A PROJECT REPORT ON

GESTURE DRIVEN PRESENTATION CONTROL

Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, KAKINADA

For Partial Fulfilment of Award of the Degree of

BACHELOR OF TECHNOLOGY
IN
CSE-ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Submitted By

G. Surendra Reddy	**20X41A4215**
R. Usha Sri	**20X41A4245**
R. Hemanth	**20X41A4246**
P. Mohana Kalyan	**21X45A4203**

Under the esteemed guidance of
Ms. D. Sirisha
Assistant Professor, Department of CSE- Artificial Intelligence and Machine Learning

DEPARTMENT OF CSE- ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING
S.R.K INSTITUTE OF TECHNOLOGY

(Approved AICTE, New Delhi &Affiliated To JNTU, Kakinada)

(An Iso 9001:2015 Certified Institution & Accredited by NAAC With 'A' Grade)

Enikepadu, Vijayawada – 521108.

April 2024

S.R.K INSTITUTE OF TECHNOLOGY

(Approved AICTE, New Delhi &Affiliated To JNTU, Kakinada)

(An Iso 9001:2015 Certified Institution & Accredited by NAAC With 'A' Grade)

Enikepadu, Vijayawada – 521108.

DEPARTMENT OF CSE-ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

CERTIFICATE

This is to certify that this project report entitled **"GESTURE DRIVEN PRESENTATION CONTROL"** is the bonafide work of **G. Surendra Reddy (20X41A4215), R. Usha Sri (20X41A4245), R. Hemanth (20X41A4246), P. Mohana Kalyan (21X45A4203)** in partial fulfillment of the requirements for the award of the graduate degree in **BACHELOR OF TECHNOLOGY** during the academic year 2023-2024. This Work has carried out under our supervision and guidance.

(Ms. D. Sirisha)
Signature of the Guide

(Dr. D. Anusha)
Signature of the HOD

Signature of the External Examiner

DECLARATION

We G. Surendra Reddy, R. Usha Sri, R. Hemanth, P. Mohana Kalyan hereby declare that the project report entitled "**GESTURE DRIVEN PRESENTATION CONTROL**" is an original work done in the Department of CSE – Artificial Intelligence and Machine Learning, SRK Institute of Technology, Enikepadu, Vijayawada, during the academic year 2023-2024, in partial fulfillment for the award of the Degree of Bachelor of Technology. We assure you that this project is not submitted to any other College or University.

PROJECT ASSOCIATES

Roll No	Name of the Student	Signature
20X41A4215	G. Surendra Reddy	
20X41A4245	R. Usha Sri	
20X41A4246	R. Hemanth	
21X45A4203	P. Mohana Kalyan	

ACKNOWLEDGEMENT

Firstly, we would like to convey our heart full thanks to the Almighty for the blessings on us to carry out this project work without any disruption.

We are extremely thankful to **Ms. D. Sirisha**, our guide throughout the project. We also thank her for most independence and freedom throughout the given to us during various phases of the project.

We are also thankful for our project coordinator **Dr. D. Anusha**, for their valuable guidance which helped us to bring this project successfully.

We are very much grateful to **Dr. D. Anusha**, H.O.D of C.S.E – Artificial Intelligence and Machine Learning Department, for her valuable guidance which helped us to bring out this project successfully. Her wise approach made us learn the minute details of the subject. Her matured and patient guidance paved away for completing our project with a sense of satisfaction and pleasure.

We are greatly thankful to our principal **Dr. M. Ekambaram Naidu** for his kind support and facilities provided at our campus which helped us to bring out this project successfully.

Finally, we would like to convey our heart full thanks to our Technical Staff, for their guidance and support in every step of this project. We convey our sincere thanks to all the faculty and friends who directly or indirectly helped us with the successful completion of this project.

G. Surendra Reddy **(20X41A4215)**

R. Usha Sri **(20X41A4245)**

R. Hemanth **(20X41A4246)**

P. Mohana Kalyan **(21X45A4203)**

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

This project presents a real-time interactive Presentation Control System that redefines the conventional method of delivering presentations through innovative hand gesture recognition. Utilizing the robust hand tracking capabilities of OpenCV and the precise gesture recognition functionalities of mediapipe, the system empowers users to seamlessly navigate through slides, annotate content, and interact with presentations using intuitive hand movements. By overlaying the live camera feed onto the presentation interface, users can visualize their gestures alongside the slides in real-time, enhancing the interactive experience. The system interprets specific gestures, such as using the little finger to advance to the next slide and the thumb to go back to the previous slide. Additionally, users can utilize the index finger and middle finger to draw annotations, while the index finger alone serves as a pointer. To undo actions, users can gesture with the index, middle, and ring fingers, while raising all fingers clears annotations. This novel approach bridges the gap between traditional presentation tools and modern interactive interfaces, offering a more efficient and engaging way to deliver presentations, particularly in scenarios where conventional input devices may be impractical or cumbersome. With its combination of OpenCV and Mediapipe libraries, the system ensures high accuracy and responsiveness, enabling smooth and intuitive control of presentations through natural hand movements.

Keywords: Open CV, Mediapipe, Hand Gesture Recognition, Presentation Controller.

Chapter 1
INTRODUCTION

1.1 Overview

In the contemporary landscape of communication, presentations serve as crucial tools across diverse sectors, including academia, business, and entertainment. They provide a platform for conveying information, persuading audiences, and fostering engagement. However, amidst the abundance of presentations, this project stands out by initiating a paradigm shift in presentation methodologies.

At the heart of this innovation lies the integration of cutting-edge technologies, particularly real-time hand tracking and gesture recognition. By harnessing the power of these technologies, we not only revolutionize the conventional approach to presentations but also inaugurate a new era of dynamic interaction. Gone are the days of static slideshows controlled by traditional input devices; instead, our system empowers presenters with the ability to navigate through content and interact with their audience using intuitive hand movements.

This departure from traditional methods toward dynamic interaction represents a significant leap forward in presentation delivery. Not only does it enhance engagement by offering a more immersive and interactive experience for the audience, but it also elevates the overall effectiveness of presentations. Presenters can now seamlessly transition between slides, annotate content, and engage with their audience in real-time, fostering a deeper connection and facilitating better information retention.

Moreover, by offering presenters intuitive control through hand gestures, we aim to streamline the delivery process and reduce reliance on cumbersome input devices. This not only enhances the presenter's comfort and confidence but also allows for a more natural and fluid delivery, ultimately resulting in a more impactful and memorable experience for audiences.

In essence, this project represents a bold step forward in the evolution of presentation technology. By embracing innovation and leveraging the latest advancements in computer vision and human-computer interaction, we aim to redefine the way presentations are delivered, experienced, and remembered in the modern era of communication.

1.2 ABOUT THE PROJECT

The development of this real-time interactive Presentation Control System involved several key steps to ensure its effectiveness and usability. Initially, our project team conducted extensive research into computer vision techniques, focusing particularly on hand tracking and gesture recognition algorithms. This laid the foundation for the system's functionality, which would rely on accurately interpreting user hand movements. Our team then proceeded to integrate OpenCV and MediaPipe libraries into the system, harnessing the robust hand tracking capabilities of OpenCV and the precise gesture recognition functionalities of MediaPipe.

Once the foundational software components were in place, we began designing the user interface and interaction flow. This involved mapping out how users would navigate through slides, annotate content, and interact with the presentation using intuitive hand gestures. The goal was to create a seamless and intuitive user experience that would enhance engagement and productivity during presentations. Overlaying the live camera feed onto the presentation interface was a crucial aspect of this design, as it allowed users to visualize their gestures alongside the slides in real-time, providing immediate feedback on their interactions.

The implementation phase involved coding and testing the various features of the system, ensuring that hand gestures were accurately interpreted and translated into the desired actions. Specific gestures, such as using the little finger to advance to the next slide and the thumb to go back to the previous slide, were programmed into the system to provide users with intuitive control over the presentation. Additionally, the ability to draw annotations using the forefinger and middle finger, as well as using the forefinger as a pointer, added further functionality to the system.

Throughout the development process, we conducted thorough testing to identify and address any bugs or issues. This iterative approach allowed for continual refinement and improvement of the system's performance and reliability. Finally, with the combination of OpenCV and MediaPipe libraries, the system achieved high accuracy and responsiveness, enabling smooth and intuitive control of presentations through natural hand movements. This project represents a significant advancement in presentation technology, offering a more efficient and engaging way to deliver presentations in various settings.

1.2.1 What is Gesture :

A gesture is a non-verbal form of communication or expression, often involving physical movements of the body, hands, arms, or face. Gestures can convey a wide range of meanings, emotions, or intentions, and they are an integral part of human interaction and communication. Gestures can be deliberate or subconscious, and they vary greatly across different cultures and contexts. Examples of gestures include waving, nodding, pointing, and making hand signals.

1.2.2 Gesture Recognition :

Gesture recognition is a technology that allows devices to interpret human gestures as commands or inputs. It involves using sensors, cameras, or other input devices to capture and interpret gestures made by humans, such as hand movements, facial expressions, or body motions. These gestures are then analyzed by algorithms to understand their meaning and trigger appropriate responses.

1.2.3 Hand Tracking :

Hand tracking refers to the process of detecting and analyzing the movements and positions of a person's hands in real-time using technology, typically through cameras or sensors. This technology allows devices such as computers, smartphones, or virtual reality headsets to interpret and respond to hand movements without the need for physical controllers or input devices.

Hand tracking technology often relies on computer vision algorithms to recognize and track the key points of the hand, such as the positions of the fingers and palm.

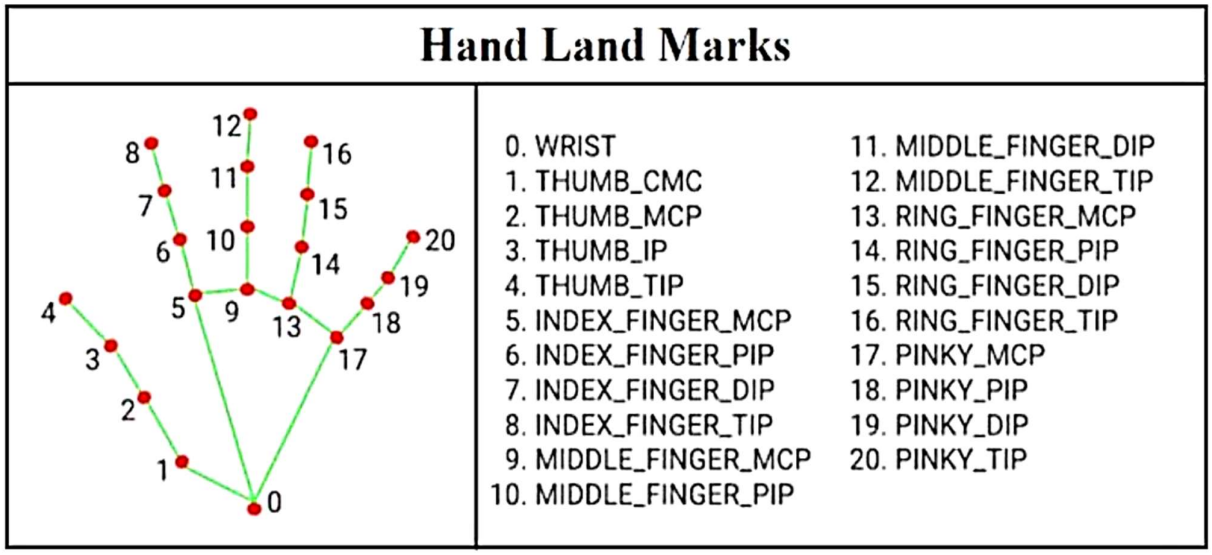

FIG:1.1 Representation of hand landmarks

1.3 Problem Statement:

In the realm of presentation delivery, conventional methods often rely on static interfaces and traditional input devices like keyboards and mice, which can limit interactivity and hinder user engagement. Furthermore, in dynamic presentation environments or scenarios where physical movement is constrained, such as during live demonstrations or in crowded spaces, the use of conventional input devices may prove impractical or disruptive. Additionally, existing gesture recognition systems may exhibit limitations in terms of precision, responsiveness, or ease of use, thus failing to meet the demands of real-time interactive presentation control.

To address these challenges, this project seeks to develop a robust and intuitive real-time interactive Presentation Control System utilizing innovative hand gesture recognition technology. The primary goal is to redefine the conventional approach to delivering presentations by empowering users to navigate slides, annotate content, and interact with presentations seamlessly and intuitively through natural hand movements. By leveraging the advanced hand tracking capabilities of OpenCV and the precise gesture recognition functionalities of mediapipe, the system aims to bridge the gap between traditional presentation tools and modern interactive interfaces.

Specific objectives of the project include: designing and implementing a system architecture capable of real-time hand gesture recognition and presentation control, considering factors such as scalability, efficiency, and ease of integration; developing robust gesture interpretation algorithms capable of accurately recognizing and responding to specific hand movements for tasks such as slide navigation, annotation, and interaction with presentation elements; integrating the live camera feed onto the presentation interface to provide users with visual feedback of their gestures in real-time, enhancing the interactive experience and facilitating gesture-based interaction; ensuring high accuracy, reliability, and responsiveness of the system through rigorous testing and optimization, leveraging the capabilities of OpenCV and mediapipe libraries to achieve optimal performance; and evaluating the system's usability, effectiveness, and user experience through comprehensive user testing, feedback collection, and iterative refinement to address any identified issues or areas for improvement.

By addressing these objectives, the project aims to provide a transformative solution for delivering presentations in various settings, offering a more efficient, engaging, and immersive experience for both presenters and audiences. Moreover, the system's adaptability and versatility make it suitable for a wide range of applications beyond traditional presentation environments, including education, training, interactive exhibits, and remote collaboration.

1.4 Purpose:

At its core, this project seeks to instigate a transformation in the very essence of presentation delivery and audience engagement. Recognizing the limitations inherent in traditional presentation methods, which often result in passive audiences and diminished interest, our endeavor focuses on redefining these experiences through the integration of cutting-edge technologies.

The adoption of real-time hand tracking and gesture recognition technologies serves as the cornerstone of our approach. These innovations represent a departure from the static nature of traditional presentations, offering a dynamic means of interaction that captivates audiences and fosters engagement. By leveraging these technologies, we aim to create an immersive and interactive presentation environment where presenters and audiences can actively participate in the exchange of information.

Central to our mission is the empowerment of presenters to navigate their slides and engage with their audience in a natural and intuitive manner. Through the use of hand gestures, presenters can seamlessly guide their audience through the presentation, facilitating smoother transitions and enhancing comprehension. This interactive element not only sustains audience interest but also promotes better information retention, ultimately leading to more impactful communication outcomes.

Moreover, by prioritizing audience engagement and interaction, we aspire to create presentation experiences that leave a lasting impression. Beyond merely conveying information, our goal is to evoke emotions, provoke thoughts, and inspire action. Through the integration of real-time hand tracking and gesture recognition technologies, we endeavor to elevate presentations from passive spectacles to dynamic exchanges that leave a lasting impact on both presenters and audiences alike.

In essence, the purpose of this project extends beyond mere technological innovation; it embodies a vision for reimagining the very nature of presentations. By embracing new possibilities and pushing the boundaries of traditional approaches, we strive to create presentation experiences that are not only informative but also memorable, engaging, and transformative.

1.5 Scope:

The scope of this project is expansive, covering both the technical intricacies and the user-centric aspects essential for a successful presentation control system. From a technical standpoint, the system must possess the capability to accurately detect and interpret a diverse array of hand gestures in real-time. This necessitates the development and implementation of robust algorithms for hand tracking and gesture recognition, ensuring precise and responsive interactions between the presenter and the presentation software.

Additionally, seamless integration with existing presentation software, such as PowerPoint, is paramount. The system must seamlessly interface with these platforms, allowing presenters to control their presentations using intuitive hand gestures without any disruptions or compatibility issues.

Furthermore, the system's scope extends to accommodate various presentation styles and preferences. Presenters should have the flexibility to customize their interactions based on their individual needs and preferences, whether it involves controlling slide transitions, annotating content, or engaging with the audience in unique ways.

Overall, the scope of this project is ambitious, aiming to redefine the presentation experience through the innovative integration of hand tracking and gesture recognition technologies. By addressing both the technical complexities and the user-centric considerations, the project seeks to deliver a comprehensive and impactful solution that enhances the effectiveness and engagement of presentations across diverse settings and audiences.

Chapter 2
LITERATURE REVIEW

1. Rajeshwari Kumar Dewangan, H. Jabnoun, A. Jaiswal. Hand Gesture Recognition: Tracing Significant Points and Distance Calculation, 2015.

In their proposal for hand gesture recognition, Dewangan, Jabnoun, and Jaiswal present a sophisticated system leveraging advanced algorithms and techniques. The system's key features include the precise tracing of significant points within images and the calculation of distances between these points. By tracking the tip positions of the fingers, particularly the index finger and counters, the system enables intuitive gesture recognition.

One of the system's most notable advantages is its simplicity and efficiency in interacting with sound devices. Users can effortlessly control these devices without the need for complex manual configurations. Moreover, the system's design eliminates the necessity for specialized markers or gloves, making it accessible and cost-effective.

Operating in real-time on standard PCs equipped with low-cost cameras, this hand gesture recognition system demonstrates the potential for seamless integration into various applications, from interactive media to assistive technologies.

2. F. Zhigang. Computer gesture input and its application in human computer interaction, 1999.

The paper by F. Zhigang titled "Computer gesture input and its application in human-computer interaction" highlights the limitations of traditional input devices such as the mouse and keyboard in human-computer interaction (HCI). These devices create a barrier between the user and the computer, hindering the naturalness of the interface. To overcome this limitation, the paper proposes a robust marker-less hand gesture recognition system capable of efficiently tracking both static and dynamic hand gestures.

By employing a marker-less approach, the system eliminates the need for users to wear specialized equipment or markers on their hands, making the interaction process more natural and user-friendly.

One notable aspect of the system is its support for dynamic gestures, which allows users to perform actions such as navigating through slides in a presentation with fluid hand movements. This capability enhances the user experience by providing a more interactive and engaging way to interact with digital content.

The paper presents results demonstrating the effectiveness of the proposed system in achieving an intuitive HCI with minimal hardware requirements. By leveraging advances in computer vision and gesture

recognition technology, the system offers a compelling alternative to traditional input devices, enabling users to interact with computers in a more natural and intuitive manner. Overall, the paper contributes to the advancement of HCI by introducing a novel approach to gesture-based interaction that enhances usability and user experience.

3. Muhammad Idrees1 , Ashfaq Ahmad2 , Muhammad Arif Butt3 , and Hafiz Muhammad Danish. Controlling Power Point using Hand Gestures in Python, 2021.

The paper authored by Muhammad Idrees, Ashfaq Ahmad, Muhammad Arif Butt, and Hafiz Muhammad Danish titled "Controlling PowerPoint using Hand Gestures in Python" underscores the significance of presentations across various domains of life, from academia to business. While PowerPoint presentations are commonplace, they can sometimes lack liveliness due to the reliance on traditional input methods such as keyboards or dedicated gadgets. The authors aimed to address this issue by empowering users to control slideshows using hand gestures, thus enhancing the interactive and engaging nature of presentations.

The motivation behind this research stems from the increasing applications of gestures in human-computer interaction (HCI) in recent years. Gestures offer a more intuitive and natural way for users to interact with digital interfaces, moving away from the constraints imposed by conventional input devices. Leveraging machine learning techniques, the research endeavors to detect and interpret subtle differences in hand gestures, mapping them to fundamental PowerPoint slideshow control functions.

The implementation of this research is carried out using Python, a versatile and widely used programming language known for its simplicity and ease of integration with various technologies. By harnessing machine learning algorithms, the system is trained to recognize a diverse range of hand gestures, allowing users to perform operations such as advancing slides, navigating between slides, and controlling other aspects of the slideshow.

The use of machine learning adds a layer of intelligence to the system, enabling it to adapt and learn from user interactions over time, thereby improving the accuracy and reliability of gesture recognition. This dynamic approach ensures that the system can effectively distinguish between different gestures, even those with subtle variations, enhancing its usability and robustness.

Overall, the research contributes to the advancement of HCI by exploring novel ways to enhance the user experience in PowerPoint presentations through gesture-based interaction. By leveraging machine learning techniques and Python programming, the authors demonstrate the feasibility of controlling PowerPoint slideshows using hand gestures, opening up new possibilities for more immersive and engaging presentations across various domains.

4. K. Prasanna Kumari, Bandaram Bharath Goud, Kalvakuntla Sumana, Bathula Naresh, Bellamkonda Harish. Automated Gesture Controlled Presentation Using Machine Learning, 2022.

The paper authored by K. Prasanna Kumari, Bandaram Bharath Goud, Kalvakuntla Sumana, Bathula Naresh, and Bellamkonda Harish titled "Automated Gesture Controlled Presentation Using Machine Learning" addresses the growing significance of presentations across various fields of life, highlighting their importance for students, entrepreneurs, businessmen, and corporate workers alike. Despite the ubiquity of PowerPoint presentations, they can sometimes lack engagement and liveliness due to the reliance on conventional input methods such as keyboards or dedicated gadgets.

The authors' aim is to enhance the presentation experience by enabling users to control slideshows using hand gestures, thus eliminating the need for traditional input devices. This approach not only enhances the interactivity of presentations but also makes the presenter's task easier and more intuitive.

The motivation behind this research lies in the increasing utilization of gestures in human-computer interaction (HCI) in recent years. Gestures offer a more natural and intuitive way for users to interact with digital interfaces, enhancing user experience and engagement. Leveraging machine learning techniques, the research endeavors to detect and interpret subtle differences in hand gestures, allowing users to perform various operations on PowerPoint slideshows.

Python is chosen as the programming language for implementing this research due to its versatility and ease of integration with machine learning libraries. By training machine learning models on a dataset of hand gestures, the system is able to recognize and classify different gestures accurately, mapping them to fundamental PowerPoint slideshow control functions.

The use of machine learning adds intelligence to the system, enabling it to adapt and improve over time based on user interactions. This dynamic approach ensures that the system can effectively recognize a wide range of gestures, even those with subtle variations, thus enhancing its usability and reliability.

Overall, the research contributes to the advancement of HCI by exploring innovative ways to enhance the presentation experience through gesture-based interaction. By leveraging machine learning and Python programming, the authors demonstrate the feasibility of automating gesture-controlled presentations, thereby making presentations more engaging, intuitive, and effortless for presenters.

5. Smith K. Exploring the Use of Gesture-Controlled Presentation Systems in Education, 2019.

The groundbreaking case study conducted by Smith and collaborators (2019) and published in the esteemed Journal of Educational Technology, delves deep into the transformative potential of gesture-controlled presentation systems within educational settings. Through meticulous exploration and analysis,

the researchers shed light on the myriad benefits and challenges associated with integrating such innovative technology into pedagogical practices.

At the heart of the study lies a profound investigation into the impact of gesture-controlled presentation systems on student engagement and learning outcomes. Through a combination of quantitative metrics and qualitative observations, the researchers meticulously examine how these systems shape the educational landscape, offering tantalizing glimpses into a future where interactive technology revolutionizes traditional classroom dynamics.

Furthermore, the study meticulously navigates the complex terrain of challenges inherent in implementing gesture-controlled presentation systems in educational environments. From technical constraints to pedagogical considerations, the researchers offer a nuanced understanding of the obstacles that must be overcome to harness the full potential of this transformative technology.

6. Akshaya Ramachandran, Raksha Aruloli, Taran Akshay. Hand Gesture Recognition Based Presentation System, 2024.

A system that employs hand gestures to control the presentation is called hand gesture recognition for PPT presentations. The user's hand is detected by the system using a camera, which identifies the hand and its landmarks using a hand detection algorithm. The hand motions can be used by the system to control the display once it has recognized the hand and its landmarks. There are several methods in which the presentation can be managed by the system. To continue to the next slide, return to the previous slide, or pause the presentation, for instance, the user can make a gesture. Annotations can also be made on the slides by the user with motions. When compared to conventional presentation control approaches, the system offers several advantages. For instance, the technology allows the user to operate the presentation more naturally by eliminating the need for a keyboard or mouse. Because the user may manipulate the display using movements in a manner akin to controlling a live presentation, the system feels even more natural.

7. Shreya Sawant, Tanvi Sawant, Sayali Narale, Hemant Kasturiwale. Hand Gesture Controlled PowerPoint Presentation using OpenCV, 2023.

The literature review for "Hand Gesture Controlled PowerPoint Presentation using OpenCV" by Shreya Sawant, Tanvi Sawant, Sayali Narale, and Hemant Kasturiwale (2023) explores the interdisciplinary field of human-computer interaction (HCI), computer vision, and presentation techniques. It begins by examining the historical evolution and significance of gesture control in HCI, tracing its applications from media players and gaming consoles to robotics. Previous studies on gesture detection systems are reviewed, highlighting the challenges associated with traditional methods that rely on specialized peripherals like

gloves or markers. The review then delves into recent advancements in artificial intelligence (AI) and its application to gesture recognition, emphasizing the potential of AI-driven approaches to overcome limitations of traditional methods. OpenCV, a widely used library for computer vision tasks, emerges as a key component in gesture recognition systems, offering capabilities for image processing and feature extraction. The review also explores the feasibility and advantages of using webcams for gesture detection, emphasizing factors such as accuracy and user accessibility. Finally, the literature review examines existing research on gesture-based interaction in PowerPoint presentations, providing insights into its effectiveness in enhancing audience engagement and presentation quality.

Through this comprehensive review, the authors lay the groundwork for their proposed solution, which leverages AI-based hand gesture detection using OpenCV to enhance the control and engagement of PowerPoint presentations.

8. Hope Orovwode, John Amanesi Abubakar, Onuora Chidera Gaius, Ademola Abdullkareem. The Use of Hand Gestures as a Tool for Presentation, 2023.

This paper explores the integration of hand gestures as a means of controlling presentations, offering a dynamic alternative to traditional input devices such as mice or keyboards. While conventional methods tether presenters to fixed positions and rely on device proximity, hand gesture controls promise a more fluid and engaging presentation style. The study utilizes the HaGRID dataset supplemented by custom-recorded data, with a training set comprising 80% of the data and 10% each for validation and testing. Data preprocessing is conducted, followed by the implementation of a linear classifier with four dense layers and a SoftMax activation layer. Additionally, a motion classifier incorporating an LSTM (Long Short-Term Memory) architecture tailored for long-distance body pose estimation is utilized. The resulting application, a standalone desktop tool, boasts high accuracy in gesture classification across training, validation, and testing sets. Various features, including PowerPoint selection, distance mode, gesture toggling, and appearance mode, enhance user interaction and presentation management.

9. Charan Meena, Harshit Soni, Priyanshu soni, Lokesh Lowanshi, Prof. Satish Chadokar. Smart Hand Gestured Controlled Presentation System,2015.

This paper introduces a novel approach to presentation control using hand gesture recognition technology, highlighting the significance of this innovation within the broader context of human-computer interaction. Hand gesture recognition has emerged as a prominent research area in computer vision, with a

focus on developing algorithms capable of accurately detecting and classifying gestures in real-time. Notable

frameworks like MediaPipe have advanced this field by integrating machine learning techniques to enhance gesture recognition capabilities. Leveraging such frameworks, researchers have explored the potential of hand gestures as intuitive input methods for controlling various digital interfaces, including presentation slides.

The application of hand gesture recognition in presentation control systems represents a compelling avenue for enhancing user experience and engagement during presentations. By allowing users to navigate slides and manipulate content through natural hand movements, these systems offer a modern and interactive alternative to traditional input devices. Previous studies have demonstrated the feasibility and effectiveness of using hand gestures for presentation control, emphasizing the importance of intuitive interfaces in facilitating seamless interaction with digital content.

In conclusion, this paper contributes to the existing literature by proposing a hand gesture presentation control system and contextualizing its significance within the broader landscape of human-computer interaction. By reviewing previous studies and developments in hand gesture recognition technology, this paper underscores the potential of gesture-based interfaces to revolutionize interactive systems, particularly in the domain of presentation control.

10. Chetana Patil, Amrita Sonare, Aliasgar Husain, Aniket Jha, Ajay Phirke. Controlled Hand Gestures Using Python and OpenCV, 2023.

This paper delves into the significance of gesture recognition within contemporary trends, emphasizing its role in facilitating communication between humans and machines. Gesture-based interactions serve as a form of non-verbal communication, enabling intuitive exchanges between users and computers. In the realm of artificial intelligence, the integration of hand gesture detection enhances functionality and fosters personalized commerce experiences. Utilizing counterplotted action dyads and Python libraries such as OpenCV and cvzone, researchers have devised systems adept at executing specific tasks through gesture-driven commands. These libraries offer critical functionalities for tasks such as image capture, pre-processing, and pattern recognition, thereby empowering the development of robust gesture recognition systems.

By starting with the paper described in the abstract, this literature matter provides a foundation for exploring the evolving landscape of gesture recognition technology and its applications in human-computer interaction and artificial intelligence.

Chapter 3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Traditional presentation control systems rely on manual input devices, lacking intuitiveness and interactivity. Remote control software exists but requires additional hardware or software installations, adding complexity. Standalone interactive whiteboards lack integration with popular presentation software like PowerPoint. Existing systems offer varying interactivity but suffer from limitations like complexity and lack of integration. This project proposes a solution for controlling presentations using hand gestures, aiming for intuitiveness, accessibility, and integration.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM

- Limited Interaction
- Cost of Additional Hardware
- Lack of Customization Options
- Complex Setup

3.2 PROPOSED SYSTEM

Our proposed system represents a paradigm shift in presentation control, utilizing state-of-the-art hand tracking and gesture recognition technology to provide a seamless and intuitive user experience. By harnessing the power of natural hand gestures, presenters can effortlessly navigate slides, annotate content, and engage with their audience in real-time. We have used OpenCV and Mediapipe in our project. OpenCV has more than 2,500 algorithms in which CNN works efficiently to detect hands and Mediapipe to Recognize Gestures of detected hand. Below, we detail the key components and functionalities of the proposed system:

1. Integration with Presentation Software: Our system seamlessly integrates with popular presentation software platforms, such as Microsoft PowerPoint. Presenters can control the presentation flow directly from the integrated interface, eliminating the need for external input devices.
2. Live Camera Feed Overlay: The system incorporates a live camera feed overlay onto the presentation interface, providing visual feedback of the presenter's gestures in real-time. This feature enhances presenter awareness and facilitates smoother interaction with the presentation content.
3. Real-Time Annotation and Interaction: Presenters can annotate slides in real-time using intuitive hand gestures, enhancing the clarity and impact of their presentations. Gestures for drawing and erasing enable dynamic content customization and emphasis during the presentation.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM

- Intuitive interaction: The proposed system enables users to control presentations using natural hand gestures, making the interaction more intuitive and engaging.
- Enhanced Flexibility: Unlike traditional input devices such as keyboards or mice, hand gestures allow for more dynamic and expressive control over presentation content. Users can easily navigate slides, annotate content, and interact with audience members in real-time.
- Customizable Gestures: Unlike existing systems with predefined gesture sets, the proposed system allows users to define and customize their own gestures according to their preferences and presentation style.
- Effortless Setup: The proposed system offers a straightforward setup process, requiring minimal hardware and software configuration.

3.2.2 WORKING OF CNN:

In our project, we integrated an algorithm for hand tracking and gesture recognition to facilitate seamless interaction with presentations. Our chosen approach involved leveraging convolutional neural networks (CNNs), a powerful class of deep learning models known for their effectiveness in image recognition tasks.

Convolutional neural networks (CNNs) are particularly well-suited for tasks like hand tracking and gesture recognition due to their ability to learn hierarchical representations of visual data. These networks excel at automatically extracting meaningful features from images, making them ideal for detecting and tracking the movements of the presenter's hands in real-time.

The process begins with the hand tracking component of the algorithm. Here, CNN is trained on a dataset of hand images to learn to accurately locate and track the position of the presenter's hands within the frame. This involves identifying key landmarks or key points on the hands, such as fingertips and palm centers, which serve as reference points for gesture recognition.

Once the hands are successfully tracked, the algorithm proceeds to the gesture recognition phase. Again, CNNs play a central role in this process by learning to classify different hand gestures based on their visual characteristics. The network is trained on a dataset containing examples of various gestures, allowing it to learn the unique patterns and features associated with each gesture.

During inference, the trained CNN analyzes the current hand positions and movements captured by the tracking component, predicting the corresponding gesture being performed by the presenter. This information is then used to trigger specific actions within the presentation software, such as advancing slides, highlighting content, or interacting with multimedia elements.

By utilizing CNNs for both hand tracking and gesture recognition, our project benefits from their ability to learn complex patterns and relationships in visual data, leading to more accurate and robust performance. This approach enables us to accurately detect and track the movements of the presenter's hands in real-time, allowing for intuitive and seamless interaction with the presentation content.

Overall, the integration of convolutional neural networks enhances the effectiveness and reliability of our hand tracking and gesture recognition system, providing users with a more natural and immersive presentation experience.

3.2.3 Convolutional Neural Network(CNN) Layers:

1. Input Image: The input to the CNN is a frame captured from the video feed, containing the presenter's hand

2. Preprocessing: steps may include resizing the input image to a fixed size, normalization (scaling pixel values to a certain range).

3. Convolutional Layers: The preprocessed image passes through a series of convolutional layers. Each convolutional layer applies a set of filters (kernels) to extract features from the input image. ReLU (Rectified Linear Unit) activation function is typically applied to introduce non-linearity

4. Pooling Layers: After each convolutional layer, max-pooling or average-pooling is applied to down sample the feature maps, reducing computational complexity and spatial dimensions. Pooling layers help in retaining the most salient features while reducing spatial resolution.

5. Flattening: The output feature maps from the last convolutional layer are flattened into a one-dimensional vector to prepare for the fully connected layers.

6. Fully Connected Layers: The flattened features are passed through one or more fully connected layers (also known as dense layers). These layers perform high-level feature extraction and classification.

7. Output Layer: The output layer consists of neurons corresponding to different classes or categories (e.g., hand position). SoftMax activation function is often used to convert raw scores into probabilities, representing the likelihood of each class.

8. Output: The final output of the CNN is the predicted hand position, which can be used for subsequent tasks such as gesture recognition or interaction with presentation software.

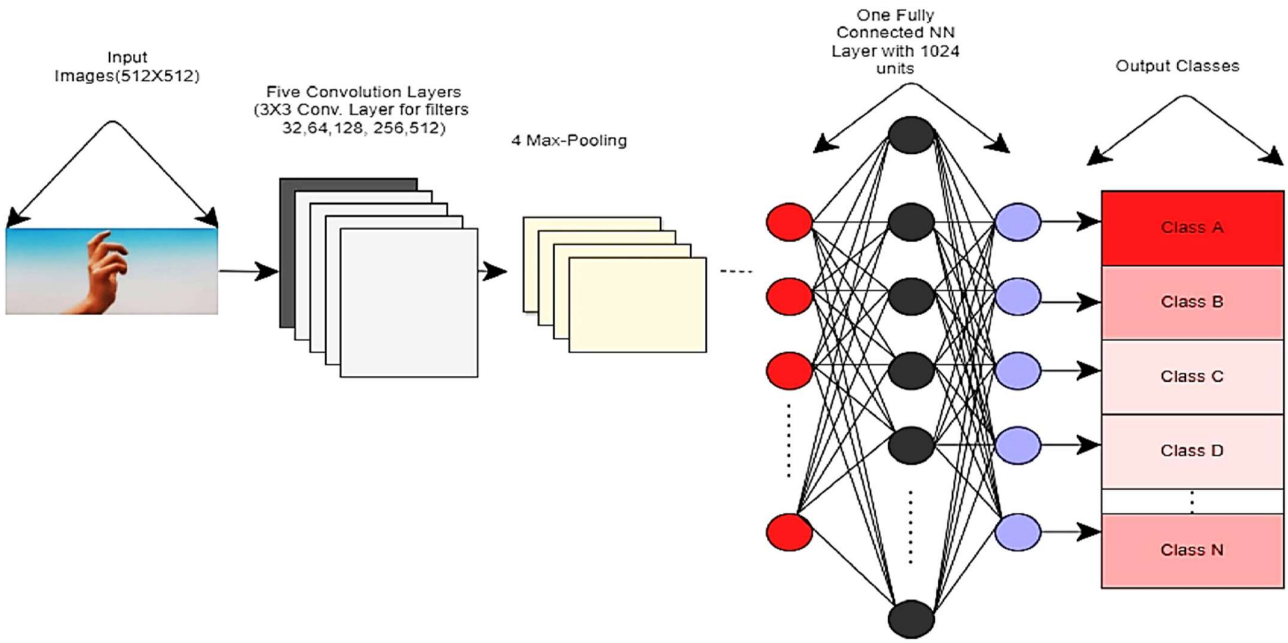

FIG:3.1 Architecture of CNN

3.2.4 METHODOLOGY

The proposed system is implemented using Python and makes use of computer vision techniques to detect and classify hand gestures. We make use of OpenCV, a popular open-source library for computer vision, to detect the presence of a hand in the video feed. Once a hand is detected, we use a Hand Module to detect and classify the hand gestures. Data is trained using a vector set of hand gestures, which includes examples of various gestures such as changing slides, next slide, previous slide, pointing, and highlight points.

Hand gesture recognition is done using Python programming language and OpenCV as library. Python programming language produces simple and easy system code to understand. Also, Python package used here is NumPy. The image that is captured using web camera will be processed in a region called as Region of Interest (ROI) where act as a region of wanted area while ignoring the outside region, called background.

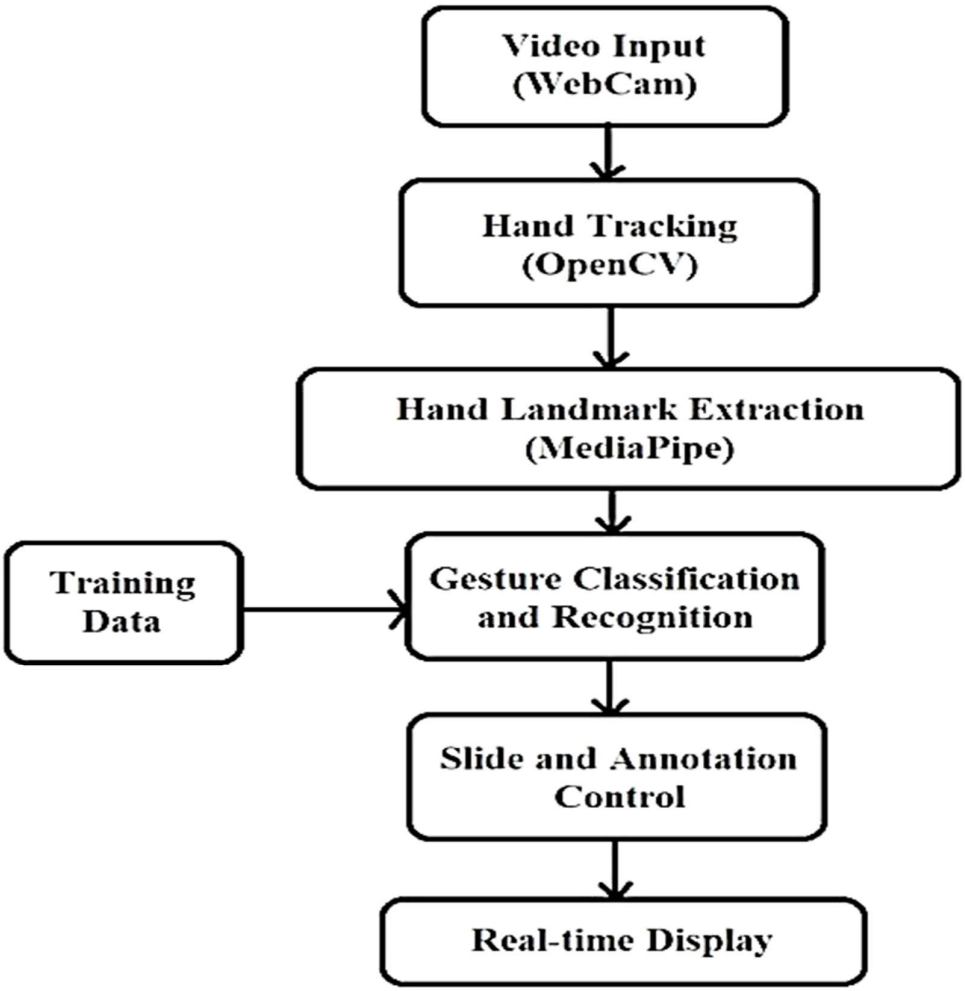

FIG:3.2 Representation of System Architecture

3.2.5 Working of the Project :

The OpenCV and NumPy packages were used to develop the project's code in the Python programming language. The libraries used in this work are initially imported before being used for additional input and output processing. The system captures video input from a webcam, and each frame is processed to detect and track the presenter's hand. The Hand tracking Module, based on the MediaPipe Hands solution, is employed to identify hand landmarks and extract relevant information for gesture recognition. By analyzing the finger positions and movements, the system determines the active gesture and performs the associated action.

The system follows a systematic flow to enable seamless interaction with presentation slides:

i. Hand Detection:

The system utilizes OpenCV to capture live video input from a webcam. By analyzing the video frames, the system detects and identifies the presenter's hand within the captured images.

ii. Hand Tracking:

Once the hand is detected, MediaPipe is employed to track the hand movements and extract hand landmarks. These landmarks represent specific points on the hand, such as fingertips and joints, which will be used for gesture recognition.

iii. Gesture Recognition:

By analyzing the positions and movements of the hand landmarks, the system accurately recognizes predefined gestures. Each gesture corresponds to a specific action within the presentation software.

The gestures supported are as follows:

Gesture 1: Thumb Finger - Move to Previous Slide Gesture

Gesture 2: Little Finger - Move to Next Slide Gesture

Gesture 3: Index Finger and Middle Finger Together - Drawing on the Slide Gesture

Gesture 4: Index Finger - Holding the Pointer

Gesture 5: Middle Three Fingers - Undo the Previous Annotation

Gesture 5: All Five Fingers Open – Clear All Annotations

A. Slide Navigation: The system recognizes the Thumb Finger gesture, allowing presenters to move to the previous slide. Similarly, the Little Finger gesture enables presenters to progress to the next slide, providing seamless slide navigation.

B. Pointer Control: Presenters can hold a virtual pointer by bringing the Index Finger. This gesture enables them to highlight specific areas of the slide, directing the audience's attention and emphasizing key points. The dynamic pointer control adds an interactive element to the presentation.

C. Drawing and Annotations: The system enables presenters to draw on the slide using the Index finger and Middle finger gesture. By moving their finger across the screen, presenters can create real-time annotations, underline important details, or emphasize specific elements. This feature allows for on the-fly visual enhancements and effective communication of the content.

D. Erasing: To remove or revise previous annotations, presenters can utilize the Middle Three Fingers gesture. This gesture activates the erasing function, allowing presenters to effortlessly erase specific annotations or clear the entire slide, ensuring a clean and polished presentation.

E. Clear: To clear the annotations , presenters can utilize the all five fingers up gesture. This gesture enables clear all the annotations on the slides.

F. Interaction during Presentation: The recognized gestures and corresponding actions are then communicated to the presentation software. The system integrates with popular presentation tools, enabling the seamless execution of the desired actions, such as moving between slides, highlighting content, creating annotations, and erasing or undoing annotations.

3.2.6 Tools and Technologies :

1. **OpenCV:**

OpenCV is a robust open-source computer vision and machine learning library that provides a plethora of tools and algorithms for various vision-related tasks. It offers functionalities ranging from image processing and object detection to video analysis and camera calibration. OpenCV is widely used for real-time applications due to its efficiency and extensive documentation. With pre-trained models and algorithms, developers can quickly implement features like object detection, tracking, and recognition. Its versatility makes it suitable for a wide range of applications, from robotics and augmented reality to healthcare and automotive systems.

2. **Mediapipe :**

Mediapipe developed by Google, is a modular framework designed for building machine learning pipelines for media processing tasks. It offers pre-trained models and a streamlined development process, making it easier for developers to create real-time applications for tasks like hand tracking, pose estimation, and face detection. Mediapipe's integration with TensorFlow allows for the seamless incorporation of custom machine learning models, expanding its capabilities beyond the provided modules. With its focus on real-time performance and cross-platform support, Mediapipe is well-suited for applications in fields such as augmented reality, human-computer interaction, and gesture recognition. MediaPipe Hands is a high-fidelity hand andfinger tracking solution. It employs machine learning (ML) to infer 21 3D landmarks of a hand from just a single frame.

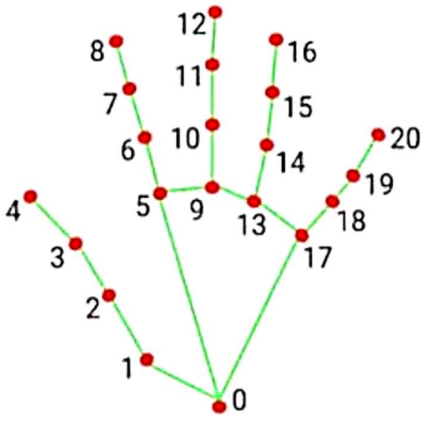

FIG:3.3 Co-ordinates of hand dots.

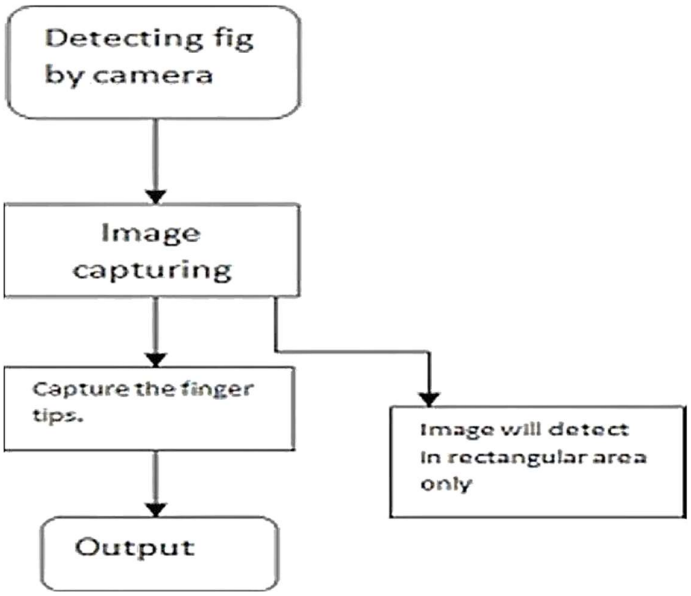

FIG:3.4 Media-Pipe hand tip recognition layout.

3.2.7 Libraries :

1. cv2 (OpenCV):

OpenCV library in python is a computer vision library that is widely used for image analysis, image processing, detection, recognition, etc. OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products

Installation command: `pip install OpenCV-python`

2. numpy:

numpy is a fundamental package for scientific computing with Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays efficiently.

Installation command: `pip install numpy`

3. os:

os module provides a way to interact with the operating system. It allows you to perform tasks such as file and directory manipulation, running system commands, and accessing environment variables.

4. win32:

win32 module is specific to Windows operating systems and provides access to various Win32 API functions. It is commonly used for tasks such as interacting with the Windows registry, accessing system information, and controlling GUI applications.

Installation command: `pip install pywin32`

5. mediapipe:

Mediapipe is a cross-platform library developed by Google that provides amazing ready-to use ML solutions for computer vision tasks. MediaPipe Hands is a high-fidelity hand andfinger tracking solution. It employs machine learning (ML) to infer 21 3D landmarks of a hand from just a single frame.

Installation command: `pip install mediapipe`

6. math:

math module provides mathematical functions and constants. It includes functions for basic arithmetic operations, trigonometry, logarithms, exponentiation, and more.

3.2.8 GESTURES PERFORMED:

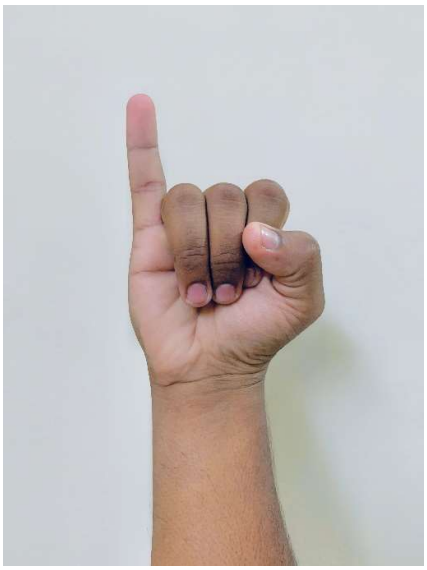

FIG:3.5 to next slide

FIG:3.6 to previous slide

FIG:3.7 to obtain the Pointer

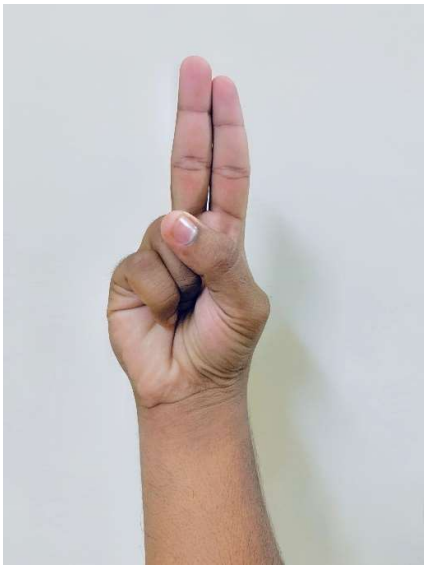

FIG:3.8 to Draw on the slides

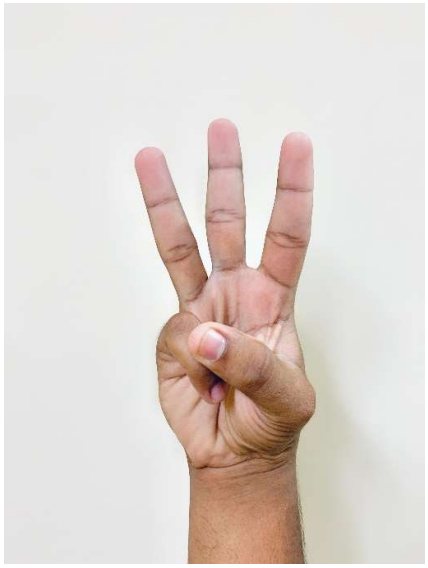

FIG:3.9 undo

FIG:3.10 clear screen

3.2.9 Gesture Inputs:

Gesture 1: to next slide.

[0,0,0,0,1]

Only the little finger is open in this gesture, while the other four fingers are all closed.

Gesture 2: to Previous slide.

[1,0,0,0,0]

Only the thumb is open in this gesture, while the other fingers are all closed.

Gesture 3: to obtain the pointer.

[0,1,0,0,0]

Only the index finger is open in this gesture, while the other fingers are all closed.

Gesture 4: to draw on the slides.

[0,1,1,0,0]

In this gesture the index finger and middle finger are open, the other fingers are all closed.

Gesture 5: Undo.

[0,1,1,1,0]

In this gesture the index finger, middle finger, and ring finger are open, all the other fingers are closed. The most recent written part is to be erased with this gesture.

Gesture 6: Clear Screen.

[1,1,1,1,1]

In this gesture, all the fingers are open. It clears all the annotations drawn on the Screen.

3.3 FEASIBILITY STUDY

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system are feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Economical Feasibility
- Operational Feasibility
- Technical Feasibility

3.3.1 ECONOMIC FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems.

Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available, there is nominal expenditure and economical feasibility for certain.

3.3.2 OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization's operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following:

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So, there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

3.3.3 TECHNICAL FEASIBILITY

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipment's have the technical capacity to hold the data required to use the new system?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and those are available as free as open source. The work for the project is done with the current equipment and existing software technology.

Chapter 4
SYSTEM SPECIFICATIONS

4.1 HARDWARE REQUIREMENTS

- Webcam (For real-time hand Detection)
- System Type : 64-bit operating system
- Processor : Intel(R)Pentium(R) CPU N3710 @1.60GHz
- Installed Ram : 4 GB

4.2 SOFTWARE REQUIREMENTS

- Operating system : windows 7 and above.
- Python version : 3.6 and above.
- Presentation Software : Microsoft Power point.
- Python compiler.

Chapter 5
SYSTEM DESIGN

5.1 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. UML uses mostly graphical notations to express the design of software projects.

GOALS

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

5.1.1 USE CASE DIAGRAM

A use case diagram in the Unified Modelling Language (UML) is a type of behavioural diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

FIG:5.1 Representation of Use Case Diagram

5.1.2 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

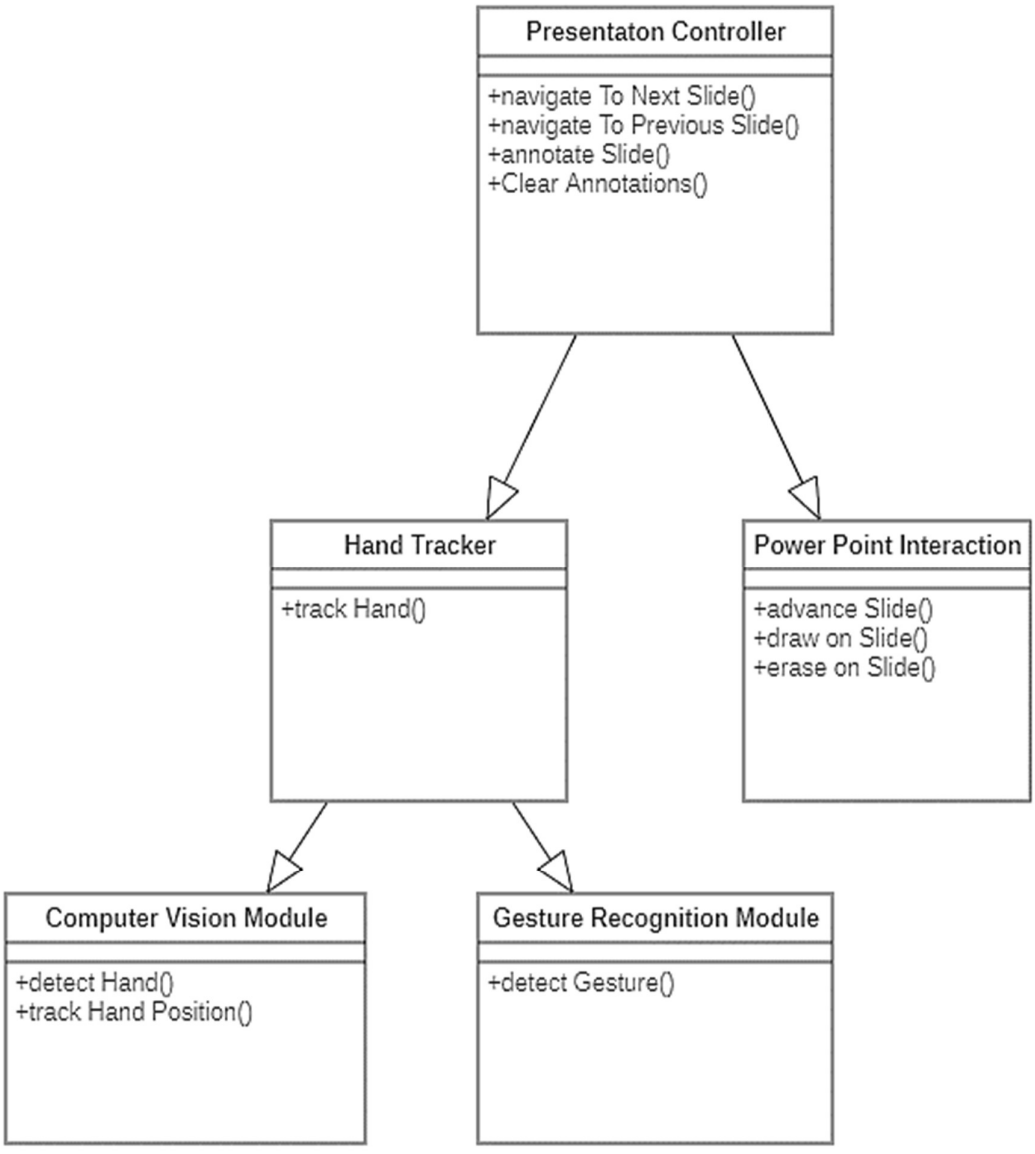

FIG:5.2 Representation of Class Diagram

5.1.3 SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

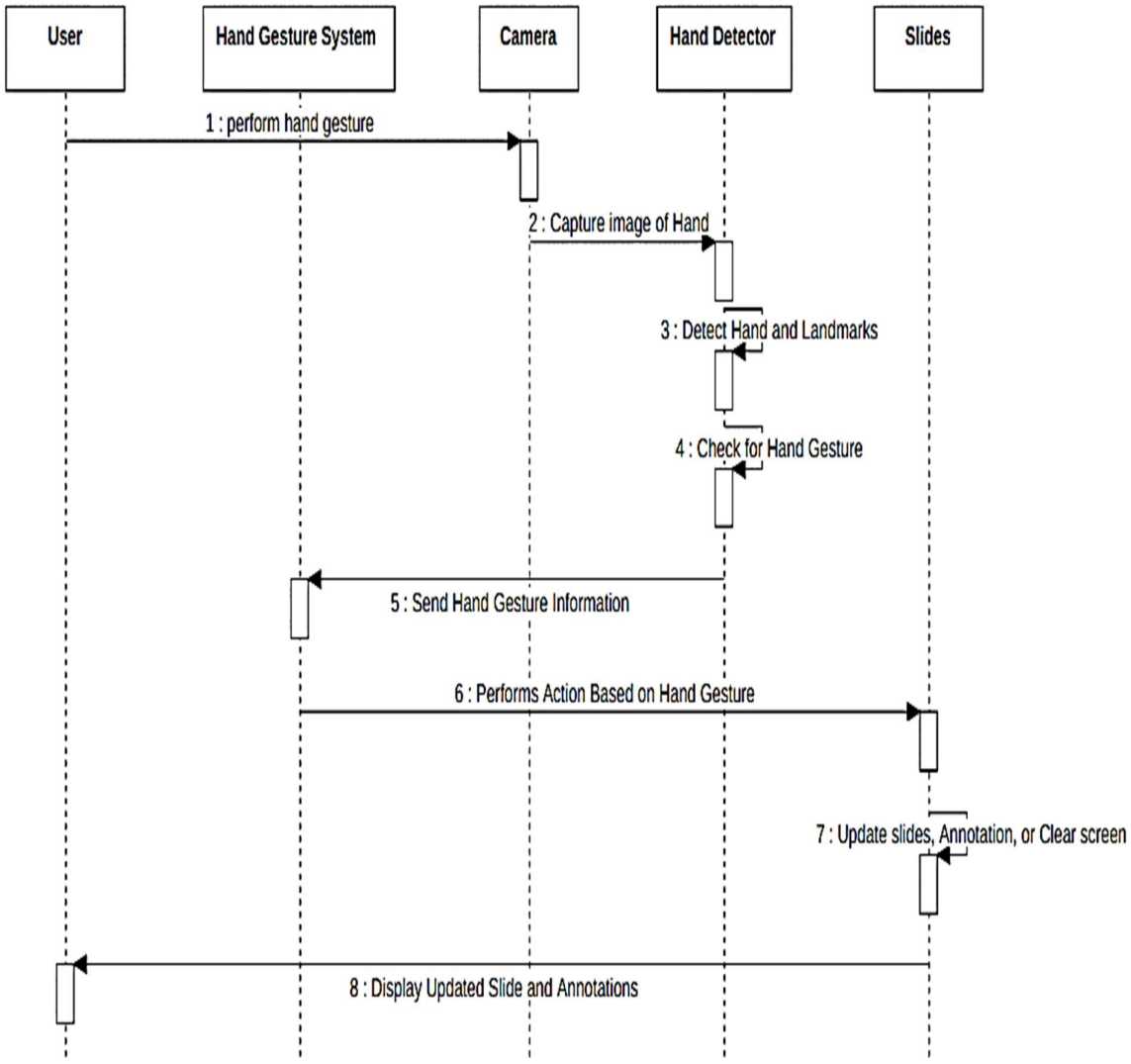

FIG:5.3 Representation of Sequence Diagram

5.1.4 COLLABORATION DIAGRAM

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of several features.

Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

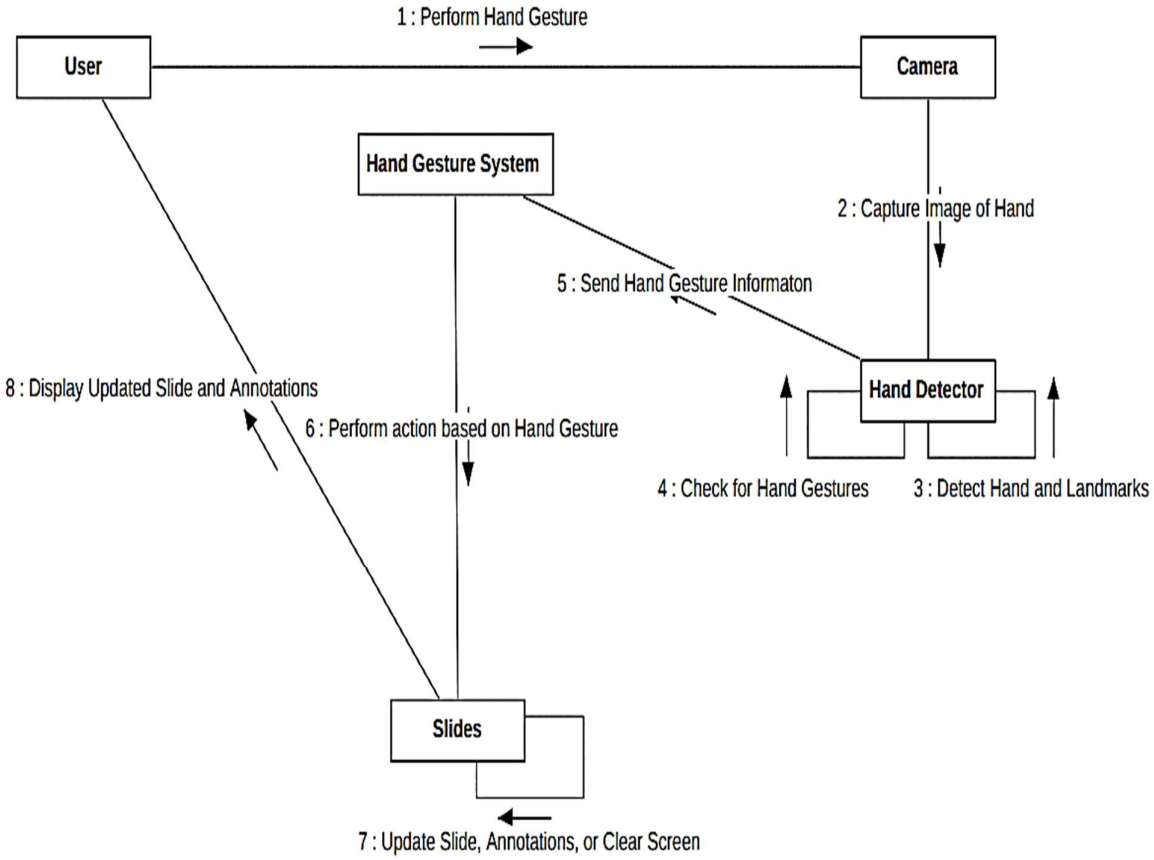

FIG:5.4 Representation of Collaboration Diagram

5.1.5 COMPONENT DIAGRAM

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node.

It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, and it necessitates an interface to execute a function. It is like a black box whose behaviour is explained by the provided and required interfaces.

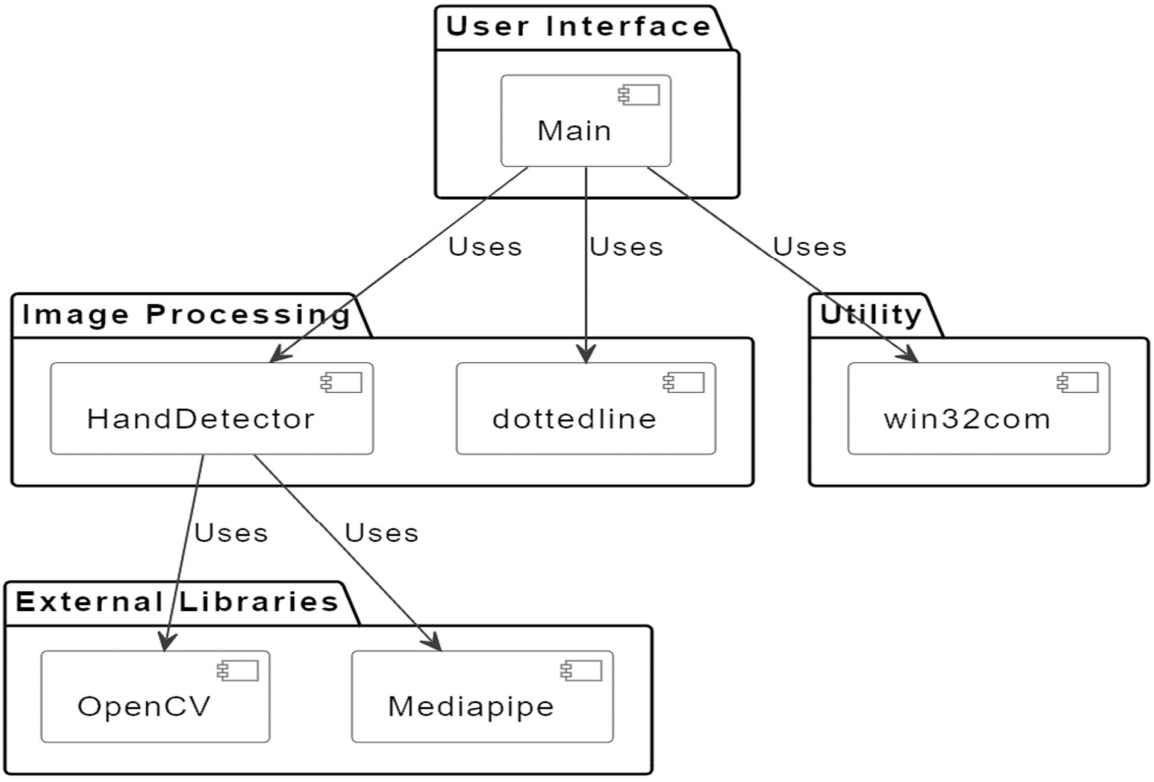

FIG:5.5 Representation of Component Diagram

CHAPTER 6
IMPLEMENTATION AND TESTING

6.1 IMPLEMENTATION

Implementation is the process of translating a concept, plan, or design into tangible action or realization. It involves executing the intended strategy or solution through practical steps, such as coding software, assembling hardware, or deploying systems, to achieve the desired outcome. In various fields, including technology, business, and education, implementation is fundamental to turning ideas into practical solutions or products. It often entails careful planning, resource allocation, and execution to ensure that the final result aligns with the initial goals and requirements.

6.1.1 PYTHON INSTALLATION

We have to install the current latest version of python for the better and quick performance and best outcome.

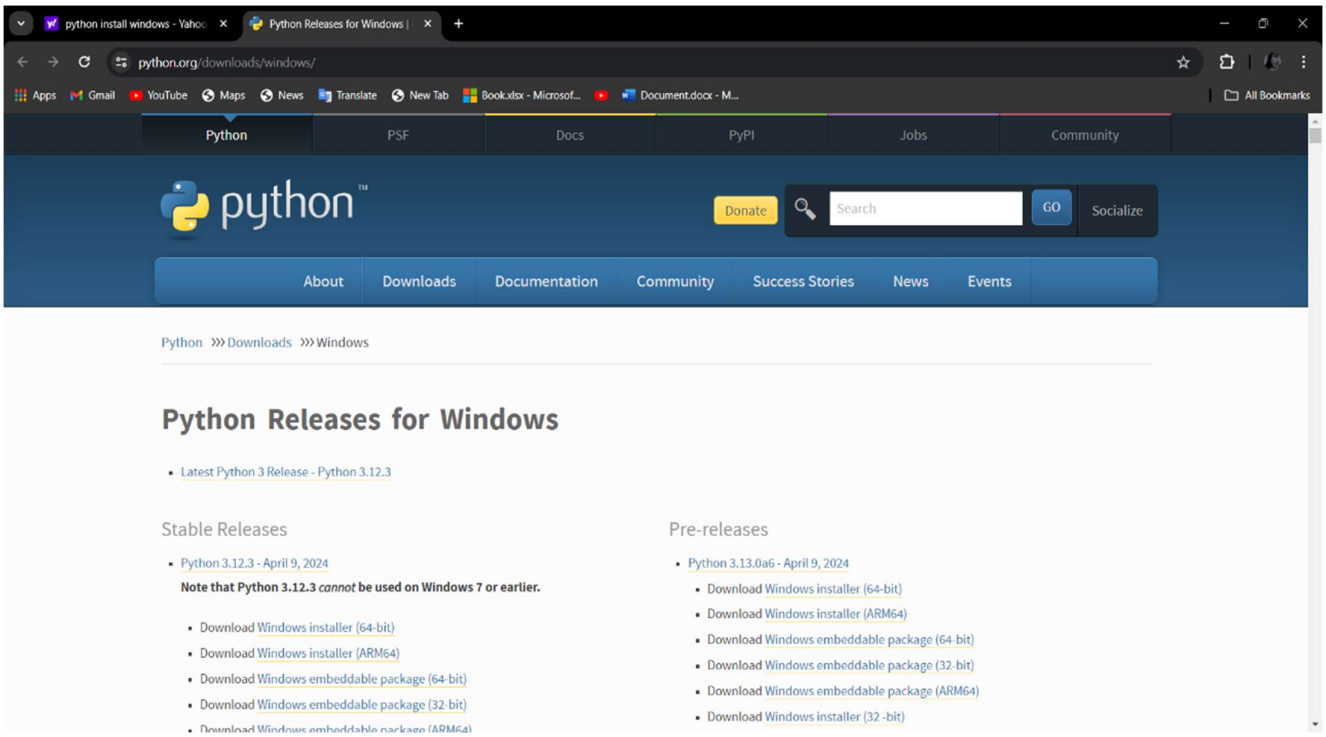

FIG: 6.1 Python download page

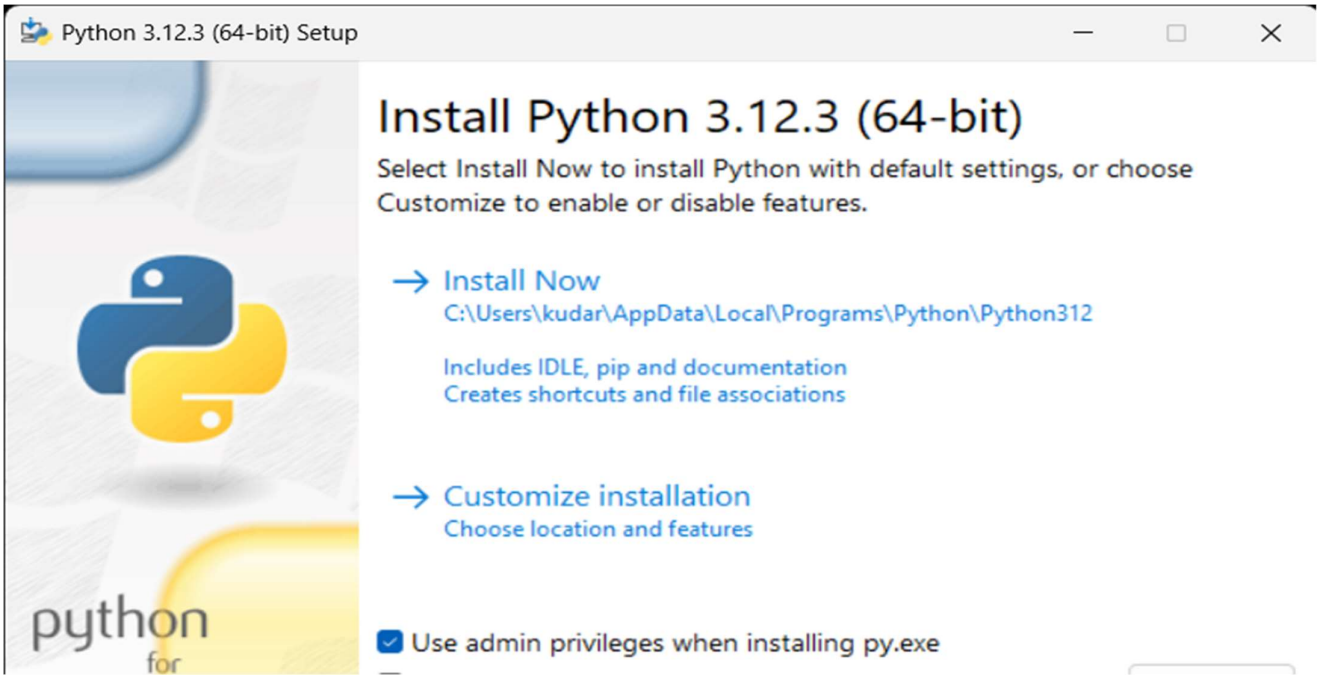

FIG:6.2 Python install process

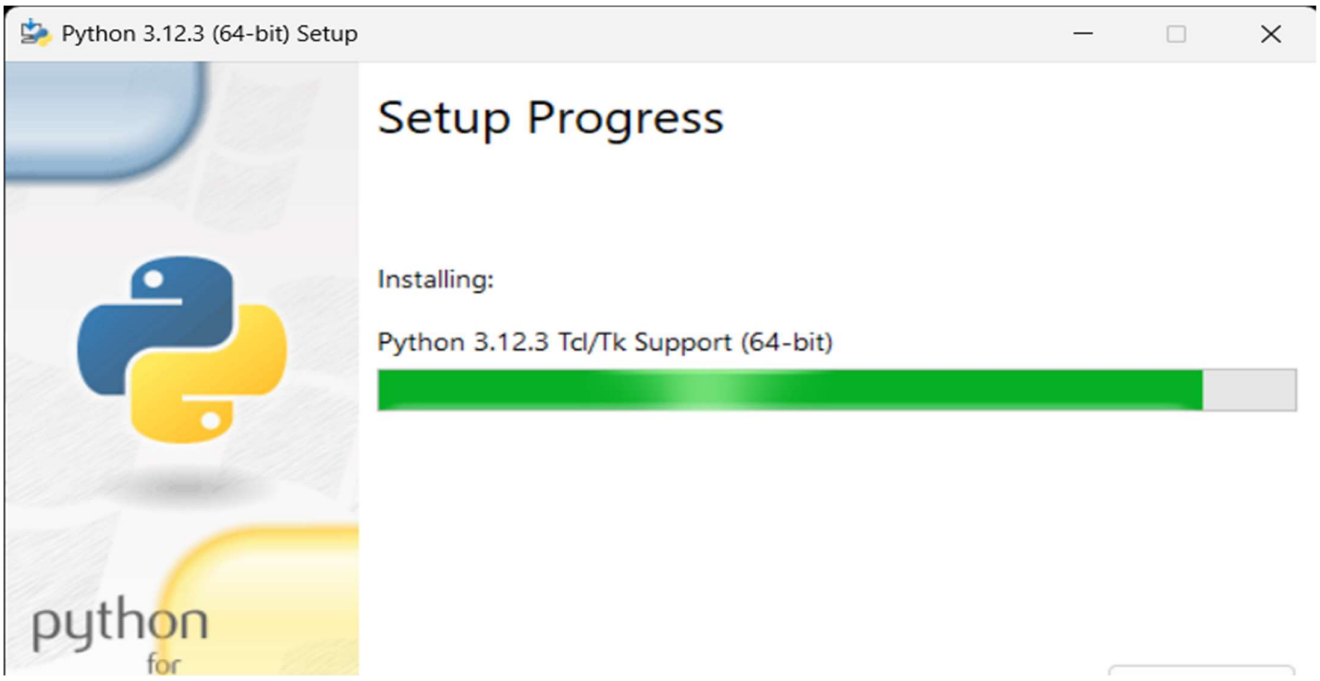

FIG: 6.3 Python setup progress in system

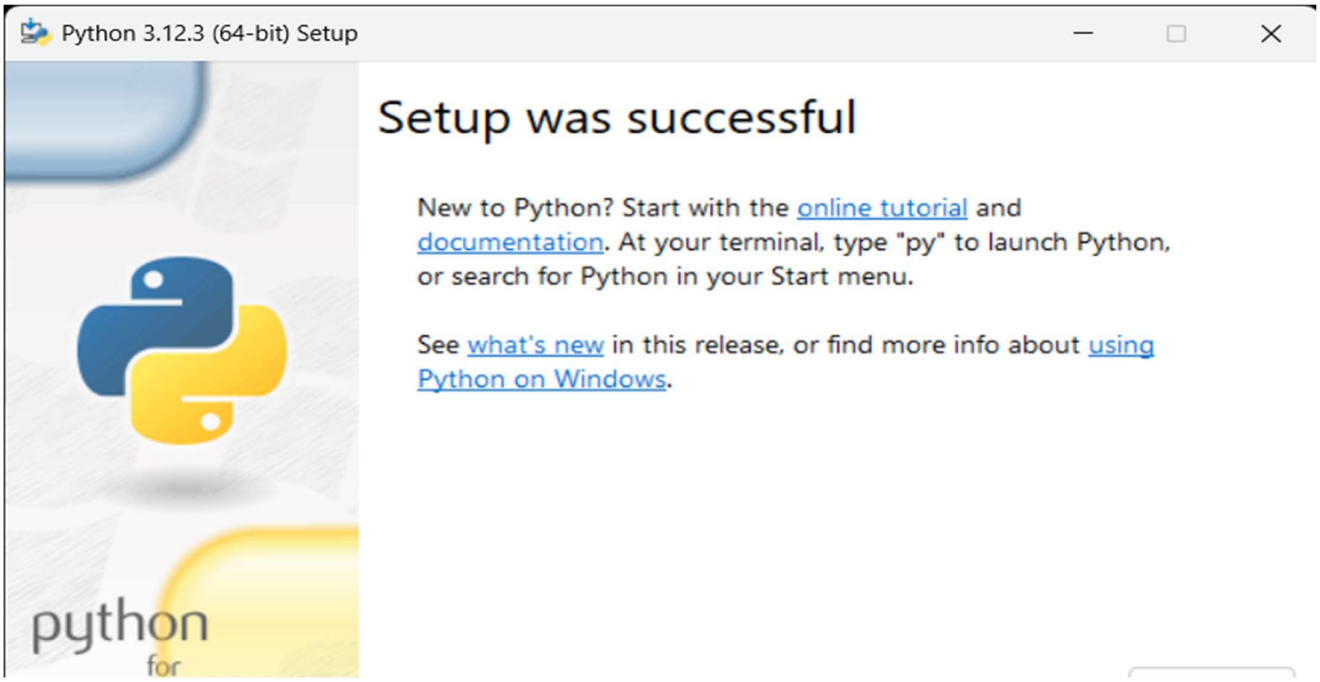

FIG:6.4 Setup was successful in system

6.1.2 Visual Studio Code Installation

Step 1: Visit the Official Website of the Visual Studio Code using any web browser like Google Chrome, Microsoft Edge, etc.

Step 2: Press the "**Download for Windows**" button on the website to start the download of the Visual Studio Code Application.

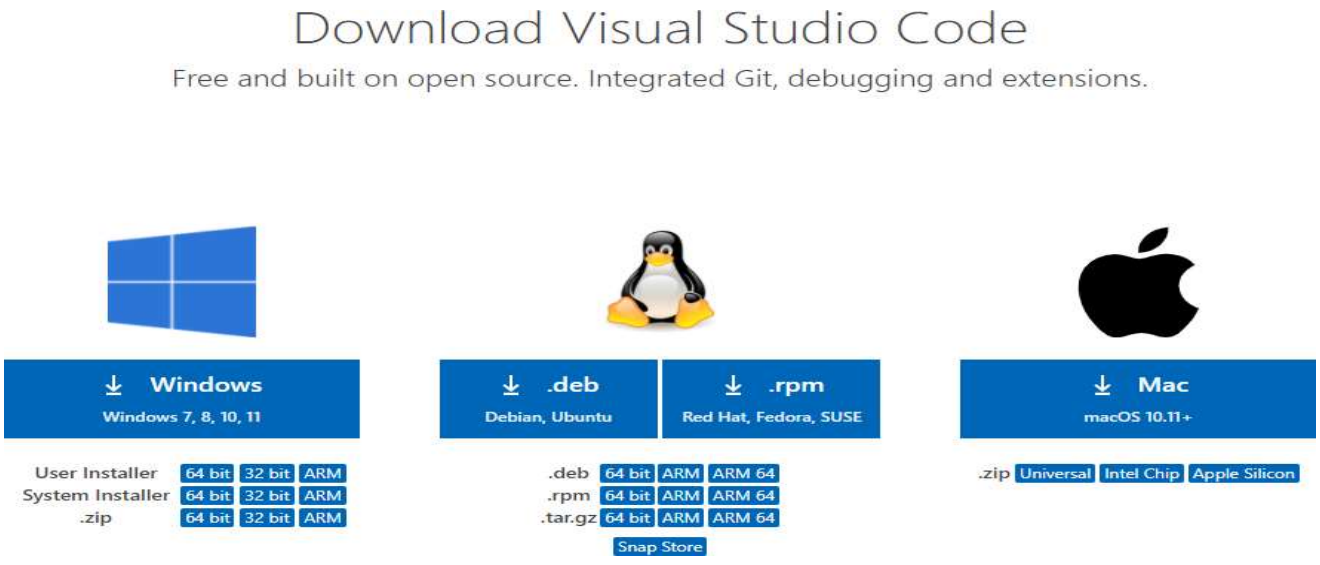

FIG:6.5 Visual Studio Code website

Step 3: When the download finishes, then the Visual Studio Code Icon appears in the downloads folder.

Step 4: Click on the Installer icon to start the installation process of the Visual Studio Code.

Step 5: After the Installer opens, it will ask you to accept the terms and conditions of the Visual Studio Code. Click on I accept the agreement and then click the Next button.

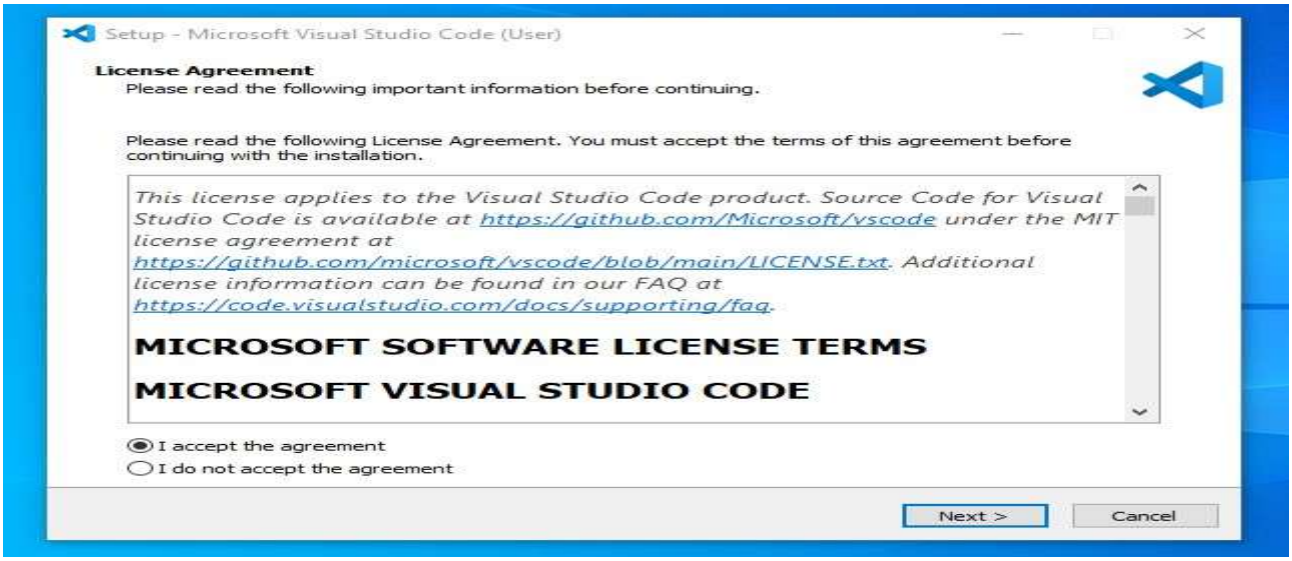

FIG:6.6 License agreement

Step 6: Choose the location data for running the Visual Studio Code. It will then ask you to browse the location. Then click on the Next button.

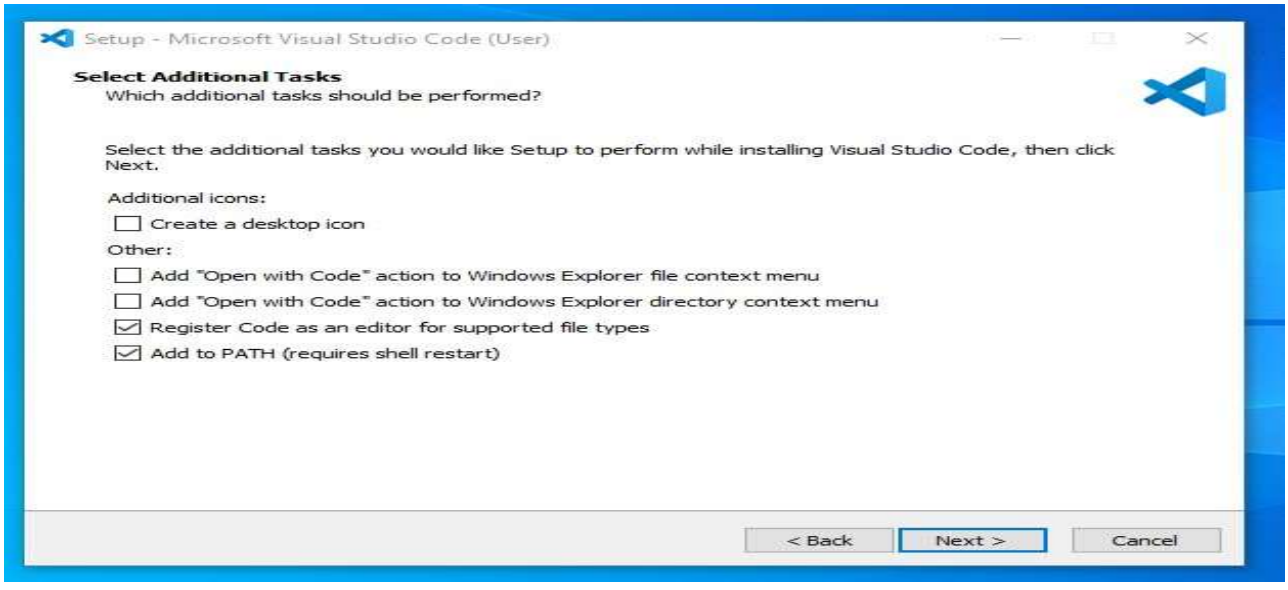

FIG:6.7 The Location Data For Running The Visual Studio Code

Step 7: Then it will ask to begin the installation setup. Click on the **Install** button.

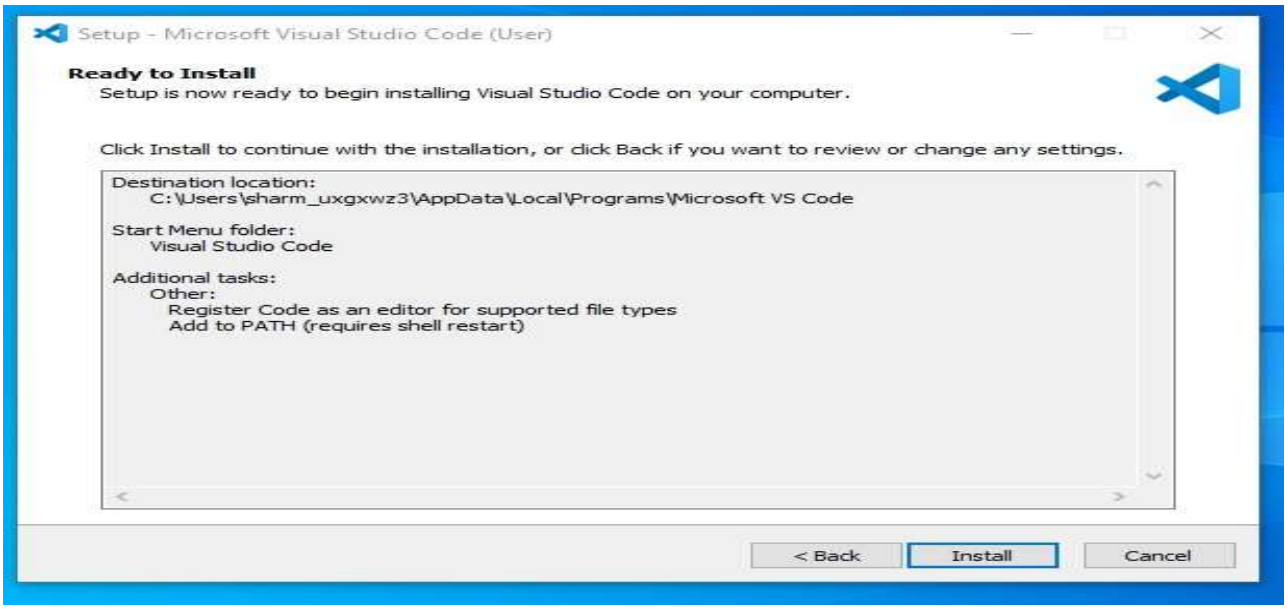

FIG:6.8 Installation Setup of Visual Studio Code

Step 8: After clicking on Install, it will take about 1 minute to install the Visual Studio Code on your device.

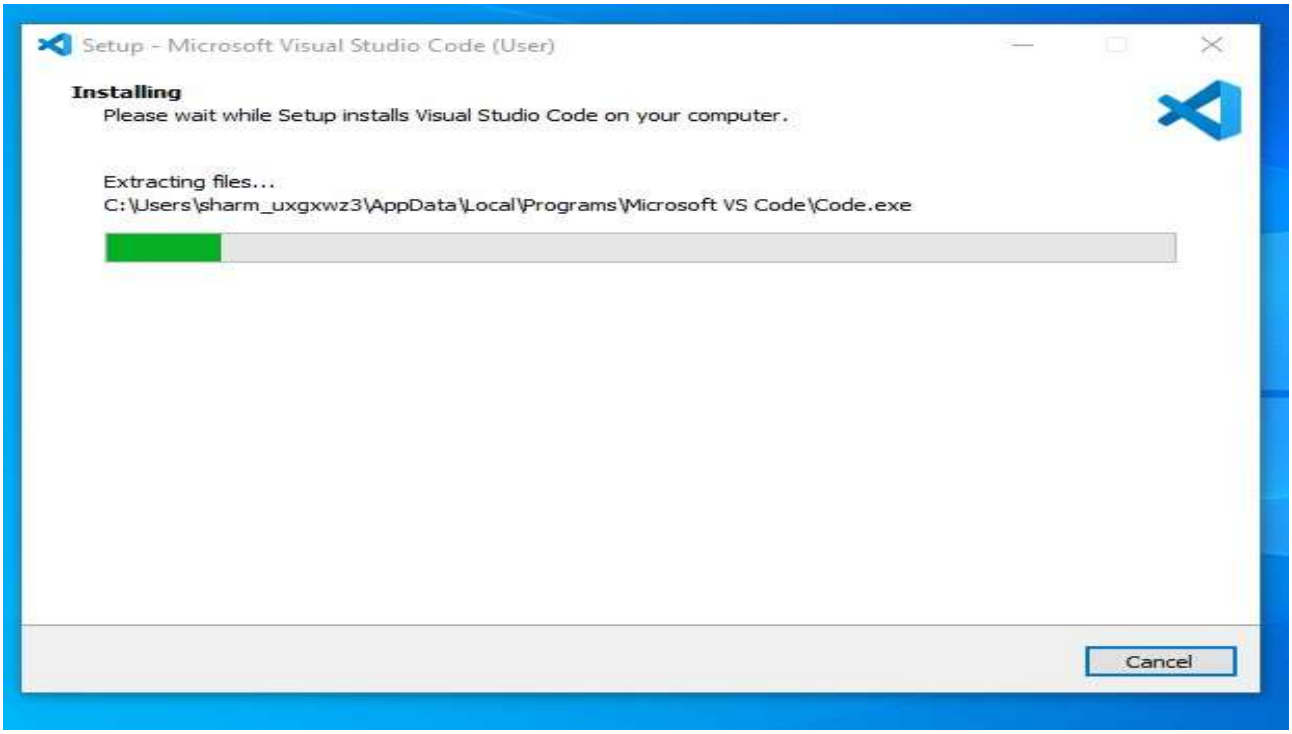

FIG:6.9 Installation of Visual Studio Code

Step 9: After the Installation setup for Visual Studio Code is finished, it will show a window like this below. Tick the "**Launch Visual Studio Code**" checkbox and then click **Next**.

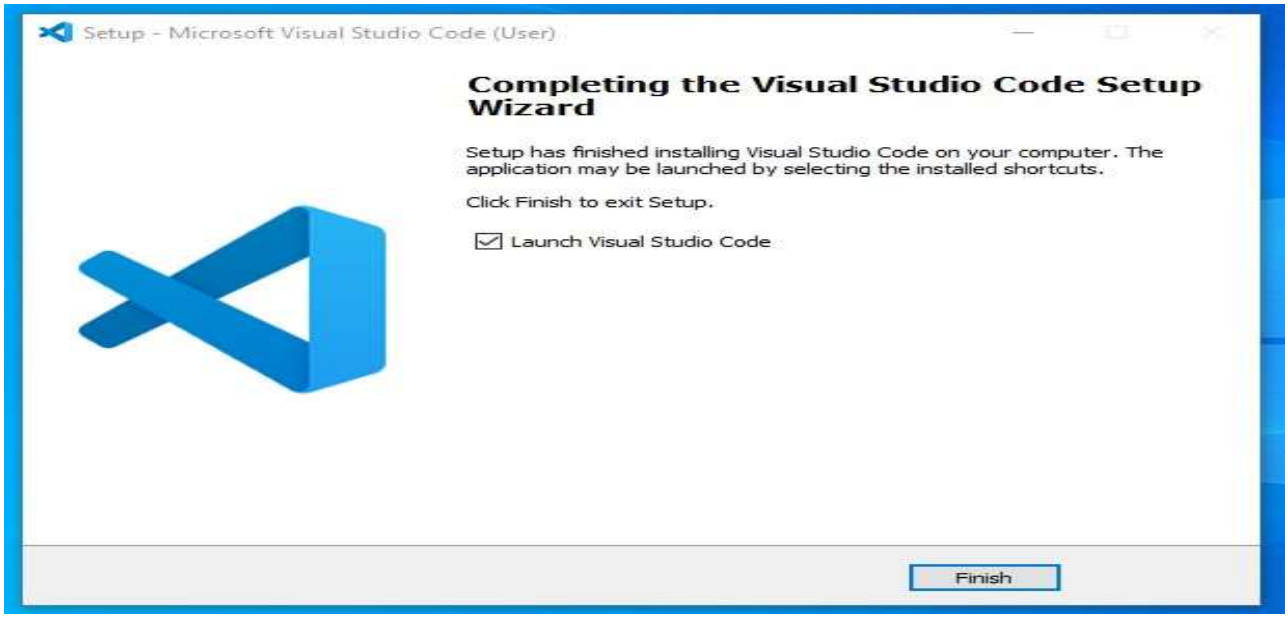

FIG:6.10 Installation Completed

Step 10: After the previous step, the **Visual Studio Code window** opens successfully. Now you can create a new file in the Visual Studio Code window and choose a language of yours to begin your programming journey!

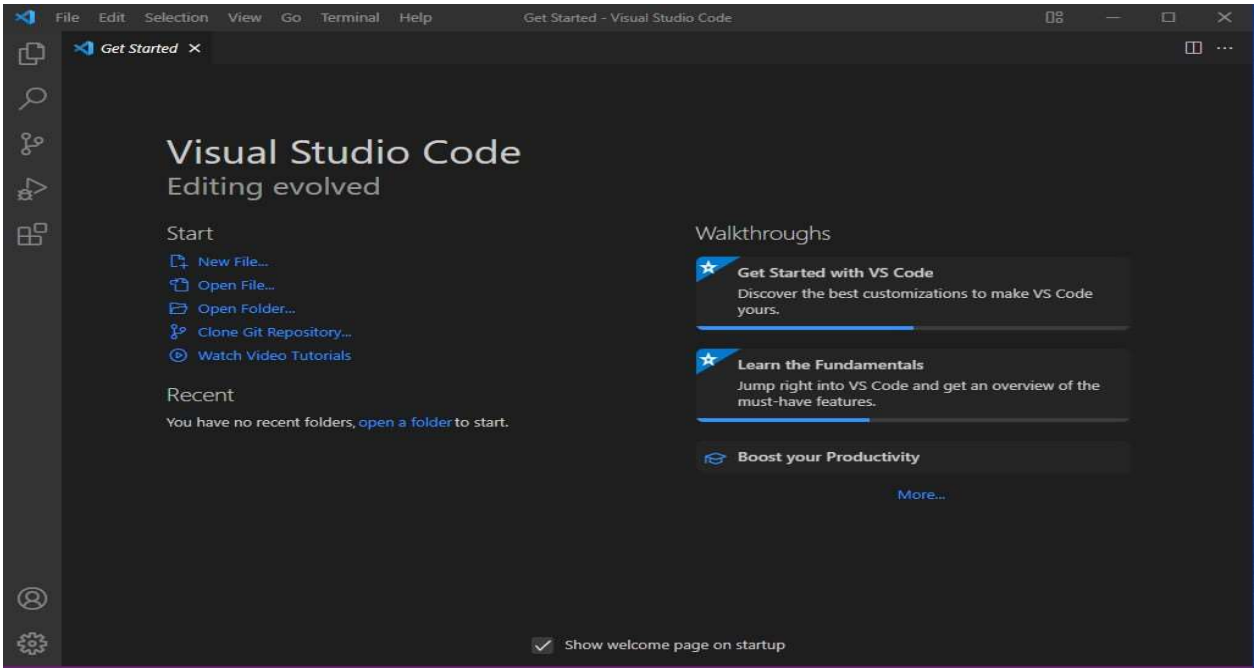

FIG:6.11 Visual Studio Code home page

6.2 CODING

6.2.1 Dottedline.py:

```
import cv2
import numpy as np
def drawline(img,pt1,pt2,color,thickness=1,style='dotted',gap=20):
    dist =((pt1[0]-pt2[0])**2+(pt1[1]-pt2[1])**2)**.5
    pts= []
    for i in  np.arange(0,dist,gap):
        r=i/dist
        x=int((pt1[0]*(1-r)+pt2[0]*r)+.5)
        y=int((pt1[1]*(1-r)+pt2[1]*r)+.5)
        p = (x,y)
        pts.append(p)

    if style=='dotted':
        for p in pts:
            cv2.circle(img,p,thickness,color,-1)
    else:
        s=pts[0]
        e=pts[0]
        i=0
        for p in pts:
            s=e
            e=p
            if i%2==1:
                cv2.line(img,s,e,color,thickness)
            i+=1

def drawpoly(img,pts,color,thickness=1,style='dotted',):
    s=pts[0]
    e=pts[0]
    pts.append(pts.pop(0))
    for p in pts:
        s=e
        e=p
        drawline(img,s,e,color,thickness,style)

def drawrect(img,pt1,pt2,color,thickness=1,style='dotted'):
    pts = [pt1,(pt2[0],pt1[1]),pt2,(pt1[0],pt2[1])]
      drawpoly(img,pts,color,thickness,style)
```

6.2.2 Handtracker.py:

```
import cv2
import mediapipe as mp
import math


class HandDetector:
    """
    Finds Hands using the mediapipe library. Exports the landmarks
    in pixel format. Adds extra functionalities like finding how
    many fingers are up or the distance between two fingers. Also
    provides bounding box info of the hand found.
    """

    def __init__(self, mode=False, maxHands=2, detectionCon=0.5, minTrackCon=0.5):
        """
        :param mode: In static mode, detection is done on each image: slower
        :param maxHands: Maximum number of hands to detect
        :param detectionCon: Minimum Detection Confidence Threshold
        :param minTrackCon: Minimum Tracking Confidence Threshold
        """
        self.mode = mode
        self.maxHands = maxHands
        self.detectionCon = detectionCon
        self.minTrackCon = minTrackCon

        self.mpHands = mp.solutions.hands
                        self.hands     =     self.mpHands.Hands(static_image_mode=self.mode,
max_num_hands=self.maxHands,min_detection_confidence=self.detectionCon,min_tracking_confid
ence=self.minTrackCon)
        self.mpDraw = mp.solutions .drawing_utils
        self.tipIds = [4, 8, 12, 16, 20]
        self.fingers = []
        self.lmList = []

    def findHands(self, img, draw_lm=True, draw_bbox=True, flipType=True):
        """
        Finds hands in a BGR image.
        :param img: Image to find the hands in.
        :param draw: Flag to draw the output on the image.
        :return: Image with or without drawings
        """
        imgRGB = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        self.results = self.hands.process(imgRGB)
        allHands = []
        h, w, c = img.shape
        if self.results.multi_hand_landmarks:
                        for   handType,   handLms   in   zip(self.results.multi_handedness,
self.results.multi_hand_landmarks):
                myHand = {}

                ## lmList
                mylmList = []
```

```
            xList = []
            yList = []
            for id, lm in enumerate(handLms.landmark):
                px, py, pz = int(lm.x * w), int(lm.y * h), int(lm.z * w)
                mylmList.append([px, py, pz])
                xList.append(px)
                yList.append(py)

            ## bbox
            xmin, xmax = min(xList), max(xList)
            ymin, ymax = min(yList), max(yList)
            boxW, boxH = xmax - xmin, ymax - ymin
            bbox = xmin, ymin, boxW, boxH
            cx, cy = bbox[0] + (bbox[2] // 2), bbox[1] + (bbox[3] // 2)
            bx, by = xmin, ymin
            tx, ty = xmax, ymax


            myHand["lmList"] = mylmList
            myHand["bbox"] = bbox
            myHand["center"] = (cx, cy)
            myHand["bottom"] = (bx, by)
            myHand["top"] = (tx, ty)

            if flipType:
                if handType.classification[0].label == "Right":
                    myHand["type"] = "Left"
                else:
                    myHand["type"] = "Right"
            else:
                myHand["type"] = handType.classification[0].label
            allHands.append(myHand)

            ## draw
            if draw_lm:
                self.mpDraw.draw_landmarks(img, handLms,
                                           self.mpHands.HAND_CONNECTIONS)
            if draw_bbox:
                cv2.rectangle(img, (bbox[0] - 20, bbox[1] - 20),
                     (bbox[0] + bbox[2] + 20, bbox[1] + bbox[3] + 20),(255, 0, 255), 2)



    if draw_lm or draw_bbox:
        return allHands, img
    else:
        return allHands

def fingersUp(self, myHand):

    """
    Finds how many fingers are open and returns in a list.
```

```
        Considers left and right hands separately
        :return: List of which fingers are up
        """
        myHandType = myHand["type"]
        myLmList = myHand["lmList"]
        if self.results.multi_hand_landmarks:
            fingers = []
            # Thumb
            if myHandType == "Right":
                if myLmList[self.tipIds[0]][0] > myLmList[self.tipIds[0] - 1][0]:
                    fingers.append(1)
                else:
                    fingers.append(0)
            else:
                if myLmList[self.tipIds[0]][0] < myLmList[self.tipIds[0] - 1][0]:
                    fingers.append(1)
                else:
                    fingers.append(0)

            # 4 Fingers
            for id in range(1, 5):
                if myLmList[self.tipIds[id]][1] < myLmList[self.tipIds[id] - 2][1]:
                    fingers.append(1)
                else:
                    fingers.append(0)
        return fingers

    def findDistance(self, p1, p2, img=None):
        """
        Find the distance between two landmarks based on their
        index numbers.
        :param p1: Point1
        :param p2: Point2
        :param img: Image to draw on.
        :param draw: Flag to draw the output on the image.
        """

        x1, y1 = p1
        x2, y2 = p2
        cx, cy = (x1 + x2) // 2, (y1 + y2) // 2
        length = math.hypot(x2 - x1, y2 - y1)
        info = (x1, y1, x2, y2, cx, cy)
        if img is not None:
            cv2.circle(img, (x1, y1), 15, (255, 0, 255), cv2.FILLED)
            cv2.circle(img, (x2, y2), 15, (255, 0, 255), cv2.FILLED)
            cv2.line(img, (x1, y1), (x2, y2), (255, 0, 255), 3)
            cv2.circle(img, (cx, cy), 15, (255, 0, 255), cv2.FILLED)
            return length, info, img


        else:
            return length, info
```

```
def findPosition(self, img, handNo=0, draw=True):
    xList = []
    yList = []
    bbox = []
    self.lmList = []
    if self.results.multi_hand_landmarks:
        myHand = self.results.multi_hand_landmarks[handNo]
        for id, lm in enumerate(myHand.landmark):
            # print(id, lm)
            h, w, c = img.shape
            cx, cy = int(lm.x * w), int(lm.y * h)
            xList.append(cx)
            yList.append(cy)
            # print(id, cx, cy)
            self.lmList.append([id, cx, cy])
            if draw:
                cv2.circle(img, (cx, cy), 5, (255, 0, 255), cv2.FILLED)

    xmin, xmax = min(xList), max(xList)
    ymin, ymax = min(yList), max(yList)
    bbox = xmin, ymin, xmax, ymax

    if draw:
        cv2.rectangle(img, (xmin - 20, ymin - 20), (xmax + 20, ymax + 20),
        (0, 255, 0), 2)

    return self.lmList, bbox


def main():
    cap = cv2.VideoCapture(0)
    detector = HandDetector(detectionCon=0.8, maxHands=2)
    while True:
        # Get image frame
        success, img = cap.read()
        # Find the hand and its landmarks
        hands, img = detector.findHands(img)  # with draw
        # hands = detector.findHands(img, draw=False)  # without draw

        if hands:
            # Hand 1
            hand1 = hands[0]
            lmList1 = hand1["lmList"]  # List of 21 Landmark points
            bbox1 = hand1["bbox"]  # Bounding box info x,y,w,h
            centerPoint1 = hand1['center']  # center of the hand cx,cy
            handType1 = hand1["type"]  # Handtype Left or Right

            fingers1 = detector.fingersUp(hand1)

            if len(hands) == 2:
                # Hand 2
                hand2 = hands[1]
```

```
                lmList2 = hand2["lmList"]  # List of 21 Landmark points
                bbox2 = hand2["bbox"]  # Bounding box info x,y,w,h
                centerPoint2 = hand2['center']  # center of the hand cx, cy
                handType2 = hand2["type"]  # Hand Type "Left" or "Right"

                fingers2 = detector.fingersUp(hand2)

                # Find Distance between two Landmarks. Could be same hand or different hands
                length, info, img = detector.findDistance(lmList1[8][0:2], lmList2[8][0:2],
img)  # with draw

        # Display
        cv2.imshow("Image", img)
        cv2.waitKey(1)

if __name__ == "__main__":
    main()
```

6.2.3 Main.py:

```
import cv2
import os
from HandTracker import HandDetector
from dottedline import drawrect, drawline
import numpy as np
import os
import win32com.client as win32

#slides to images
def convert_pptx_to_jpeg(input_file, output_folder):
    """
       Conversion of Slides to Images
        input_file (str): Path to the input PowerPoint presentation file.
        output_folder (str): Path to the folder where the output JPEG images will be saved.
    """
    # Remove existing images from the output folder
    for filename in os.listdir(output_folder):
        if filename.endswith('.jpg'):
            os.remove(os.path.join(output_folder, filename))

    ppt = win32.Dispatch('PowerPoint.Application')
    presentation = ppt.Presentations.Open(input_file)

    for slide_number in range(1, presentation.Slides.Count + 1):
        slide = presentation.Slides(slide_number)
        output_filename = os.path.join(output_folder, f"slide_{slide_number}.jpg")

        slide.Export(output_filename, "JPG")

    presentation.Close()
    ppt.Quit()
```

```
# Example usage
input_file = r"C:\Users\pokur\Desktop\PROJECT (MAIN)\hand gesture.pptx"
output_folder=r"C:\Users\pokur\Desktop\HandGestures\finalize\Hand-Gesture-Presentation
\Images”
convert_pptx_to_jpeg(input_file,output_folder)

# variables
width, height = 1280, 720
frames_folder = "Images"
slide_num = 0
hs, ws = int(120 * 1.2), int(213 * 1.2)
ge_thresh_y = 400
ge_thresh_x = 750
gest_done = False
gest_counter = 0
delay = 15
annotations = [[]]
annot_num = 0
annot_start = False

# Get list of presentation images
path_imgs = sorted(os.listdir(frames_folder), key=len)
print(path_imgs)

# Camera Setup
cap = cv2.VideoCapture(0)
cap.set(3, width)
cap.set(4, height)

# HandDetector
detector = HandDetector(detectionCon=0.8, maxHands=1)

while True:
    # Get image frame
    success, frame = cap.read()
    frame = cv2.flip(frame, 1)
    pathFullImage = os.path.join(frames_folder, path_imgs[slide_num])
    slide_current = cv2.imread(pathFullImage)
    slide_current = cv2.resize(slide_current, (1280, 720))

    # Find the hand and its landmarks
    hands, frame =  detector.findHands(frame)

    # Draw Gesture Threshold line
    drawrect(frame, (width, 0), (ge_thresh_x, ge_thresh_y), (0, 255, 0), 5,'dotted')

    if hands and gest_done is False:  # If hand is detected

        hand = hands[0]
        cx, cy = hand["center"]
        lm_list = hand["lmList"]  # List of 21 Landmark points
```

```
        fingers = detector.fingersUp(hand)

        # Constrain values for easier drawing
        x_val = int(np.interp(lm_list[8][0], [width//2, w], [0, width]))
        y_val = int(np.interp(lm_list[8][1], [150, height - 150], [0, height]))
        index_fing = x_val, y_val

        if cy < ge_thresh_y and cx > ge_thresh_x :
            annot_start = False

            # gest_1 (previous)
            if fingers == [1, 0, 0, 0, 0]:
                # print("Left")
                annot_start = False
                if slide_num > 0:
                    gest_done = True
                    slide_num -= 1
                    annotations = [[]]
                    annot_num = 0

            # gest_2 (next)
            if fingers == [0, 0, 0, 0, 1]:
                # print("Right")
                annot_start = False
                if slide_num < len(path_imgs) - 1:
                    gest_done = True
                    slide_num += 1
                    annotations = [[]]
                    annot_num = 0

            # gest_3 (clear screen)
            if fingers == [1, 1, 1, 1, 1]:
                if annotations:
                    annot_start = False
                    if annot_num >= 0:
                        annotations.clear()
                        annot_num = 0
                        gest_done = True
                        annotations = [[]]

        # gest_4 (show pointer)
        if fingers == [0, 1, 0, 0, 0]:
            cv2.circle(slide_current, index_fing, 4, (0, 0, 255), cv2.FILLED)
            annot_start = False



        # gest_5 (draw)
        if fingers == [0, 1, 1, 0, 0]:
            if annot_start is False:
                annot_start = True
                annot_num += 1
```

```
                annotations.append([])
            # print(annot_num)
            annotations[annot_num].append(index_fing)
            cv2.circle(slide_current, index_fing, 4, (0, 0, 255), cv2.FILLED)

        else:
            annot_start = False

        # gest_6 (erase)
        if fingers == [0, 1, 1, 1, 0]:
            if annotations:
                annot_start = False
                if annot_num >= 0:
                    annotations.pop(-1)
                    annot_num -= 1
                    gest_done = True

    else:
        annot_start = False



    # Gesture Performed Iterations:
    if gest_done:
        gest_counter += 1
        if gest_counter > delay:
            gest_counter = 0
            gest_done = False

    for i, annotation in enumerate(annotations):
        for j in range(len(annotation)):
            if j != 0:
                  cv2.line(slide_current, annotation[j - 1], annotation[j], (0, 0, 255), 6)

    # Adding cam img on slides
    img_small = cv2.resize(frame, (ws, hs))
    h, w, _ = slide_current.shape
    slide_current[h-hs:h, w-ws:w] = img_small
    cv2.imshow("Slides", slide_current)
    # cv2.imshow("Image", frame)
    key = cv2.waitKey(1)
    if key == ord('q'):
        break
```

6.3 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

6.4 TYPES OF TESTING

6.4.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on

knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.4.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems.

6.4.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.
Invalid Input : identified classes of invalid input must be rejected.
Functions : identified functions must be exercised.
Output : identified classes of application outputs must be exercised.
Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing.

6.4.4 SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

6.4.5 WHITE BOX TESTING

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

6.4.6 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a test in which the software under test is treated as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

6.4.7 ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

6.4.8 TESTING RESULTS

All the test cases mentioned above passed successfully. No defects encountered.

6.5 TESTING METHODOLOGIES

The following are the Testing Methodologies:

- Unit Testing.
- Integration Testing.
- User Acceptance Testing.
- Output Testing.
- Validation Testing.

6.5.1 UNIT TESTING

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually, and the module interfaces are verified for the consistency with design specification. All-important processing path are tested for the expected results. All error handling paths are also tested.

6.5.2 INTEGRATION TESTING

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

1)Top-Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with

the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2) Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom-up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program.

Structure

The bottom-up approach tests each module individually and then each module is module is integrated with a main module and tested for functionality.

6.5.3 USER ACCEPTANCE TESTING

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required.

The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

6.5.4 OUTPUT TESTING

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the

system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

6.5.5 VALIDATION TESTING

Validation Checking:

Validation checks are performed on the following fields.

Using Live Test Data:

In our project, the utilization of live test data in validation testing represents a fundamental aspect of ensuring the efficacy and reliability of our real-time interactive Presentation Control System. By integrating live data into our testing procedures, we embark on a journey to scrutinize the system's performance in authentic, real-world contexts.

Moreover, live test data enables us to simulate diverse user interactions with the system in real-time. We observe how users navigate through slides, annotate content, and interact with the presentation interface using intuitive hand movements. This firsthand insight allows us to fine-tune the system's user experience, ensuring that it remains intuitive and user-friendly across different usage scenarios.

Furthermore, the iterative nature of live test data allows us to continuously improve the system over time. By collecting user feedback and analyzing their interactions with the system in real-time, we identify areas for enhancement and implement iterative improvements accordingly. This iterative approach ensures that the system evolves to meet the evolving needs and expectations of its users, ultimately enhancing its overall quality and reliability.

In essence, the integration of live test data into our validation testing process serves as a cornerstone in our quest to deliver a truly exceptional presentation control system. By subjecting the system to real-world scenarios, we validate its performance, fine-tune its user experience, and iteratively improve its reliability, ensuring that it remains at the forefront of innovation in presentation technology.

6.6 Test Cases:

Test Case#	TC 1
Test Name	Next Slide
Test Description	To Navigate to the Next Slide [0, 0, 0, 0, 1]
Input	Only Little Finger Up Gesture as Input
Output	Navigating to the Next Slide Implemented
Test Result	Success

Table 6.1 : Test Case 1

Test Case#	TC 2
Test Name	Previous Slide
Test Description	To Navigate to the Previous Slide [1, 0, 0, 0, 0]
Input	Only Thumb Finger Up Gesture as Input
Output	Navigating to the Previous Slide Implemented
Test Result	Success

Table 6.2 : Test Case 2

Test Case#	TC 3
Test Name	Pointer
Test Description	To Obtain Pointer [0, 1, 0, 0, 0]
Input	Only Index Finger Up Gesture as Input
Output	Obtaining Pointer Implemented
Test Result	Success

Table 6.3: Test Case 3

Test Case#	TC 4
Test Name	Draw Annotations
Test Description	To Draw Annotations on the Slides [0, 1, 1, 0, 0]
Input	Index and Middle Fingers Up Gesture as Input
Output	Drawing Annotations on Slides Implemented
Test Result	Success

Table 6.4: Test Case 4

Test Case#	TC 5
Test Name	Undo
Test Description	To Undo Annotations on the Slide [0, 1, 1, 1, 0]
Input	Index, Middle and Ring Fingers Up Gesture as Input
Output	Undo actions Implemented
Test Result	Success

Table 6.5: Test Case 5

Test Case#	TC 6
Test Name	Clear
Test Description	To Clear all Annotations on Slides [1, 1, 1, 1, 1]
Input	All Fingers Up Gesture as Input
Output	Clear Annotations Implemented
Test Result	Success

Table 6.6: Test Case 6

Chapter 7
SCREEN SHOTS

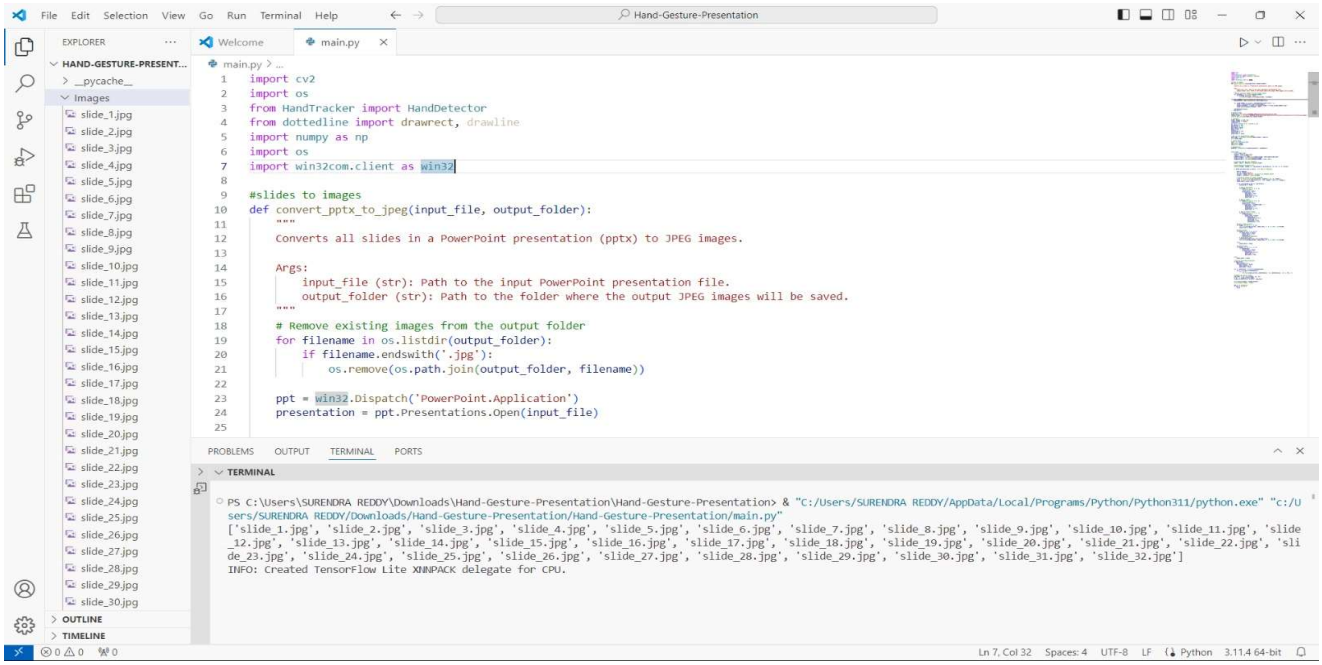

FIG:7.1 converting slides to images and saving images on images folder

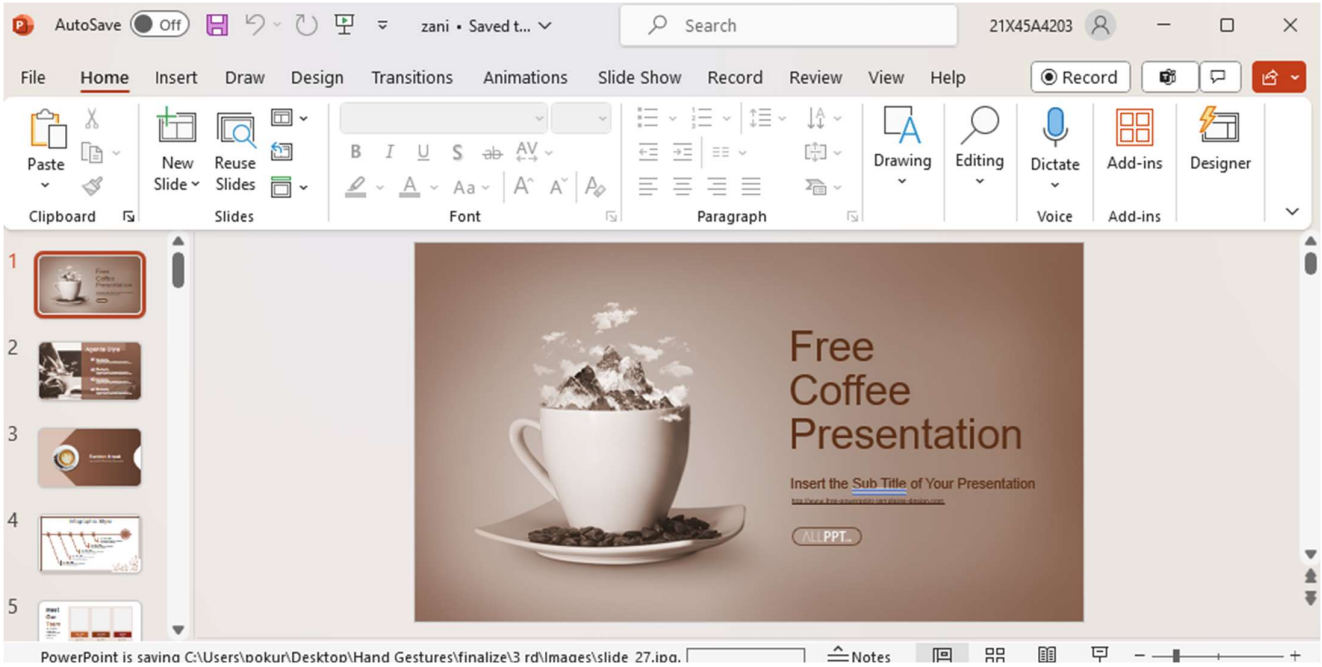

FIG:7.2 slides to images conversion

FIG:7.3 to next slide

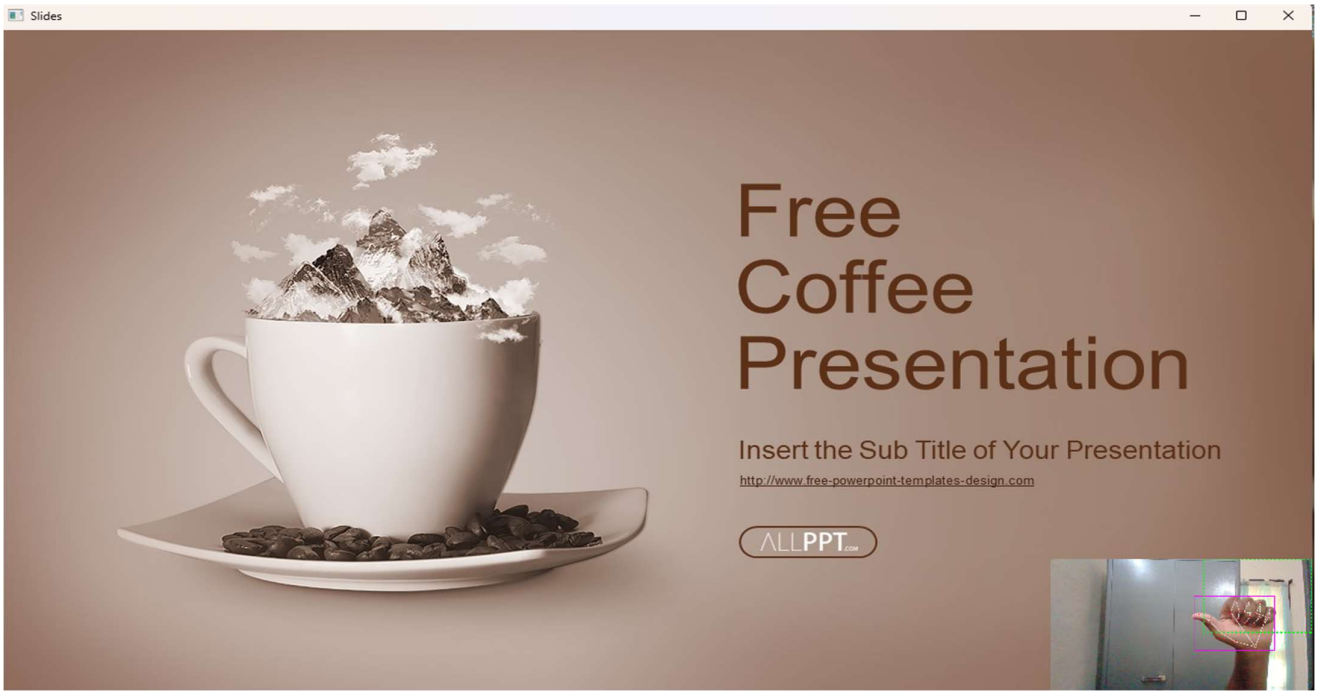

FIG:7.4 to Previous slide

FIG:7.5 to obtain pointer

FIG:7.6 to draw on the slides

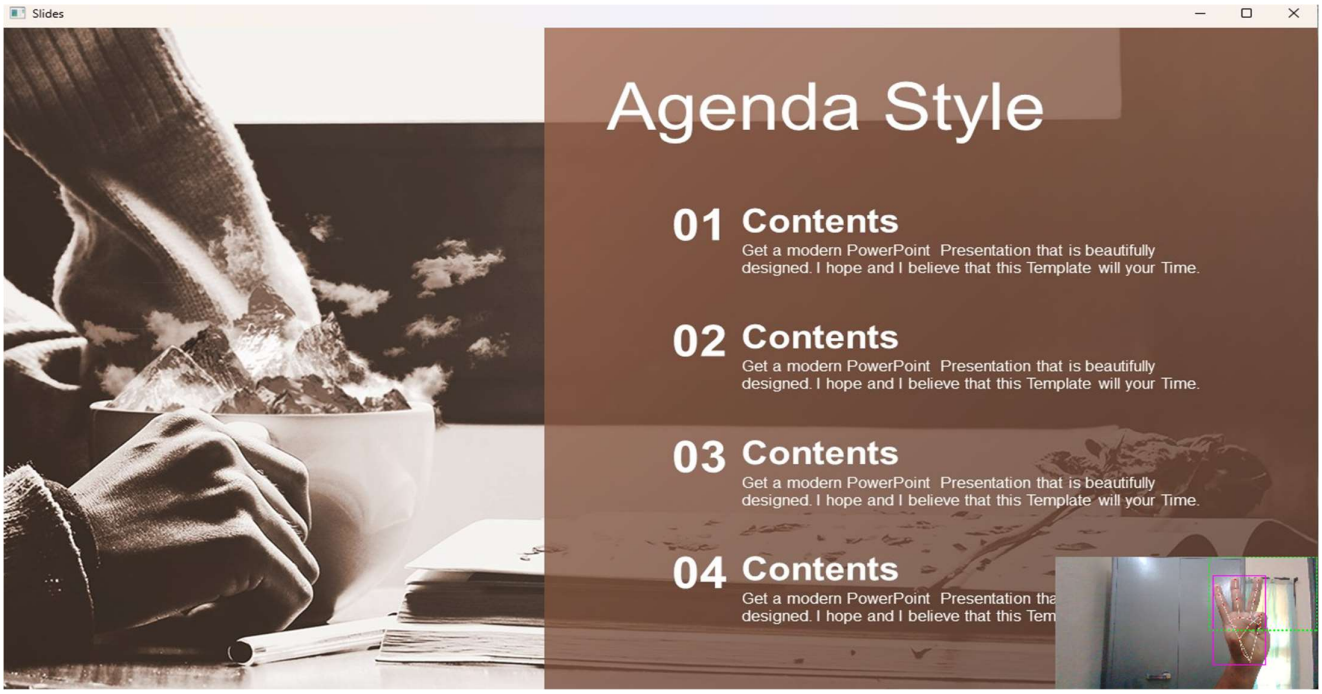

FIG:7.7 undo

FIG:7.8 Clear Screen

Chapter 8
CONCLUSION

In conclusion, the integration of real-time hand tracking and gesture recognition technologies into PowerPoint presentations represents a significant advancement in the realm of interactive communication. By leveraging these innovative technologies, users can transcend the limitations of traditional slide-based presentations and engage their audience in a more dynamic and immersive manner.

This project's core functionality, centered around detecting and interpreting specific hand gestures, empowers presenters to navigate slides, annotate content, and interact with their presentations seamlessly. By eliminating the reliance on conventional input devices like keyboards or mice, this intuitive interaction mechanism not only enhances user experience but also fosters active audience participation and engagement.

Furthermore, the adoption of such interactive presentation systems not only enhances the effectiveness of conveying information but also opens up new avenues for creativity and expression in communication. As technology continues to evolve, the potential for further innovation in this field is vast, promising even more captivating and immersive presentation experiences in the future.

Chapter 9
FUTURE SCOPE

Gestures are utilized in numerous disciplines and have considerable value. Gestures are the future of real time interactions. According to current circumstances, there is a need for a more natural communication method with computers and technology. The technology aids pupils with human-computer interaction when gestures come into play. By adding additional movements, we can control computer programs like cut, copy, paste, etc. We can expand our system to manage the PowerPoint application as well. The same technology or algorithm may be employed for any objective, rather than multiple approaches for each goal.

Chapter 10
REFRENCES

1. harris, m., & agoes, a. s. (2021, november). applying hand gesture recognition for user guide application using mediapipe. in 2nd international seminar of science and applied technology (issat 2021) (pp. 101-108). atlantis press.

2. sung, g., sokal, k., uboweja, e., bazarevsky, v., baccash, j., bazavan, e. g., … & grundmann, m. (2021). on-device real-time hand gesture recognition. arxiv preprint arxiv:2111.00038.

3. cohen, c. j., beach, g., & foulk, g. (2001, october). a basic hand gesture control system for pc applications. in proceedings 30th applied imagery pattern recognition workshop (aipr 2001). analysis and understanding of time varying imagery (pp. 74-79). ieee.

4. zeng, b., wang, g., & lin, x. (2012). a hand gesture based interactive presentation system utilizing heterogeneous cameras. tsinghua science and technology, 17(3), 329-336.

5. 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.

6. ahamed, s. f., sandeep, p., tushar, p., & srithar, s. (2023, january). efficient gesture-based presentation controller using transfer learning algorithm. in 2023 international conference on computer communication and informatics (iccci) (pp. 1-5). ieee.

7. charan, c. s., meenakshi, k., reddy, v. b., & kashyap, v. (2023, april). controlling powerpoint presentation using hand gestures in real- time. in 2023 7th international conference on trends in electronics and informatics (icoei) (pp. 251-254). ieee.

8. vidya, m., vineela, s., sathish, p., & reddy, a. s. (2023, august). gesture-based control of presentation slides using opencv. in 2023 second international conference on augmented intelligence and sustainable systems (icaiss) (pp. 1786-1791). ieee.

9. mali, c., sayyad, b., ankushe, v., navghane, a., kulkarni, s., & prawin gawande, p. g. (2023). design and implementation of hand gesture assistant command control video player interface for physically challenged people. available at ssrn 4626576

10. kumar, p., jaiswal, a., deepak, b., & reddy, g. r. m. (2018). hand gesture-based stable powerpoint presentation using kinect. in progress in intelligent computing techniques: theory, practice, and applications: proceedings of icacni 2016, volume 1 (pp. 81-94). springer Singapore.

11. Hajeera Khanum, Pramod H.B, Smart presentation control using hand gestures, IRJET (2022).

12. Meera Paulson, Natasha, Shilpa Davis on the Smart presentation using gesture recognition and OpenCV, Asian Journal of Convergence in Technology, Vol 5, (2019)

13. Mujahid, A., et al.: Real-time hand gesture recognition based on deep learning YOLOv3 model. Appl. Sci. 11(9), 4164 (2021).

14. Viraj Shinde, Tushar Bacchav, Jitendra Pawar, Mangesh Sanap, Hand recognition system using camera, Navsahyadri education society, IJERT (2020)

15. Dnyanda R. Jadhav, L. M. Lobo, Navigation of Power point using hand gestures, Walchand Institute of technology, Solapur IJSR (2018)

16. D. Jadhav, Prof. L.M.R.J. Lobo, Hand Gesture Recognition System to Control Slide Show Navigation IJAIEM, Vol. 3, No. 4 (2014)

17. D.O. Lawrence, and Dr. M.J. Ashleigh, Impact Of Human-Computer Interaction (Hci) on Users in Higher Educational System: Southampton University As A Case Study, Vol.6, No 3, pp. 1-12, September (2019)

18. S.S. Abhilash, L. Thomas, N. Wilson and C. Chaithanya, "Virtual Mouse Using Hand Gesture", International Research Journal of Engineering and Technology (IRJET), vol. 5, no. 4, pp. 3903-3906, 2018

PAPER PUBLICATION REPORT

CERTIFICATES OF AUTHORS

IJESAT International Journal of Engineering, Science and Advanced Technology

ISSN:2250-3676, Web:https://ijesat.com/

UGC CARE APPROVED GROUP 'A' JOURNAL

CERTIFICATE OF PUBLICATION

This is to certify that the paper entitled

GESTURE DRIVEN PRESENTATION CONTROL

Authored by

D.SIRISHA

From

SRK Institute of Technology

Has been published in

IJESAT JOURNAL, VOLUME 24, ISSUE 05, MAY - 2024

Editors-in-Chief

Dr.Domniz & Dr.Anand

IJESAT

IJESAT International Journal of Engineering, Science and Advanced Technology

ISSN:2250-3676, Web:https://ijesat.com/

UGC CARE APPROVED GROUP 'A' JOURNAL

CERTIFICATE OF PUBLICATION

This is to certify that the paper entitled

GESTURE DRIVEN PRESENTATION CONTROL

Authored by

G.SURENDRA REDDY

From

SRK Institute of Technology

Has been published in

IJESAT JOURNAL, VOLUME 24, ISSUE 05, MAY - 2024

Editors-in-Chief
Dr.Domniz & Dr.Anand
IJESAT

IJESAT International Journal of Engineering, Science and Advanced Technology

ISSN:2250-3676, Web:https://ijesat.com/

UGC CARE APPROVED GROUP 'A' JOURNAL

CERTIFICATE OF PUBLICATION

This is to certify that the paper entitled

GESTURE DRIVEN PRESENTATION CONTROL

Authored by

R.USHA SRI

From

SRK Institute of Technology

Has been published in

IJESAT JOURNAL, VOLUME 24, ISSUE 05, MAY - 2024

Editors-in-Chief
Dr.Domniz & Dr.Anand
IJESAT

IJESAT International Journal of Engineering, Science and Advanced Technology

ISSN:2250-3676, Web:https://ijesat.com/

UGC CARE APPROVED GROUP 'A' JOURNAL

CERTIFICATE OF PUBLICATION

This is to certify that the paper entitled

GESTURE DRIVEN PRESENTATION CONTROL

Authored by

R.HEMANTH

From

SRK Institute of Technology

Has been published in

IJESAT JOURNAL, VOLUME 24, ISSUE 05, MAY - 2024

Editors-in-Chief
Dr.Domniz & Dr.Anand
IJESAT

IJESAT International Journal of Engineering, Science and Advanced Technology

ISSN:2250-3676, Web:https://ijesat.com/

UGC CARE APPROVED GROUP 'A' JOURNAL

CERTIFICATE OF PUBLICATION

This is to certify that the paper entitled

GESTURE DRIVEN PRESENTATION CONTROL

Authored by

P.MOHANA KALYAN

From

SRK Institute of Technology

Has been published in

IJESAT JOURNAL, VOLUME 24, ISSUE 05, MAY - 2024

Editors-in-Chief
Dr.Domniz & Dr.Anand
IJESAT

PUBLISHED PAPER DOCUMENT

GESTURE DRIVEN PRESENTATION CONTROL

D.SIRISHA[1], G.SURENDRA REDDY[2], R.USHA SRI[3], R.HEMANTH[4], P.MOHANA KALYAN[5]

[1]Assisant Professor, Department of CSE-Artificial Intelligence and Machine Learning, S.R.K Institute of Technology, NTR, Andhra Pradesh, India.
[2,3,4,5]Student, Department of CSE-Artificial Intelligence and Machine Learning, S.R.K Institute of Technology, NTR, Andhra Pradesh, India.

ABSTRACT: - This project presents a real-time interactive Presentation Control System that redefines the conventional method of delivering presentations through innovative hand gesture recognition. Utilizing the robust hand tracking capabilities of OpenCV and the precise gesture recognition functionalities of Mediapipe, the system empowers users to seamlessly navigate through slides, annotate content, and interact with presentations using intuitive hand movements. By overlaying the live camera feed onto the presentation interface, users can visualize their gestures alongside the slides in real-time, enhancing the interactive experience. The system interprets specific gestures, such as using the little finger to advance to the next slide and the thumb to go back to the previous slide. Additionally, users can utilize the index finger and middle finger to draw annotations, while the index finger alone serves as a pointer. To undo actions, users can gesture with the index, middle, and ring fingers, while raising all fingers clears annotations. This novel approach bridges the gap between traditional presentation tools and modern interactive interfaces, offering a more efficient and engaging way to deliver presentations, particularly in scenarios where conventional input devices may be impractical or cumbersome. With its combination of OpenCV and Mediapipe libraries, the system ensures high accuracy and responsiveness, enabling smooth and intuitive control of presentations through natural hand movements

KEYWORDS: - OpenCV, Mediapipe, Hand Gesture Recognition, Presentation Controller.

I. INTRODUCTION

The development of this real-time interactive Presentation Control System involved several key steps to ensure its effectiveness and usability. Initially, our project team conducted extensive research into computer vision techniques, focusing particularly on hand tracking and gesture recognition algorithms. This laid the foundation for the system's functionality, which would rely on accurately interpreting user hand movements. Our team then proceeded to integrate OpenCV and MediaPipe libraries into the system, harnessing the robust hand tracking capabilities of OpenCV and the precise gesture recognition functionalities of MediaPipe.

Once the foundational software components were in place, we began designing the user interface and interaction flow. This involved mapping out how users would navigate through slides, annotate content, and interact with the presentation using intuitive hand gestures. The goal was to create a seamless and intuitive user experience that would enhance engagement and productivity during presentations. Overlaying the live camera feed onto the presentation interface was a crucial aspect of this design, as it allowed users to visualize their gestures alongside the slides in real-time, providing immediate feedback on their interactions.

The implementation phase involved coding and testing the various features of the system, ensuring that hand gestures were accurately interpreted and translated into the desired actions. Specific gestures, such as using the little finger to advance to the next slide and the thumb to go back to the previous slide, were programmed into the system to provide users with intuitive control over the presentation. Additionally, the ability to draw annotations using the forefinger and middle finger, as well as using the forefinger as a pointer, added further functionality to the system.

Throughout the development process, we conducted thorough testing to identify and address any bugs or issues. This iterative approach allowed for continual refinement and improvement of the system's performance and reliability. Finally, with the combination of OpenCV and MediaPipe libraries, the system achieved high accuracy and responsiveness, enabling smooth and intuitive control of presentations through natural hand movements. This project represents a significant advancement in presentation technology, offering a more efficient and engaging way to deliver presentations in various settings.

II. EXISTING METHOD

Traditional presentation control systems rely on manual input devices, lacking intuitiveness and interactivity. Remote control software exists but requires additional hardware or software installations, adding complexity. Standalone interactive whiteboards lack integration with popular presentation software like PowerPoint. Existing systems offer varying interactivity but suffer from limitations like complexity and lack of integration. This project proposes a solution for controlling presentations using hand gestures, aiming for intuitiveness, accessibility, and integration.

Disadvantages:

1. Limited Interaction, 2. Cost of Additional Hardware, 3. Lack of Customization Options, 4. Complex Setup

III. Proposed System

Our proposed system represents a paradigm shift in presentation control, utilizing state-of-the-art hand tracking and gesture recognition technology to provide a seamless and intuitive user experience. By harnessing the power of natural hand gestures, presenters can effortlessly navigate slides, annotate content, and engage with their audience in real-time. We have used OpenCV and Mediapipe in our project. OpenCV has more than 2,500 algorithms in which CNN works efficiently to detect hands and Mediapipe to Recognize Gestures of detected hand. Below, we detail the key components and functionalities of the proposed system:

1. Integration with Presentation Software: Our system seamlessly integrates with popular presentation software platforms, such as Microsoft PowerPoint. Presenters can control the presentation flow directly from the integrated interface, eliminating the need for external input devices.

2. Live Camera Feed Overlay: The system incorporates a live camera feed overlay onto the presentation interface, providing visual feedback of the presenter's gestures in real-time. This feature enhances presenter awareness and facilitates smoother interaction with the presentation content.

3.Real-Time Annotation and Interaction: Presenters can annotate slides in real-time using intuitive hand gestures, enhancing the clarity and impact of their presentations. Gestures for drawing and erasing enable dynamic content customization and emphasis during the presentation.

Advantages of Proposed System:

1. Intuitive interaction: The proposed system enables users to control presentations using natural hand gestures, making the interaction more intuitive and engaging.

2. Enhanced Flexibility: Unlike traditional input devices such as keyboards or mice, hand gestures allow for more dynamic and expressive control over presentation content. Users can easily navigate slides, annotate content, and interact with audience members in real-time.

3. Customizable Gestures: Unlike existing systems with predefined gesture sets, the proposed system allows users to define and customize their own gestures according to their preferences and presentation style.

4. Effortless Setup: The proposed system offers a straightforward setup process, requiring minimal hardware and software configuration.

IV. SYSTEM ARCHITECTURE

The system architecture consists of the following components:

1. Video Input (Webcam): This is the source of data for the system. It captures video input in real-time using a webcam.
2. Hand Tracking (OpenCV): This component is responsible for tracking hands in the video input. It uses OpenCV, an open-source computer vision library, to perform this task.
3. Hand Landmark Extraction (MediaPipe): Once the hands are tracked, the next step is to extract key landmarks from the hand. This is done using MediaPipe, a framework for building multimodal applied machine learning pipelines.
4. Training Data: The extracted hand landmarks are used as input for training a machine learning model. The details of the training process are not provided in the context.
5. Gesture Classification and Recognition: After training, the model is used to classify and recognize gestures based on the hand landmarks.
6. Slide and Annotation Control: Based on the recognized gestures, the system can control slides and annotations. This could be useful in a presentation or a teaching environment.
7. Real-time Display: The final component of the system is the real-time display of the video input, hand tracking, hand landmark extraction, and the results of gesture classification and recognition.

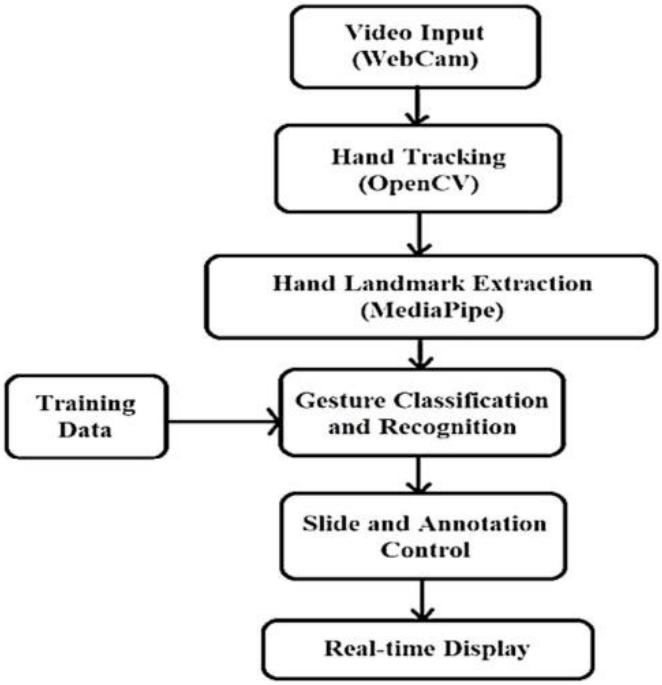

Fig 1: Architecture Diagram.

V. METHODOLOGY

The proposed system is implemented using Python and makes use of computer vision techniques to detect and classify hand gestures. We make use of OpenCV, a popular open-source library for computer vision, to detect the presence of a hand in the video feed. Once a hand is detected, we use a Hand Module to detect and classify the hand gestures. Data is trained using a vector set of hand gestures, which includes examples of various gestures such as

changing slides, next slide, previous slide, pointing, and highlight points. Hand gesture recognition is done using Python programming language and OpenCV as library. Python programming language produces simple and easy system code to understand. Also, Python package used here is NumPy. The image that is captured using web camera will be processed in a region called as Region of Interest (ROI) where acts as a region of wanted area while ignoring the outside region, called background.

The system follows a systematic flow to enable seamless interaction with presentation slides:
i. Hand Detection: The system utilizes OpenCV to capture live video input from a webcam. By analyzing the video frames, the system detects and identifies the presenter's hand within the captured images.

ii. Hand Tracking: Once the hand is detected, MediaPipe is employed to track the hand movements and extract hand landmarks. These landmarks represent specific points on the hand, such as fingertips and joints, which will be used for gesture recognition.

iii. Gesture Recognition: By analyzing the positions and movements of the hand landmarks, the system accurately recognizes predefined gestures. Each gesture corresponds to a specific action within the presentation software. The gestures supported are as follows:
Gesture 1: Thumb Finger - Move to Previous Slide Gesture
Gesture 2: Little Finger - Move to Next Slide Gesture
Gesture 3: Index Finger and Middle Finger Together - Drawing on the Slide Gesture
Gesture 4: Index Finger - Holding the Pointer
Gesture 5: Middle Three Fingers - Undo the Previous Annotation
Gesture 6: All Five Fingers Open – Clear All Annotations

A. Slide Navigation: The system recognizes the Thumb Finger gesture, allowing presenters to move to the previous slide. Similarly, the Little Finger gesture enables presenters to progress to the next slide, providing seamless slide navigation.
B. Pointer Control: Presenters can hold a virtual pointer by bringing the Index Finger. This gesture enables them to highlight specific areas of the slide, directing the audience's attention and emphasizing key points. The dynamic pointer control adds an interactive element to the presentation.
C. Drawing and Annotations: The system enables presenters to draw on the slide using the Index finger and Middle finger gesture. By moving their finger across the screen, presenters can create real-time annotations, underline important details, or emphasize specific elements. This feature allows for on-the-fly visual enhancements and effective communication of the content.
D. Erasing: To remove or revise previous annotations, presenters can utilize the Middle Three Fingers gesture. This gesture activates the erasing function, allowing presenters to effortlessly erase specific annotations or clear the entire slide, ensuring a clean and polished presentation.
E. Clear: To clear the annotations , presenters can utilize all five fingers up gesture. This gesture enables clearing all the annotations on the slides.

Tools and Technologies :
1. OpenCV:
OpenCV is a robust open-source computer vision and machine learning library that provides a plethora of tools and algorithms for various vision-related tasks. It offers functionalities ranging from image processing and object detection to video analysis and camera calibration. OpenCV is widely used for real-time applications due to its efficiency and extensive documentation.

With pre-trained models and algorithms, developers can quickly implement features like object detection, tracking, and recognition. Its versatility makes it suitable for a wide range of applications, from robotics and augmented reality to healthcare and automotive systems.

2. Mediapipe :

Mediapipe developed by Google, is a modular framework designed for building machine learning pipelines for media processing tasks. It offers pre-trained models and a streamlined development process, making it easier for developers to create real-time applications for tasks like hand tracking, pose estimation, and face detection. Mediapipe's integration with TensorFlow allows for the seamless incorporation of custom machine learning models, expanding its capabilities beyond the provided modules. With its focus on real-time performance and cross-platform support, Mediapipe is well-suited for applications in fields such as augmented reality, human-computer interaction, and gesture recognition. MediaPipe Hands is a high-fidelity hand and finger tracking solution. It employs machine learning (ML) to infer 21 3D landmarks of a hand from just a single frame.

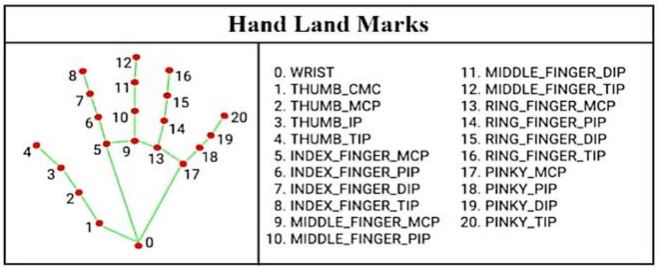

FIG 2: Representation of hand landmarks

Gestures Performed:

Gesture 1: to next slide. [0,0,0,0,1]

Only the little finger is open in this gesture, while the other four fingers are all closed.

FIG 3: to next slide

Gesture 2: to Previous slide. [1,0,0,0,0]

Only the thumb is open in this gesture, while the other fingers are all closed.

FIG 4: to previous slide

Gesture3:toobtainthe pointer[0,1,0,0,0]

Only the index finger is open in this gesture, while the other fingers are all closed.

FIG 5: to obtain the Pointer

Gesture4: to draw on the slides. [0,1,1,0,0]

In this gesture the index finger and middle finger are open, the other fingers are all closed.

FIG 6: to Draw on the slides.

Gesture 5: Undo. [0,1,1,1,0]

In this gesture the index finger, middle finger, and ring finger are open, all the other fingers are closed. The most recent written part is to be erased with this gesture.

FIG 7: undo

Gesture 6: Clear Screen. [1,1,1,1,1]

In this gesture, all the fingers are open. It clears all the annotations drawn on the Screen.

FIG 8: clear screen

VI. RESULTS

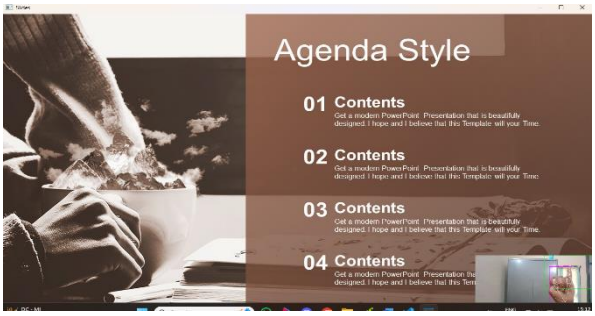

FIG 9: to next slide

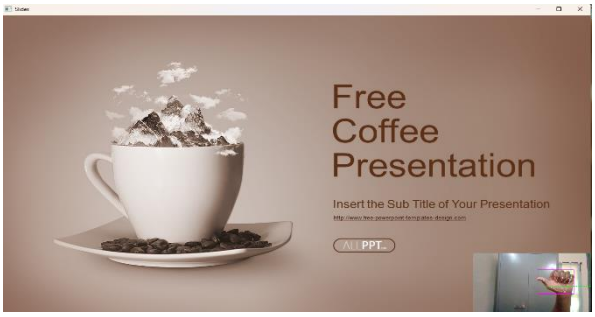

FIG 10: to Previous slide

FIG 11: to obtain pointer.

FIG 12: to draw on the slides.

FIG 13: undo

FIG 14: Clear Screen

VII. CONCLUSION

In conclusion, the integration of real-time hand tracking and gesture recognition technologies into PowerPoint presentations represents a significant advancement in the realm of interactive communication. By leveraging these innovative technologies, users can transcend the limitations of traditional slide-based presentations and engage their audience in a more dynamic and immersive manner.

This project's core functionality, centered around detecting and interpreting specific hand gestures, empowers presenters to navigate slides, annotate content, and interact with their presentations seamlessly. By eliminating the reliance on conventional input devices like keyboards or mice, this intuitive interaction mechanism not only enhances user experience but also fosters active audience participation and engagement.

Furthermore, the adoption of such interactive presentation systems not only enhances the effectiveness of conveying information but also opens up new avenues for creativity and expression in communication. As technology continues to evolve, the potential for further innovation in this field is vast, promising even more captivating and immersive presentation experiences in the future.

VIII. FUTURE DIRECTIONS

Gestures are utilized in numerous disciplines and have considerable value. Gestures are the future of real time interactions. According to current circumstances, there is a need for a more natural communication method with computers and technology. The technology aids pupils with human-computer interaction when gestures come into play. By adding additional movements, we can control computer programs like cut, copy, paste, etc. We can expand our system to manage the PowerPoint application as well. The same technology or algorithm may be employed for any objective, rather than multiple approaches for each goal.

IX. REFERENCES

1. harris, m., & agoes, a. s. (2021, november). applying hand gesture recognition for user guide application using mediapipe. in 2nd international seminar of science and applied technology (issat 2021) (pp. 101-108). atlantis press.
2. sung, g., sokal, k., uboweja, e., bazarevsky, v., baccash, j., bazavan, e. g., … & grundmann, m. (2021). on-device real-time hand gesture recognition. arxiv preprint arxiv:2111.00038.
3. cohen, c. j., beach, g., & foulk, g. (2001, october). a basic hand gesture control system for pc applications. in proceedings 30th applied imagery pattern recognition workshop (aipr 2001). analysis and understanding of time varying imagery (pp. 74-79). ieee.
4. zeng, b., wang, g., & lin, x. (2012). a hand gesture based interactive presentation system utilizing heterogeneous cameras. tsinghua science and technology, 17(3), 329-336.
5. 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.

6. ahamed, s. f., sandeep, p., tushar, p., & srithar, s. (2023, january). efficient gesture-based presentation controller using transfer learning algorithm. in 2023 international conference on computer communication and informatics (iccci) (pp. 1-5). ieee.
7. charan, c. s., meenakshi, k., reddy, v. b., & kashyap, v. (2023, april). controlling powerpoint presentation using hand gestures in real- time. in 2023 7th international conference on trends in electronics and informatics (icoei) (pp. 251-254). ieee.
8. vidya, m., vineela, s., sathish, p., & reddy, a. s. (2023, august). gesture-based control of presentation slides using OpenCV. in 2023 second international conference on augmented intelligence and sustainable systems (icaiss) (pp. 1786-1791). ieee.
9. mali, c., sayyad, b., ankushe, v., navghane, a., kulkarni, s., & prawin gawande, p. g. (2023). design and implementation of hand gesture assistant command control video player interface for physically challenged people. available at ssrn 4626576
10. kumar, p., jaiswal, a., deepak, b., & reddy, g. r. m. (2018). hand gesture-based stable powerpoint presentation using kinect. in progress in intelligent computing techniques: theory, practice, and applications: proceedings of icacni 2016, volume 1 (pp. 81-94). springer Singapore.
11. Hajeera Khanum, Pramod H.B, Smart presentation control using hand gestures, IRJET (2022).
12. Meera Paulson, Natasha, Shilpa Davis on the Smart presentation using gesture recognition and OpenCV, Asian Journal of Convergence in Technology, Vol 5, (2019)
13. Mujahid, A., et al.: Real-time hand gesture recognition based on deep learning YOLOv3 model. Appl. Sci. 11(9), 4164 (2021).
14. Viraj Shinde, Tushar Bacchav, Jitendra Pawar, Mangesh Sanap, Hand recognition system using camera, Navsahyadri education society, IJERT (2020)
15. Dnyanda R. Jadhav, L. M. Lobo, Navigation of Power point using hand gestures, Walchand Institute of technology, Solapur IJSR (2018)
16. D. Jadhav, Prof. L.M.R.J. Lobo, Hand Gesture Recognition System to Control Slide Show Navigation IJAIEM, Vol 3,No.4(2014)
17. D.O. Lawrence, and Dr. M.J. Ashleigh, Impact of Human-Computer Interaction (Hci) on Users in Higher Educational System: Southampton University as a Case Study, Vol.6, No 3, pp. 1-12, September (2019)
18. S.S. Abhilash, L. Thomas, N. Wilson, and C. Chaithanya, "Virtual Mouse Using Hand Gesture", International Research Journal of Engineering and Technology (IRJET), vol. 5, no. 4, pp. 3903-3906, 2018