient typing experience, particularly for tasks that require extensive text input, such as document creation, email communication, or coding. [1]

Another problem is the inconvenience of typing in public places. Physical keyboards, especially external ones, can be impractical in public or shared spaces due to privacy concerns, noise, and the general awkwardness of setting up and using a full-sized keyboard in a non-private environment. Similarly, using an on-screen keyboard in public

spaces can be uncomfortable, as it requires holding or positioning the device in a specific way, which can draw unwanted attention or limit the user's mobility. In certain situations, such as on public transport or in waiting areas, neither physical nor on-screen keyboards provide an ideal solution.

In light of these issues, the Ergonomic Virtual Keyboard seeks to solve the problems of space inefficiency, ergonomic strain, reduced typing speed, and inconvenience in public spaces by offering a contactless, gesture-based typing system. This virtual keyboard allows users to type using natural hand movements captured by a camera, eliminating the need for physical input devices and maximizing screen real estate. By leveraging both hands for typing, it offers a faster, more intuitive, and ergonomic typing experience. Furthermore, it provides a clean, contactless solution for typing in public spaces, offering users greater flexibility and comfort in a variety of environments

Chapter 2

Literature Review

2.1 Virtual Hands: Real Time Keyboard, Desktop & Application Navigation

The demand for alternative input systems, especially in light of the global pandemic, has increased the relevance of virtual keyboard interfaces that reduce physical contact with devices. The paper ”Virtual Hands: Real Time Keyboard, Desktop & Application Navigation using Gestures” [2] provides an innovative approach to solving this issue by using hand gestures recognized through a webcam, eliminating the need for traditional physical keyboards. This literature review aims to explore the foundations of gesture-based virtual input systems, highlighting significant contributions and their limitations that led to the development of the proposed system.

2.1.1 Historical Background and Early Contributions

The concept of gesture-based virtual keyboards is not entirely new. Researchers have been experimenting with non-physical input methods for years, particularly in areas of computer vision and pattern recognition. One of the earliest works in this field is by Segen et al. (1999) [3], who proposed a system that analyzed the user's hand shadow to detect gestures. However, the major limitation of this approach was its dependency on optimal lighting and distance conditions. If the user's hand came too close to the camera, the shadow became indistinct, leading to poor recognition performance.

Another significant contribution was made by Mantyjarvi et al. (2002) [4], who

introduced an improved gesture recognition system for a virtual keyboard. This system required the user's finger to be held perpendicular to the camera to register keystrokes, making it inconvenient and unnatural for continuous typing. The difficulty in achieving consistent finger positioning reduced the usability of this system, highlighting the need for more adaptable solutions.

Jun Hu et al. (2014) took a different approach by developing a system that used a camera and projector to detect bare fingers on flat surfaces such as walls or tables [5]. While this method was more effective in terms of precision, it required expensive additional hardware, increasing the complexity and cost of the setup. This reduced the practicality of the system for widespread use.

Another method, introduced by Jadhav et al. (2016) [6], aimed to simplify the hardware requirements by using a printed keyboard on paper, combined with camera-based hand recognition. However, this approach was limited by the fragility of the paper and the need for a stable surface, which made it unsuitable for various real-world applications.

These early attempts demonstrated the potential of gesture-based input systems but also highlighted the limitations, such as the reliance on external hardware, complex setups, and restricted usability in everyday environments. These challenges laid the groundwork for further innovations, focusing on contactless, software-driven solutions using commodity hardware like webcams.

2.1.2 Advancements in Virtual Gesture-Based Input Systems

Recent advancements in computer vision and machine learning have significantly improved the feasibility and accuracy of gesture-based virtual input systems. Technologies like OpenCV and MediaPipe have enabled real-time hand and finger detection, which are crucial for creating responsive and accurate virtual keyboards. These advancements allow for more natural hand movements to be detected and processed in a virtual environment.

The system proposed by Harshit Sharma et al. (2022) builds on these innovations, aiming to address the practical challenges of virtual keyboards by utilizing widely available hardware, such as standard webcams, and integrating sophisticated machine learning models for hand gesture recognition. The authors' use of MediaPipe, a library

developed by Google for real-time hand tracking, enables their system to detect 21 distinct hand landmarks in real time. This detection allows for highly accurate tracking of finger movements, which is essential for creating a usable virtual keyboard. [2]

The inclusion of PyVDAfor desktop navigation and PyttSX3 for text-to-speech feedback enhances the usability of the system, making it more intuitive for users to switch between windows or desktops without physically interacting with a keyboard or mouse. This integration reflects a shift in the design of virtual input systems towards user-friendly, multi-functional interfaces.

One key innovation in Sharma et al.'s system is the decision to use the BlazePalm Detector in combination with a hand landmark model from MediaPipe. The BlazePalm Detector efficiently identifies the user's hand, and the hand landmark model processes the position and depth of the fingers to detect when the user "presses" a key by bringing their index and middle fingers together. This method significantly reduces the hardware costs associated with previous systems by eliminating the need for specialized sensors or surfaces, making it a viable solution for everyday use.

2.1.3 Technological Components and Enhancements

The system leverages several key technologies that contribute to its robustness and versatility:

- **OpenCV**: This open-source library is crucial for real-time image and video processing. It captures the hand movements using the webcam and processes the video stream to detect finger positions, enabling the system to respond immediately to user inputs.

- **MediaPipe**: A central component of the system, MediaPipe is used to track hand movements in real time. It detects hand landmarks and provides the x, y coordinates, and depth information necessary to simulate keystrokes when fingers are brought together.

- **PyVDA**: This Python module allows the system to interact with Windows Virtual Desktops. By integrating desktop navigation, users can switch between different virtual desktops by using specific hand gestures, further reducing the need for physical peripherals.

- **PyttSX3**: This text-to-speech library provides auditory feedback whenever a key is ”pressed” or a desktop is switched. This feature is particularly useful for visually impaired users or those who prefer additional confirmation of their inputs.

These technologies collectively form a highly efficient system that addresses both the functionality and user experience aspects of virtual keyboard design. By combining these elements, the system achieves high accuracy, responsiveness, and ease of use.

2.1.4 Performance and Accuracy

In terms of accuracy, Sharma et al. report that their system achieves a 95% success rate in recognizing intended key presses. The system's error rate, which is about 5%, typically occurs when the user's fingers do not align properly with the virtual keys or if the system misinterprets a gesture due to poor lighting conditions or rapid movements. The developers introduced a 30-pixel gap threshold for detecting key presses, which they found to be the most versatile distance across different finger sizes, ensuring that most users could interact with the system comfortably. Additionally, a 0.2-second delay between consecutive clicks was added to prevent unintended double inputs, improving overall usability.

2.1.5 Limitations and Future Scope

Despite the significant advancements made in gesture-based input systems, some challenges remain. The current system's performance can degrade in low-light environments, as the accuracy of hand detection is heavily reliant on the quality of the video feed. The authors propose making the hand detection more robust to varying lighting conditions as a potential area for improvement in future iterations.

Furthermore, the system does not yet support multi-user functionality. Adding the capability to track multiple users simultaneously using input from multiple webcams would expand its applicability in collaborative environments, such as shared workspaces or educational settings.

Other future improvements include adding features like mouse cursor control, and trackpad functionality, and expanding the number of available keys on the virtual keyboard. These enhancements would make the system even more versatile, allowing it to replace a wider range of traditional input devices.

2.1.6 Conclusion

The development of gesture based virtual input systems represents an important evolution in human-computer interaction, particularly in light of the growing need for contactless technologies. The work by Sharma et al. (2022) builds on decades of research in this field, leveraging modern technologies like OpenCV and MediaPipe to create a practical, efficient, and user-friendly virtual keyboard system [2]. By addressing the limitations of earlier systems such as the need for specialized hardware or complex setups the proposed system offers a viable solution for real-world applications, particularly in the context of public health concerns and ergonomic design.

This review highlights the key contributions that have shaped the current state of virtual input systems and emphasizes the potential for further innovation in this space, particularly as machine learning and computer vision technologies continue to advance.

2.2 Voice Guided, Gesture Controlled Virtual Mouse

The study of gesture based virtual systems has evolved significantly over recent years, primarily in response to the growing demand for natural and intuitive human computer interaction (HCI). With the emergence of touchless technologies, particularly amid the global pandemic, gesture controlled input systems have gained traction as they offer a hygienic and efficient alternative to traditional peripherals. The paper "Voice Guided, Gesture Controlled Virtual Mouse" [7] delves into the development of a virtual mouse system that integrates hand gestures and voice commands, using affordable and accessible technology like webcams and machine learning algorithms. This literature review discusses the advancements, key contributions, and technological improvements that have shaped the field of virtual input systems, with a particular focus on gesture recognition and voice assisted control.

2.2.1 Background of Human-Computer Interaction and Gesture Control

The field of Human-Computer Interaction (HCI) has seen extensive research into gesture-based systems as an alternative to traditional input devices like the mouse and keyboard. The drive to develop more natural and contactless methods of interaction has

led to innovations in hand gesture recognition and virtual control systems. These systems leverage advancements in computer vision, machine learning, and image processing to track hand movements and translate them into commands that mimic mouse or keyboard inputs.

One of the earliest contributors to this field was Segen et al. (1999), who proposed a system using shadow gestures to detect hand movements [3], laying the groundwork for future gesture recognition systems. However, this approach was limited by lighting conditions, which affected the clarity of the shadow and, consequently, the accuracy of gesture detection. As technology progressed, researchers shifted towards more sophisticated methods, including hand tracking and fingertip detection, to improve the reliability and accuracy of virtual control systems.

In recent years, the field has seen the integration of machine learning algorithms and frameworks like MediaPipe and OpenCV, which offer real-time hand detection and tracking with high accuracy. These technologies have enabled developers to create virtual input systems that are more responsive, adaptable to different environments, and capable of recognizing complex gestures.

2.2.2 Gesture Recognition and Virtual Mouse Systems

Several studies have attempted to replace physical mouse operations with hand gesture recognition, with varying degrees of success. Traditional methods, such as those involving the use of gloves or colored markers, provided a basic level of hand tracking but were often cumbersome for users. For example, dongre et al. (2020) [8] proposed a virtual mouse interface that relied on color-based fingertip detection, using convex hull algorithms to track hand gestures. Although effective in controlled environments, the use of color markers or gloves was not practical for everyday use due to limitations in accuracy and user comfort.

Varun et al. (2019) explored hand gesture recognition as a means to control computer systems [9], demonstrating the potential of HCI systems to integrate biometric technologies. Their work focused on using hand gestures for basic operations like cursor movement and clicking, which are essential components of a virtual mouse system. This approach, however, was limited by the complexity of gesture recognition in dynamic environments, where lighting and background variations could affect detection accuracy.

In another study, Vinay Pasi et al. (2016) proposed a gesture-based cursor control system using colored bands worn on the user's fingers [10]. While the system successfully translated gestures into mouse actions, it required multiple colors for different operations, which complicated the user experience. Similarly, Reddy et al. (2020) developed a system for virtual mouse control using hand gestures recognized through color markers [11], but their approach was limited to a small set of mouse operations. These early attempts highlighted the need for more seamless, marker-free solutions that could operate effectively in real-world conditions.

The paper [12] by Adluri et al. (2023) takes a different approach by eliminating the need for external markers or gloves altogether. Instead, their virtual mouse system relies solely on a webcam and machine learning algorithms to detect hand gestures and control mouse functions like scrolling, clicking, and pointer movement. This system leverages MediaPipe, a machine learning library developed by Google, which provides real-time hand tracking and gesture recognition capabilities. By using fingertip detection and Euclidean distance calculations, the system can determine whether a click or drag action is being performed based on the relative positions of the user's fingers.

2.2.3 Integration of Voice Assistance in Virtual Systems

While gesture-based systems have shown great potential in enhancing HCI, combining them with voice assistants has further improved their functionality and user experience. Voice assistants provide an additional layer of interactivity, allowing users to issue commands verbally, thus reducing the reliance on physical gestures for more complex tasks.

Voice assistants, such as those powered by Microsoft's Speech Application Programming Interface (SAPI), enable virtual systems to interpret spoken commands and execute tasks based on natural language input. This integration is particularly beneficial for users with physical disabilities or in scenarios where hands-free operation is preferred. PyttSX3, a Python-based text-to-speech library, is commonly used in these systems to provide auditory feedback, enhancing the overall user experience by confirming that commands have been understood and executed.

The work by Dudhapachare et al.integrates a voice assistant into their gesture-controlled virtual mouse system to enable tasks such as volume control, brightness

adjustment, and application navigation. By combining voice and gesture inputs, the system offers a more versatile and intuitive user experience, making it suitable for a wide range of applications, from everyday computing to specialized use cases like gaming and virtual prototyping.

2.2.4 Technological Contributions and Frameworks

The development of gesture-controlled virtual systems has been greatly facilitated by advancements in machine learning and computer vision frameworks. The use of OpenCV for image processing and MediaPipe for hand tracking has become standard in the field, enabling real-time gesture recognition with minimal latency.

OpenCV is widely used for tasks such as object detection, image enhancement, and real-time video analysis. In virtual mouse systems, it is employed to capture hand movements through a webcam, process the video frames, and identify the positions of the user's fingers. MediaPipe complements this by providing pre-trained machine learning models that can detect and track hand landmarks, allowing for the accurate recognition of gestures like clicks, drags, and scrolls

By utilizing these technologies, the virtual mouse system developed by Dudhapachare et al. achieves a high level of accuracy (reported at 99%) in detecting mouse operations such as left-clicks, right-clicks, and pointer movements. The system's accuracy and performance in various lighting conditions further demonstrate the robustness of modern machine learning algorithms in handling real-world variability, making these systems viable alternatives to traditional input devices.

2.2.5 Challenges and Future Directions

Despite the advancements made in gesture-based input systems, several challenges remain. One of the primary issues is the system's reliance on optimal lighting conditions for accurate gesture detection. Variations in lighting, such as low light or overly bright environments, can affect the accuracy of hand tracking, leading to misinterpreted gestures. Additionally, background noise in the video feed can interfere with the system's ability to isolate hand movements, reducing overall performance.

Another challenge lies in the precision of certain gestures, particularly those requiring fine motor control, such as right-clicks. Dudhapachare et al. acknowledge that the

right-click gesture is more difficult for the system to detect accurately, due to the subtle differences in finger positioning required for the action. Addressing this issue through improved gesture recognition algorithms or alternative input methods could further enhance the system's usability.

Future research in this field could explore the integration of multi-user support, enabling multiple users to interact with the system simultaneously using different webcams. Additionally, expanding the system's functionality to include multi-touch gestures and 3D gesture recognition could open up new possibilities for virtual input systems in fields such as virtual reality and augmented reality.

2.2.6 Conclusion

The development of gesture-controlled virtual systems has significantly advanced the field of human-computer interaction, offering intuitive, touchless alternatives to traditional input devices. The work of Dudhapachare et al. builds on previous research by integrating hand gesture recognition with voice assistance to create a versatile and highly accurate virtual mouse system. By utilizing modern machine learning frameworks like MediaPipe and OpenCV, the system achieves impressive accuracy and usability in real-world conditions. While challenges such as lighting variability and gesture precision remain, the progress made in this field demonstrates the potential for gesture-based systems to become mainstream tools in both personal and professional computing environments.

Chapter 3

Proposed System

This chapter details the development of a virtual keyboard system that enables contactless typing through hand gestures, processed using computer vision and machine learning techniques. The system is designed to capture hand and finger movements via a camera, converting these into keystroke predictions with the aid of Mediapipe for gesture recognition and a neural network model trained with the Multi-Layer Perceptron (MLP) algorithm. By accurately identifying and registering each keystroke, the system offers a hygienic, efficient, and adaptable typing alternative, especially suited for mixed-reality applications and shared environments. The sections that follow cover the key features, advantages, feasibility, and functionality of this innovative typing solution.

3.1 System Overview

The proposed system is a virtual keyboard that allows users to type messages by detecting hand gestures on a flat surface. This system leverages a camera to capture the user's finger movements and employs computer vision algorithms to interpret which keys are pressed based on finger positions and motions. The primary objective of this system is to enable contactless typing, promoting hygiene and accessibility, especially in shared environments or mixed-reality settings.

The working process begins with a camera capturing frames as the user types on a flat surface. These frames are processed in real-time to identify hand and finger landmarks, which are then passed through a neural network model trained to recognize specific keystrokes based on the detected gestures. This model, implemented using the Multi-

Layer Perceptron (MLP) algorithm, accurately classifies the keystroke corresponding to each gesture. The predicted keystrokes are registered to the system using the pyAutoGUI.press() function, allowing the typed message to appear on the screen.

3.2 Key Features of the System

The proposed virtual keyboard system includes several distinctive features that enable effective and accurate typing experiences:

3.2.1 Gesture Recognition with Mediapipe

This system utilizes Mediapipe's Hand class to process captured frames, specifically using the hand.process() function. The function detects finger landmarks and generates precise x, y, and z coordinates, which represent finger angles and distances between fingertip positions. These parameters are essential inputs to the neural network model, enabling it to differentiate between distinct keystrokes based on specific hand gestures.

3.2.2 Neural Network Model for Key Classification

The core of this system is a neural network model, trained using the MLP (Multi-Layer Perceptron) algorithm. The model receives over 75 input features, including calculated angles and distances derived from the detected finger positions. The output is a classification result representing the pressed key, with 26 different classes corresponding to each letter in the English alphabet.

3.2.3 Real-time Keystroke Registration

Once a keystroke is predicted, it is registered using the pyAutoGUI.press() function. This function enables the system to seamlessly integrate the predicted keystrokes with other applications, allowing real-time display of the typed message on the screen. The system, therefore, functions as a substitute for a physical keyboard, providing a smooth and uninterrupted typing experience.

3.2.4 Custom Dataset for Model Training

The dataset used to train the MLP model was generated by capturing finger landmarks and calculating angles and distances between fingertips for various keystrokes. This data is stored in a CSV file with over 80 columns, each representing a specific feature, to ensure a comprehensive representation of finger gestures. This dataset forms the basis for the model's ability to accurately classify keystrokes.

3.3 Advantages of the System

The virtual keyboard system offers a range of advantages, addressing specific needs in user interaction and environmental adaptation:

- **Hygienic Typing Solution**: By eliminating the need for a physical keyboard, this system significantly reduces the spread of germs, especially in public or shared spaces.
- **Accessible in Mixed-Reality Environments**: This system's compatibility with mixed-reality applications opens new possibilities for immersive experiences where physical keyboards are impractical.
- **Customizability and Adaptability**: The neural network model can be retrained or adjusted for specific gesture requirements, making it adaptable to various typing needs.
- **Enhanced Portability**: With only a camera and software requirements, the system can be used on portable devices, providing a flexible solution for on-the-go typing needs.

3.4 Feasibility Study of the System

The feasibility study assesses the technical, operational, and economic viability of implementing this virtual keyboard system in real-world scenarios.

3.4.1 Technical Feasibility

The system is technically feasible due to the availability of sophisticated yet accessible libraries like Mediapipe for gesture recognition and pyAutoGUI for keystroke registration. The neural network, trained using the MLP algorithm, is capable of handling real-time classification tasks, allowing the system to perform accurately and efficiently. Additionally, the system's reliance on a camera makes it compatible with a variety of devices, including laptops, tablets, and smartphones.

3.4.2 Operational Feasibility

Operational feasibility is achieved by the intuitive nature of gesture-based typing, which users can quickly adapt to. The contactless interaction and the ease of setup requiring only a camera ensure that users can implement this system in multiple environments, such as workspaces, medical facilities, and education centers, where minimal contact with shared devices is preferable.

3.4.3 Economic Feasibility

From an economic standpoint, the system presents a cost-effective alternative to specialized touchless interfaces or holographic displays. By relying on widely available hardware (a camera) and open-source libraries, the system minimizes additional hardware costs, making it a budget-friendly solution suitable for individual users and organizations alike.

3.5 System Functionality and Workflow

The workflow of the proposed virtual keyboard system involves several steps, each contributing to the smooth functioning of gesture-based typing:

- **Frame Capture**: The camera captures frames as the user performs typing gestures on a surface.
- **Hand Landmark Detection**: The captured frames are processed by Mediapipe's hand.process() function to detect finger landmarks with precise x, y, and z coordinates.

- **Feature Calculation**: Finger angles and distances between fingertip positions are calculated from the detected landmarks. These features serve as essential inputs for predicting the keystroke.
- **Keystroke Prediction**: The neural network model processes the calculated features to classify the corresponding key based on trained data patterns.
- **Key Registration**: The predicted keystroke is registered through pyAutoGUI.press(), resulting in a real-time display of the typed character on the screen.
- **Display Output**: The system continuously updates the displayed message, showing the text typed by the user in real-time.

This workflow enables the system to function efficiently, delivering a seamless typing experience using hand gestures.

3.6 Future Enhancements and Improvements

While the current system provides a robust typing solution, several enhancements could further improve its accuracy and usability:

- **Enhanced Gesture Recognition**: Incorporating additional hand landmarks or multi-finger gesture recognition can improve the accuracy of key detection.
- **Language Support**: Extending the model to recognize symbols and other language alphabets would enhance the system's versatility.
- **Integration with AR/VR**: For a fully immersive typing experience, the system can be optimized for augmented reality (AR) and virtual reality (VR) applications.

3.7 Conclusion

In summary, the proposed virtual keyboard system presents a practical and innovative approach to contactless typing by integrating computer vision and machine learning. Through real-time hand gesture recognition with Mediapipe and keystroke prediction using an MLP-based neural network, the system demonstrates a feasible and hygienic alternative to traditional keyboards.

Chapter 4

Designs for Ergonomic Virtual Keyboard

These design elements are essential for ensuring a clear understanding of both the technical and user-focused aspects of the Ergonomic Virtual Keyboard system. The architectural layout not only defines the primary components but also outlines their interactions, allowing for efficient system functionality and data management. The DFDs break down complex processes into simplified visual representations, making it easier to track the flow of information across different modules. Additionally, the Use Case Diagram highlights typical user scenarios, ensuring the system design aligns with real-world interactions and user needs.

4.1 Architecture Diagram

This architecture ensures seamless communication between hardware and software components, enabling efficient processing and response times crucial for a smooth user experience. By leveraging machine learning algorithms, the system adapts to various gesture inputs with high accuracy, allowing for intuitive and reliable virtual typing. Furthermore, the modular design facilitates future enhancements, making it flexible for integration with additional features or improved models.

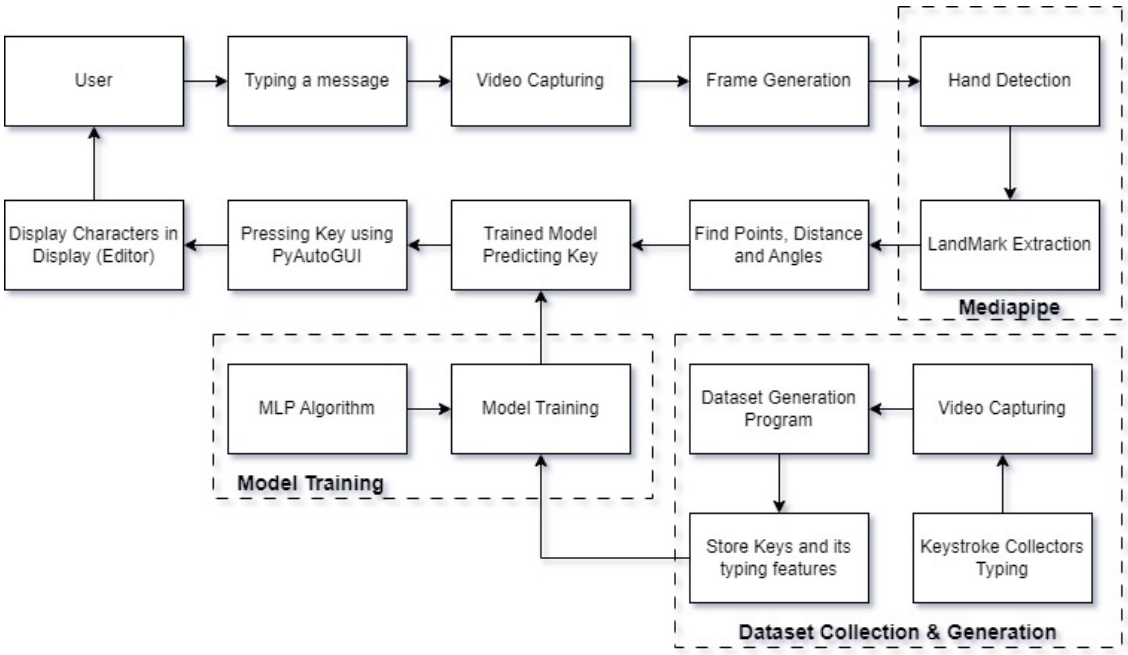

Figure 4.1: Architecture Diagram

4.1.1 User Interaction

- **User**: This represents the person interacting with the system, specifically typing a message that will be processed through the virtual keyboard.
- **Typing a Message**: The user performs hand and finger movements corresponding to typing, which is captured by the system.

4.1.2 Video Capturing and Frame Generation

- **Video Capturing**: The system continuously captures video of the user's hands as they type. This video is a primary input that allows the system to observe and interpret hand gestures.
- **Frame Generation**: The captured video is processed to extract frames. Each frame is individually analyzed, to ensure that hand movements are detected accurately in real time.

4.1.3 Hand Detection and Feature Extraction (Mediapipe)

- **Hand Detection**: Using computer vision, the system detects the user's hand in each frame. This is essential for isolating hand movements from the background

and focusing on relevant gestures.

- **Landmark Extraction**: Mediapipe, a powerful machine learning library, is utilized to detect landmarks on the hand, such as knuckle and fingertip positions. These landmarks provide reference points that help in understanding gestures.

- **Find Points, Distance, and Angles**: Based on the landmarks, the system calculates distances and angles between points on the hand. These metrics are essential for identifying specific finger positions associated with typing gestures.

4.1.4 Model Training

- **Dataset Collection & Generation**: This phase involves capturing a large set of hand gestures that correspond to different keystrokes.

- **Dataset Generation Program**: A program specifically designed to collect and organize the dataset.

- **Keystroke Collectors Typing & Storing Keystrokes and Features**: Collectors (testers or tools) simulate typing gestures, which are recorded along with their associated key features, such as position and movement.

- **MLP Algorithm (Multilayer Perceptron)**: A neural network model is trained on the dataset to recognize patterns in the collected data, associating hand positions and movements with specific keys.

- **Model Training**: Using the MLP algorithm, the system learns to map gesture features to keystrokes, creating a predictive model for real-time typing.

4.1.5 Key Prediction and Typing Simulation

- **Trained Model Predicting Key**: Once trained, the model can predict the most likely key based on the detected hand gesture in each frame. This prediction is updated as the user's hand position changes.

- **Pressing Key Using PyAutoGUI**: The system utilizes PyAutoGUI, a Python library that enables programmatic control of the keyboard and mouse. Here,

PyAutoGUI is used to simulate actual keystrokes based on predictions, allowing for virtual typing.

- **Display Characters in Display (Editor)**: The predicted keystrokes are displayed on the screen, mimicking a typing interface such as a text editor where the user can see the typed characters.

4.2 Data Flow Diagram (DFD) Level 0

The DFD Level 0 provides a high-level view of the system, illustrating the entire process as a single function with data interactions between the user and the system. This diagram highlights key interactions, providing a clear overview while avoiding the complexity of internal processes.

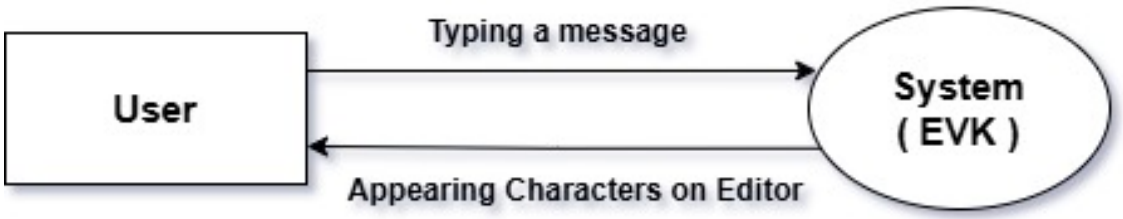

Figure 4.2: DFD Level 0

4.2.1 User

The user is the primary entity interacting with the system. In this context, the user represents the individual typing messages using the virtual keyboard system.

4.2.2 System (EVK)

The ”EVK” (Electronic Virtual Keyboard) system processes the user’s typing actions. It captures the gestures, recognizes them as keystrokes, and outputs the typed characters to the display.

4.2.3 Data Flow

- **Typing a Message**: This flow represents the input from the user, where they type messages through gestures. The system receives and interprets these gestures to

identify the intended keystrokes.

- **Appearing Characters on Editor**: After processing the gestures and determining the appropriate keystrokes, the system displays the resulting characters in an editor. This output allows the user to see the text they have ”typed” on the virtual keyboard.

4.3 Data Flow Diagram (DFD) Level 1

The Level 1 DFD expands on the Context Diagram, illustrating the internal processes of the system in greater detail. It shows how data moves through various subsystems, allowing for a deeper understanding of the core functions.

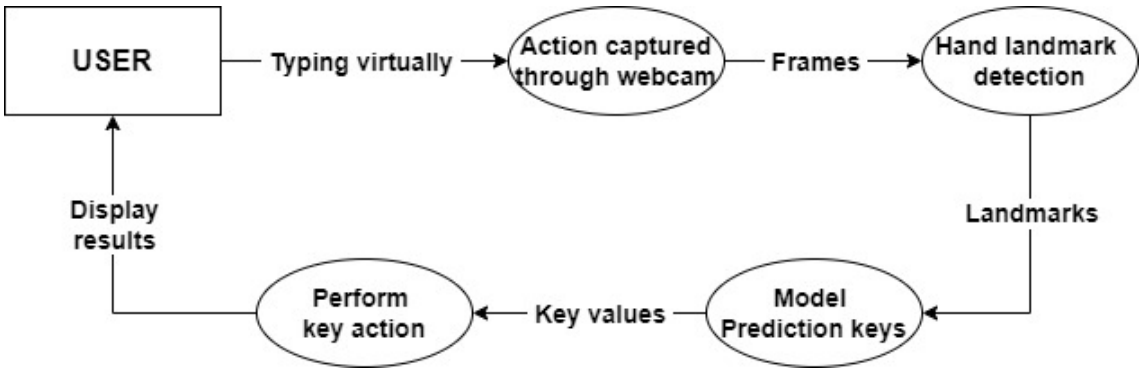

Figure 4.3: DFD Level 1

4.3.1 User

The user virtually types by performing gestures, which are captured by the system as input. This step is similar to the user interaction seen in Level 0 but with a focus on the input capture.

4.3.2 Action Captured Through Webcam

The user’s hand and finger movements are recorded via a webcam. This component represents the initial capture of gestures, with each gesture translated into frames for further processing.

4.3.3 Hand Landmark Detection

- This process involves identifying and marking key points (landmarks) on the user's hand, such as finger joints and fingertips. These landmarks are essential for recognizing specific gestures.
- Frames to Landmarks: The frames captured from the webcam are processed to extract hand landmarks, which will be used in subsequent steps.

4.3.4 Model Prediction Keys

- Once landmarks are detected, the trained model analyzes these positions to predict the intended keystroke.
- Landmarks to Key Values: The landmarks serve as input to the prediction model, which translates them into key values (specific characters or actions).

4.3.5 Perform Key Action

- Based on the predicted key, the system performs the corresponding keystroke action. This may involve simulating a key press to register the predicted character in an editor.
- Key Values to Key Action: The predicted values are processed to perform actions, allowing the user to ”type” virtually.

4.3.6 Display Results

Finally, the typed characters appear on the display, providing feedback to the user. This allows the user to see the results of their typing gestures on the screen.

4.4 Data Flow Diagram (DFD) Level 2

DFD Level 2 provides an even more detailed breakdown of the system's functions, specifically focusing on the individual tasks within each sub-process. It explains how each stage in the system operates in greater detail, showing how data is manipulated and processed at a finer level of granularity.

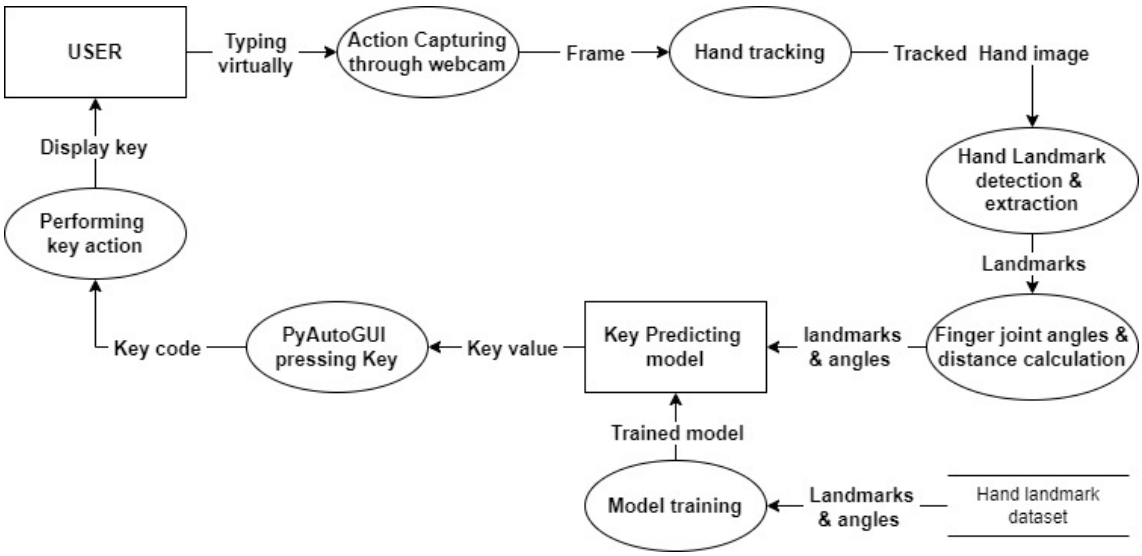

Figure 4.4: DFD Level 2

4.4.1 User

The user types virtually using hand gestures, which serve as the system's primary input.

4.4.2 Action Capturing through Webcam

The webcam captures the user's gestures, translating them into frames that will be processed by the system.

4.4.3 Hand Tracking

- This process identifies and isolates the user's hand from the captured frames, ensuring that only relevant parts of the image are processed further.
- Frame to Tracked Hand Image: The frames are refined to focus on the user's hand, improving the accuracy of the landmark detection.

4.4.4 Hand Landmark Detection & Extraction

- In this process, the system detects landmarks on the hand, marking key points (e.g., fingertips, knuckles). These landmarks serve as the foundation for interpreting gestures.

- Tracked Hand Image to Landmarks: The tracked hand image is analyzed to locate these specific points.

4.4.5 Finger Joint Angles & Distance Calculation

- Based on the landmarks, the system calculates angles and distances between finger joints. This data is essential for gesture recognition, as different key presses correspond to different hand configurations.
- Landmarks to Angles and Distances: This information refines the gesture data, helping the model recognize subtle differences in hand movements.

4.4.6 Key Predicting Model

- Using the calculated angles and distances, the key prediction model identifies the most likely keystroke associated with the gesture.
- Landmarks & Angles to Key Value: The prediction model leverages these metrics to determine the intended key with high accuracy.
- Model Training: This process involves training the prediction model on a dataset of landmarks and angles to recognize typing gestures effectively. The trained model is then used to predict keys based on input gestures.

4.4.7 PyAutoGUI Pressing Key

- PyAutoGUI is used to simulate the pressing of the predicted key, enabling the system to register the keystroke as if it were typed on a physical keyboard.
- Key Value to Key Code: The key value predicted by the model is translated into a key code that PyAutoGUI uses to perform the action.

4.4.8 Performing Key Action

The key action is executed, resulting in the display of the typed character on the screen, allowing the user to view the output of their gestures.

4.4.9 Display Key

The typed character is displayed on the virtual editor, providing immediate feedback to the user.

4.5 Use Case Diagram

The use case diagram provides an overview of the primary interactions between the User and the System in the context of the virtual keyboard project. It illustrates the main functionalities that the system provides to the user and highlights the system's core processes involved in interpreting and displaying gestures as keystrokes.

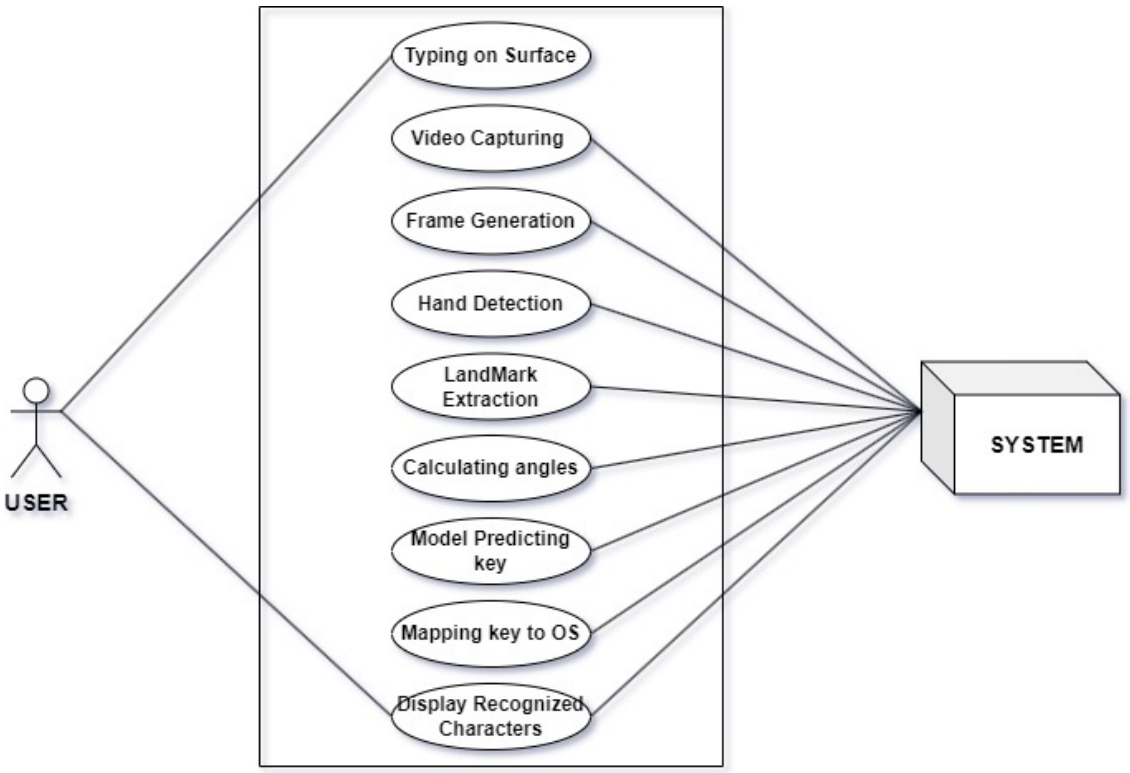

Figure 4.5: Use Case Diagram

4.5.1 Actors

- **User**: The user is the primary actor in the system, performing typing gestures that the system interprets as keystrokes.
- **System**: The system processes the user's input gestures and displays the recognized characters, simulating a typing experience.

4.5.2 Use Cases

- **Typing on Surface**: The user performs gestures on a surface to mimic typing. This is the initial interaction and serves as the input action for the virtual keyboard system.
- **Video Capturing**: The system captures the user's gestures through a webcam, generating frames that will be processed further.
- **Frame Generation**: The captured video feed is broken down into individual frames, allowing the system to analyze gestures frame-by-frame.
- **Hand Detection**: The system detects the user's hand in the captured frames, isolating it for gesture analysis.
- **Landmark Extraction**: Key points on the hand (landmarks) are extracted to identify the positions and movements of fingers and joints, which are crucial for gesture recognition.
- **Calculating Angles**: The system calculates angles between joints, aiding in recognizing different hand postures for various keys.
- **Model Predicting Key**: The trained model uses the extracted landmarks and angles to predict which key the user intends to press based on the detected gesture.
- **Mapping Key to OS**: The system translates the predicted key into a format recognized by the operating system, effectively simulating a keystroke.
- **Display Recognized Characters**: Finally, the system displays the recognized characters in an editor or output interface, allowing the user to see their typed input.

In conclusion, the designs of the Ergonomic Virtual Keyboard provide a comprehensive framework for how the system captures, processes, and interprets hand gestures, turning them into functional input. The architecture, DFDs, and use case diagrams collectively outline how the system operates, from the high-level design to the detailed steps of data processing and user interaction.

Chapter 5

Gantt Chart

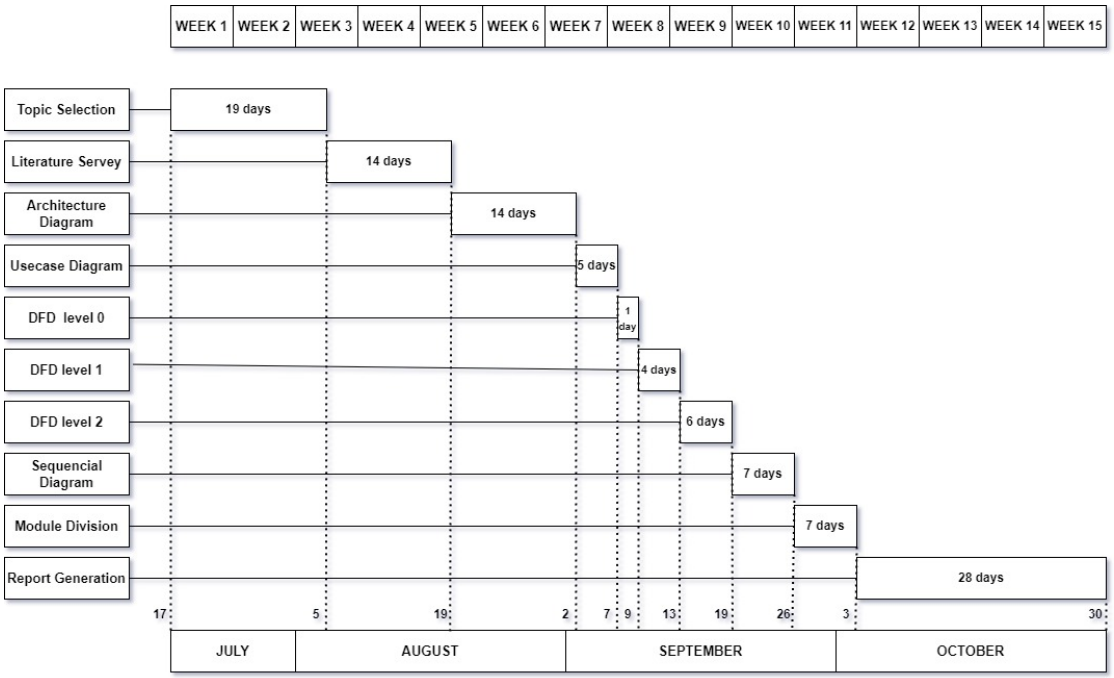

Figure 5.1: Gantt Chart

1. **Project Planning and Initial Research**: This phase usually focuses on identifying the project's scope and objectives, as well as researching similar systems and gathering relevant literature. In your case, this likely included exploring previous works on gesture-based interfaces and virtual keyboards.

2. **Dataset Collection and Feature Expansion**: During this stage, the team collects data to train the system on recognizing hand movements. This data might include various hand gestures, positions, and interactions, which will help improve the accuracy of the virtual keyboard. This is often a time-intensive phase as the quality of the dataset directly

impacts system performance.

3. **Algorithm Implementation and Training**: Here, the team designs and implements the machine learning algorithms that process hand gestures into keyboard inputs. Machine learning models, possibly including MediaPipe and MLP, are trained to recognize gestures and predict keystrokes. This phase includes testing various model architectures to find the most efficient one.

4. **System Testing and Quality Assurance**: After implementing the algorithms, the system undergoes rigorous testing to ensure it meets the design requirements. Testing focuses on accuracy, user-friendliness, and responsiveness. Feedback from this stage is used to refine the system further.

5. **Interface Implementation and Validation**: This stage involves building the user interface and integrating it with the gesture recognition system. The interface should be intuitive, allowing users to use the virtual keyboard naturally. Validation ensures that the interface is fully functional and user-friendly.

6.**Final Review and Documentation**: In this last phase, the team reviews the system for any final adjustments and documents the entire project. The documentation includes explanations of the system design, algorithm choices, and interface features.

Chapter 6

Conclusion

The Ergonomic Virtual Keyboard project presents a significant advancement in the field of human-computer interaction, leveraging state-of-the-art computer vision and machine learning technologies to create a more intuitive, flexible, and ergonomic typing solution. By shifting away from traditional physical keyboards and inefficient on-screen keyboards, this system introduces a gesture-based virtual keyboard that enables users to type through natural hand and finger movements captured by a camera or sensor. This innovative approach not only improves typing efficiency and comfort but also addresses several key challenges associated with existing input methods.

One of the most notable aspects of the proposed system is its ability to detect and track movements from both hands, allowing for a more natural and familiar typing experience. Traditional virtual keyboards often limit the user to single-hand gestures or require physical contact with the device, but the Ergonomic Virtual Keyboard overcomes these limitations by enabling full-hand interactions. This ensures that users can achieve higher typing speeds and greater accuracy, closely mimicking the feel of traditional touch typing without the need for physical keys.

The precision of keystroke prediction is another key strength of the system, made possible through the use of machine learning algorithms. By analyzing the trajectories and patterns of fingertip movements, the system can accurately predict intended keystrokes in real-time, reducing the likelihood of errors and improving the overall user experience. This high level of accuracy is essential for a virtual typing system, as it ensures that users can rely on the system for tasks that require extensive and accurate text input, such as document creation, coding, or communication.

In addition to its functional benefits, the system offers several significant advantages from an ergonomic perspective. The contactless nature of the virtual keyboard reduces the strain on the hands and wrists that often results from prolonged use of physical keyboards, helping to prevent repetitive strain injuries (RSIs) and other health issues. This is particularly important in professional settings where users spend long hours typing, as the system allows for more natural hand movements and a more comfortable posture. By providing an ergonomic alternative to traditional keyboards, the system can help to improve user well-being and productivity over time.

Another important feature of the Ergonomic Virtual Keyboard is its portability and versatility. The system can be easily paired with a wide range of devices, from desktop computers to laptops and even mobile devices, making it suitable for use in various environments. Whether users are working at a desk, on the go, or in a mixed reality (MR) or augmented reality (AR) environment, the virtual keyboard offers a practical and convenient input solution. Its compatibility with MR/AR applications further enhances its versatility, allowing users to interact with digital content in immersive environments without the need for physical peripherals.

Furthermore, the system's contactless design is particularly relevant in the context of modern public and shared spaces. In an era where hygiene and safety are of increasing concern, particularly during the transmission of contagious diseases, the ability to interact with a device without physical contact offers a significant advantage. Users can type efficiently in public settings without having to touch shared surfaces, reducing the risk of contamination and promoting a cleaner, safer work environment.

In terms of technical feasibility, the project demonstrates that the required technologies such as computer vision, machine learning, and gesture recognition are mature and capable of delivering the desired functionality. By utilizing widely available hardware, such as webcams and sensors, along with advanced software frameworks like MediaPipe and OpenCV, the system is technically sound and scalable. The integration of these technologies ensures that the virtual keyboard can function reliably and efficiently, with minimal latency and high accuracy.

Economically, the system is both cost-effective and scalable. With low hardware costs and the potential for broad market adoption, the project presents a financially viable solution. The system can be deployed across various industries, including healthcare, education, finance, and technology, where ergonomic, flexible, and efficient typing

solutions are in high demand. Its ability to integrate with both personal and professional devices further enhances its commercial appeal, offering users a solution that is both practical and affordable.

Operationally, the virtual keyboard is designed to be user-friendly and easy to integrate into existing workflows. The intuitive gesture-based interface ensures that users can quickly learn and adapt to the system, with minimal training required. Additionally, the system's adaptability to different environments whether in an office, public space, or immersive virtual reality makes it a highly practical solution for a wide range of use cases.

In conclusion, the Ergonomic Virtual Keyboard represents a forward-thinking solution to the limitations of traditional input methods. By combining the latest advancements in computer vision and machine learning with an emphasis on user comfort and efficiency, the system addresses critical issues related to space, ergonomics, hygiene, and typing efficiency. It offers a modern, adaptable, and ergonomic alternative to physical and on-screen keyboards, with broad applications in both personal and professional contexts. As technology continues to evolve and the demand for more flexible, contactless input methods grows, the Ergonomic Virtual Keyboard is poised to become a key player in the future of human-computer interaction, offering users a seamless, comfortable, and efficient typing experience in a wide variety of environments.

References

[1] M. Nazeer, G. Akshita, A. Priyadharshini, C. C. Bala Krishna, A. Anusha, and A. Amreen, "Gesture controlled virtual mouse and keyboard using opencv," *in 2023 International Conference on Emerging Techniques in Computational Intelligence (ICETCI)*, pp. 199–206, 2023.

[2] H. Sharma, R. Saxena, S. Kumar, A. Saini, M. Saad, and M. Faraz, "Virtual hands: Real time keyboard, desktop & application navigation using gestures," *in 2022 International Conference on Fourth Industrial Revolution Based Technology and Practices (ICFIRTP)*, pp. 262–267, 2022.

[3] J. Segen and S. Kumar, "Shadow gestures: 3d hand pose estimation using a single camera," *in Proceedings. 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149)*, vol. 1, pp. 479–485, 1999.

[4] J. Mantyjarvi, J. Koivumaki, and P. Vuori, "Keystroke recognition for virtual keyboard," *in Proceedings. IEEE International Conference on Multimedia and Expo*, vol. 2, pp. 429–432, 2002.

[5] J. Hu, G. Li, X. Xie, Z. Lv, and Z. Wang, "Bare-fingers touch detection by the button's distortion in a projector–camera system," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 4, pp. 566–575, 2014.

[6] K. Jadhav, F. Jagirdar, S. Maine, and J. Shahabadi, "Virtual keyboard and virtual mouse," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 5, no. 5, pp. 318–323, 2016.

[7] R. Dudhapachare, M. Awatade, P. Kakde, N. Vaidya, M. Kapgate, and R. Nakhate, "Voice guided, gesture controlled virtual mouse," *in 2023 4th International Conference for Emerging Technology (INCET)*, pp. 1–6, 2023.

[8] A. Dongre, R. Pinto, A. Patkar, and M. Lopes, "Computer cursor control using eye and face gestures," *in 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–6, 2020.

[9] K. S. Varun, I. Puneeth, and T. P. Jacob, "Virtual mouse implementation using open cv," *in 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 435–438, 2019.

[10] K. Vinay, K. Pooja, and S. Saurabh, "Cursor control using hand gestures," *International Journal of Critical Accounting*, vol. 975, pp. 25–29, 2016.

[11] V. V. Reddy, T. Dhyanchand, G. V. Krishna, and S. Maheshwaram, "Virtual mouse control using colored finger tips and hand gesture recognition," *in 2020 IEEE-HYDCON*, pp. 1–5, 2020.

[12] V. L. Adluri, P. Kadiyala, S. Gopu, and S. Jangiti, "Virtual mouse with hand gestures using machine learning," *in 2023 International Conference on Sustainable Communication Networks and Application (ICSCNA)*, pp. 1564–1568, 2023.