Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 2 (2024)

# Enhancing Human-Computer Interaction: Integrating Voice Assistants and Gesture Recognition for Innovative Mouse Control

**[1]Vinay Kumar, [2]Ankith H. Bharadwaj, [3]Dr. Mahanthesha U.,**
*[1]Department of AIML*
*B N M Institute of Technology Bengaluru, India*
*[2]Department of AIML*
*B N M Institute of Technology Bengaluru, India*
*[3]Department of AIML*
*B N M Institute of Technology Bengaluru, India*

**Abstract:-** Voice assistants, like Alexa, Cortana, Google Assistant, and Siri, have become commonplace and provide consumers with a handy voice-activated interface for using their gadgets. These software agents are capable of understanding spoken language, reacting via voice synthesizers, and carrying out a range of functions, including as answering inquiries, organizing calendars, and operating smart home appliances. In addition, hand gesture recognition—which is comparable to the biometric authentication seen in smartphones—has become a crucial component of human-computer interaction. This research suggests an innovative mouse control method by utilizing computer vision algorithms and real-time video feeds. The system translates hand gestures into mouse operations like clicking and scrolling by using image segmentation and gesture recognition, rather than modifying the mouse hardware or adding more buttons. This creative method shows how bringing together modern technologies can improve computing experiences while also streamlining user engagement.

**Keywords:** Voice Assistants; chat bots; Static voice assistant, Speech Recognition, Text-to-Speech, Computer Vision, Open CV, Deep Learning, Image Processing

## Introduction

Voice assistants have become a necessary part of our everyday lives. They do this through utilizing synthesis and language processing algorithms to comprehend and respond to user-provided intents, or precise voice commands. These assistants are built into an array of gadgets, including PCs, smart speakers, smartphones, including Alexa, Cortana, Google Assistant, and the just recently developed AIVA. They search for pertinent keywords and perform various tasks, such as ordering pizza and sending messages, while background noise plays. People with physical limitations can benefit most from this technology since it makes them more accessible. Computer mice have evolved over time, moving from mechanical to optical, laser, and even wireless models. Even with developments in precision, robustness, and wireless capabilities, mice continue to be hardware tools with constraints. But developments like eye tracking and speech recognition point to possible alternatives to conventional input techniques. This virtualization trend is exemplified by speech recognition, which can replace keyboards, and eye tracking, which replace mouse control.

Requiring fewer processing power, hand gesture recognition is another new interface. For gesture recognition, a variety of methods are used, frequently with the use of specialized equipment like data gloves or color caps, such as hand image or pixel recognition. Computer vision libraries such as OpenCV and pyautogui are used in the field of virtual mouse. Webcam images are captured by OpenCV, while keyboard and mouse operations are made easier via PyAutogui. Human-computer interaction is becoming increasingly vital as technology develops; touch screens are common in mobile devices but less so in desktop systems owing to financial limitations. Here, computer vision provides an alternative by allowing webcams to be used to create virtual mice and keyboards. Applications for virtual mice that use finger tracking and are intended for interacting with computers are prime

examples of this strategy. These applications improve accessibility and usability by utilizing computer vision techniques, addressing an array of user needs in a rapidly changing technological environment.

The main objectives of the project are as follows:

Voice Assistant Development: Using the Python programming language, create a static voice assistant with the ability to execute commands only by speech. It will have features for file manipulation, texting, and online ordering. Make sure the helper can comprehend ordinary English.

Hardware Integration: Provide the voice assistant with seamless connectivity and interoperability with a range of gadgets, including PCs and cell phones. Adopt strong interface designs and communication protocols to improve user accessibility across many platforms and devices.

Speech Recognition: To efficiently interpret and process user commands, incorporate precise speech recognition algorithms into the voice assistant. Make use of cutting- edge machine learning models and approaches to enhance the assistant's comprehension of various languages, accents, and speech patterns.

Focus on Accessibility: Tailor the voice assistant's features and functions to the unique requirements of people with physical disabilities, giving priority to improvements in both usability and accessibility. To ensure an inclusive user experience, including voice commands and interaction techniques designed with users with disabilities in mind.

Gesture-Controlled Virtual Mouse: Create a virtual mouse programmed that records and analyses hand gestures from a webcam stream using computer vision techniques like OpenCV. Develop hand gesture recognition and interpretation algorithms so that users can manipulate mouse operations using natural hand motions, such as clicking and scrolling. Assure the virtual mouse application's smooth interaction with current software environments and systems.

## 1. Literature Survey

Voice-activated assistants have taken the world of human-computer interaction by storm, becoming widely integrated into everything from speakers and smartwatches to televisions and smartphones. The growing need for voice assistants on many platforms highlights the necessity for sophisticated solutions to effectively manage large volumes of data. Machine learning, IoT, NLP, and big data access management have become essential elements in augmenting voice assistant skills, allowing them to accurately comprehend and carry out user commands.

The Artificial Intelligence - Training with Intelligent Personal Assistant track by Nil Goksel et al. examines the use of intelligent personal assistants (IPAs) powered by state-of-the-art computer technologies and natural language processing (NLP). This article explores the inner workings of IPAs in the context of AI, highlighting how their personalized learning capabilities have the potential to completely transform user interactions. The creation of intelligent home systems utilizing Wi-Fi and IoT technologies can be aided by smart assistants, as demonstrated in the study "Smart Home Using Internet of Things" by Keerthana S et al. The system makes use of temperature-sensing Wi-Fi-enabled microcontroller units (MCUs) to monitor and control electrical equipment in real-time, opening the door to effective energy management and home automation.

Similar to this, Sutar Shekhar et al.'s "An Intelligent Voice Assistant Using Android Platform" promotes the incorporation of voice commands into mobile devices in order to improve user experience and expedite daily operations. By offering tailored recommendations based on user activity patterns, prediction technology is incorporated to further enhance user connection.

In addition, Rishabh Shah et al.'s "An Intelligent Chatbot using Natural Language Processing" emphasizes how crucial NLP is to the development of advanced voice assistants that can comprehend commands in a variety of languages. This inclusiveness guarantees more usability and accessibility for a wider range of user demographics.

"JARVIS: An interpretation of AIML with integration of gTTS and Python" by Tanvee Gawand et al. explores the merging of technologies such as gTTS and AIML. The research shows the possibilities for developing flexible voice assistants with offline capabilities using dynamic Python modules for text-to-speech conversion and AIML integration.All things considered, the convergence of advances in NLP, IoT, and speech recognition technology offers a promising path towards the creation of intelligent voice assistants that improve user engagement and expedite routine activities. Voice assistants can continue to develop as essential tools in contemporary computer environments by utilizing these advancements.

## 2. Proposed Systems

In an effort to completely reinvent human- computer interaction, the suggested system provides a thorough integration of contemporary voice assistant technology with real-time camera- based mouse control. It combines the features of well-known voice assistants, such as Google Assistant, Amazon's Alexa, Microsoft's Cortana, and Apple's Siri, to allow users to carry out a variety of tasks using natural language commands. Advanced computer vision techniques, such as image segmentation and gesture recognition, are incorporated into the system simultaneously to enable mouse control through hand movements recorded by a webcam. This creative method offers voice and gesture- based input methods for improved accessibility and usability, giving users a smooth and natural way to interact with their devices.

The proposed architecture

The first step in the system design is to use a microphone to capture speech patterns as input. 2. Text conversion and audio data recognition.

3. Evaluating the input against preset directives. 4. Producing the intended results.

The first stage involves the data being ingested as microphone speech patterns. In the second stage, NLP is used to process the gathered data and turn it into textual data. The necessary output process is completed in the following step by manipulating the data using Python script on the resultant string. During the final stage, the generated output is either displayed as text or as speech using text-to-speech technology. Features The following features will be included in the development of the system:

1) It remains silent while listening all the time and responds to a call with a specific preset function. 2) Navigating the web using the user's spoken parameters, receiving a desired output via audio, and simultaneously printing the output on the screen.

3) Uses computer vision methods to track hand movements captured by a webcam, such as gesture recognition and image segmentation.

4) Offers a natural substitute for traditional mouse input techniques, especially helpful for people who need hands-free interaction or have mobility issues.

5) Incorporates virtual mouse implementation and color detection algorithms to guarantee responsiveness and accuracy.

## 3. Methodology

The suggested system makes use of the pyttsx3 library and the SAPI5 interface for text-to-speech conversion, enabling offline functionality and supporting Python 2 and 3. The core of the system's voice capabilities is provided by Microsoft's SAPI (Speech Application Programming Interface), which allows speech synthesis and recognition within Windows applications. The primary function of the system is where its capabilities are defined. The system is intended to:

(a) Request input from the user and listen for commands continuously for a configurable amount of time.

(b) If a command is not understood or is not clear, ask the user to repeat it.

(c) Provide voice customization options, letting users select between a male and a female voice.

(d) Support a number of functions, such as opening applications, checking the time, sending and receiving emails, searching Wikipedia, taking and displaying notes, and managing YouTube and Google apps.

The system incorporates OpenCV, an open- source library primarily devoted to image processing and video

capture, to enable real-time computer vision capabilities. Numpy is used by OpenCV, a Python, C++, and Java programe, to perform high-level mathematical operations and manipulate multi-dimensional arrays. It is perfect for many applications, including face detection, optical character recognition, and vision-guided robotic surgery, because of its features, which include object detection, motion estimation, background subtraction, and object tracking. Anaconda is a free software framework for data science, machine learning, and predictive analysis that runs on the Python and R programming languages. For managing packages and starting apps, Anaconda offers two tools: Anaconda Prompt, a command prompt, and Anaconda Navigator, a desktop GUI application. Package management and teamwork are made easier with the variety of Python environments, notebooks, and packages that Anaconda Cloud offers.

The system runs on powerful, GPU-compatible systems to meet the computational demands of the models for best performance. Furthermore, the system incorporates the OpenAI API key to enable access to OpenAI's advanced natural language processing capabilities. The user experience and system performance are improved by the smooth integration of voice commands, computer vision, and machine learning functionalities made possible by this complete setup.

```
import speech_recognition as sr
import os
import webbrowser
import openai
from config import apikey
import datetime
import random
```

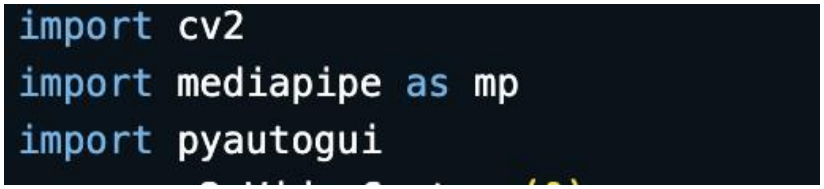
**Fig 4.1 Libraries used for Voice Assistant.**

```
from keras.models import load_model
from time import sleep
from tensorflow.keras.utils import img_to_array
from keras.preprocessing import image
import cv2
import numpy as np
```

**Fig 4.2 Libraries used for Models.**

```
import os
import openai
from config import apikey
openai.api_key = apikey
```

**Fig 4.3 Libraries used for using the openai API key**




.

**Fig 4.4 Libraries used for gesture mouse.**

From the Fig 4.1 to 4.3 the voice assistant app we're developing includes a number of essential modules to guarantee its efficacy and functionality:

- Speech Recognition Module: This module sets up voice interaction by allowing the assistant to comprehend spoken commands. It guarantees that the helper can appropriately identify and react to user input.
- DateTime Module: This module improves the assistant's usefulness for tasks requiring temporal context by providing information about the current date and time. It does this by utilizing the built-in datetime package.
- Wikipedia Module: When the assistant is integrated with the Wikipedia module, it has access to an extensive knowledge base from which it can retrieve data or conduct searches on a range of subjects.
- The PyAudio Module offers Python bindings for PortAudio, which facilitates the assistant's ability to record and handle audio input. This is crucial for tasks involving speech recognition and other audio-related applications.
- The PyQt5 Module provides thorough Python bindings for the Qt framework, making it easier to create GUIs (graphical user interfaces) for assistant applications. It improves visual feedback and user interaction.
- Python Backend: The assistant's central component, the Python backend coordinates multiple modules and manages context extraction, API calls, and command processing. It guarantees smooth communication between the application's various components.
- Text-to-Speech Module: This module allows the assistant to speak to users by translating textual output it generates into spoken words. Its aural feedback improves accessibility and user experience.
- Speech-to-Text Conversion: The assistant can effectively understand and process spoken commands by turning spoken input from users into text thanks to speech recognition capabilities. The foundation for voice communication with the assistant is this module.

By combining these modules, the voice assistant app gains functionality and strength, enabling users to perform tasks, understand spoken commands, retrieve information, and receive interactive feedback via text and speech interfaces.

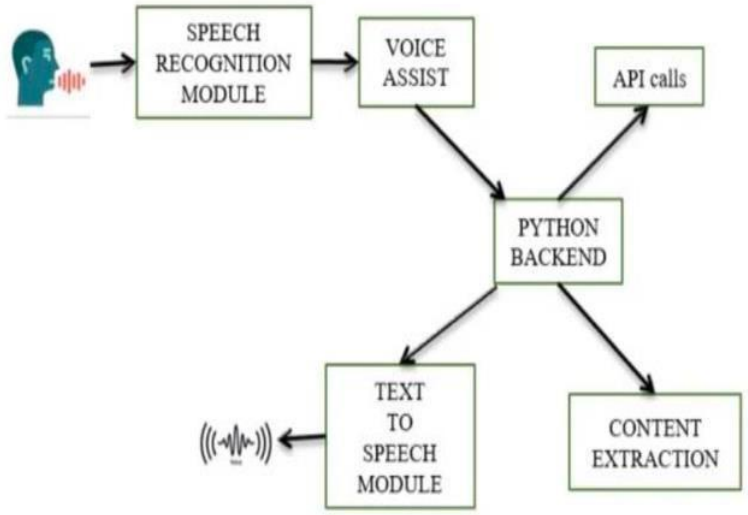
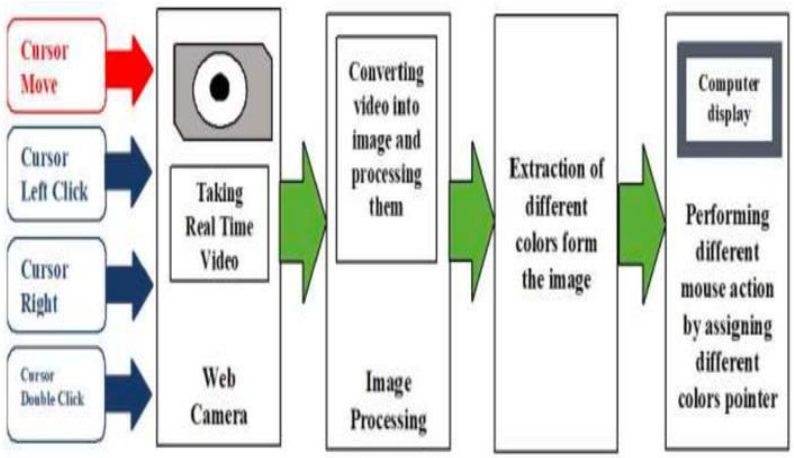



**Fig 4.4 Detailed workflow of the Speech Recognition of the model.**

From Fig 4.4 the description is that n our phones, voice assistants like Bixby, Google Voice, and Siri is already available. A recent NPR study indicates that about one in six Americans already own a smart speaker, like the Google Home or Amazon Echo, and that sales of these devices are increasing at a pace comparable to that of smartphones ten years ago. However, in the workplace, the voice revolution might still seem far off. One disincentive is the shift towards open workspaces: no one wants to be that annoying moron who is always shouting at his virtual assistant.

**Fig 4.5 Detailed workflow of the gesture mouse.**




From Fig 4.5 we can understand that we need a sensor to identify the user's hand movements in order for the system to function. The computer's webcam serves as a sensor. The hardware of the webcam controls the fixed frame rate and resolution at which real-time video is captured. If necessary, the system can adjust the frame rate and resolution. The Real Time Video is recorded using the webcam on a computer. The FPS (frames per

second) of the camera determines how image frames are separated from video.The system runs in the Anaconda environment and makes use of libraries like Numpy, and OpenCV as well as the corresponding sub-packages for each. Using the built-in system camera, the camera resolution is set to 1920 x 1080 pixels at a frame rate of 40 frames per second.The system initiates a camera interface upon activation in order to obtain user input. Users can define specific gestures and commands that go along with them by configuring the model for gesture recognition. After processing these gestures, the system follows user-defined commands to carry out the appropriate actions.Without the need for human input, the system simultaneously tracks and records mouse pointer movements. The system can be interacted with effortlessly thanks to this automated functionality, which improves efficiency and user experience.

## 4. Results

Promising results have been obtained when the assistant first starts up, it waits for user input. When a voice command is given, the assistant records it and looks for pertinent keywords in the command. After locating a pertinent keyword, the helper performs the associated action and outputs in both text and audio within the terminal window. To maintain a responsive and engaging user experience, the assistant reverts to waiting for user input if no valid keyword is found.

The system's primary functionalities rely on models created with Numpy and OpenCV. These models incorporate mouse movement based on user-defined colour highlights. The code implementation for colour detection and mouse movement is based on tracking corresponding movements, denoted by highlighted regions, and detecting specific colours in the camera feed. By using visual cues, this method allows for easy and natural interaction with the system, allowing for smooth control and operation.


**Figure 5.1: The helper will bide its time until the user issues a vocal wake-up command.**

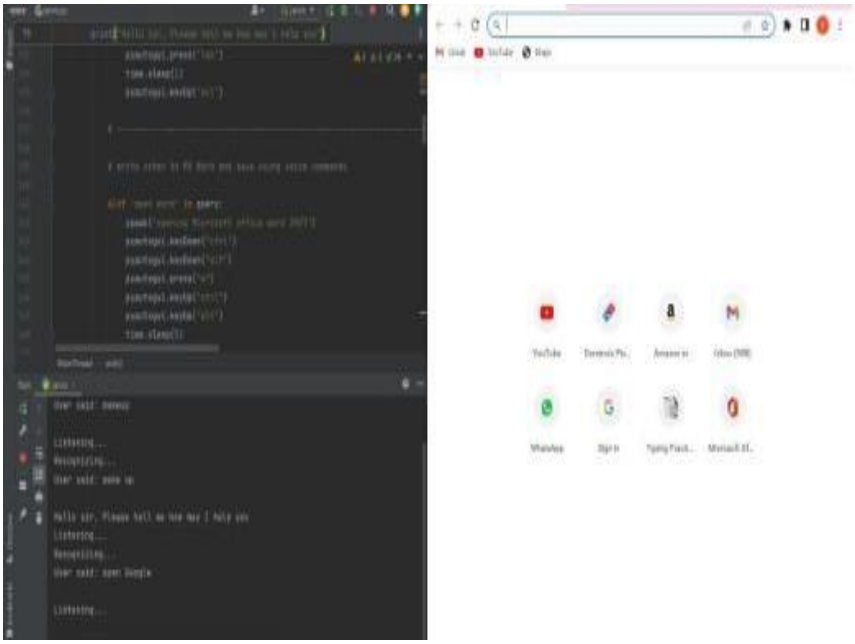


**Fig. 5.2: The assistant launches Chrome.**

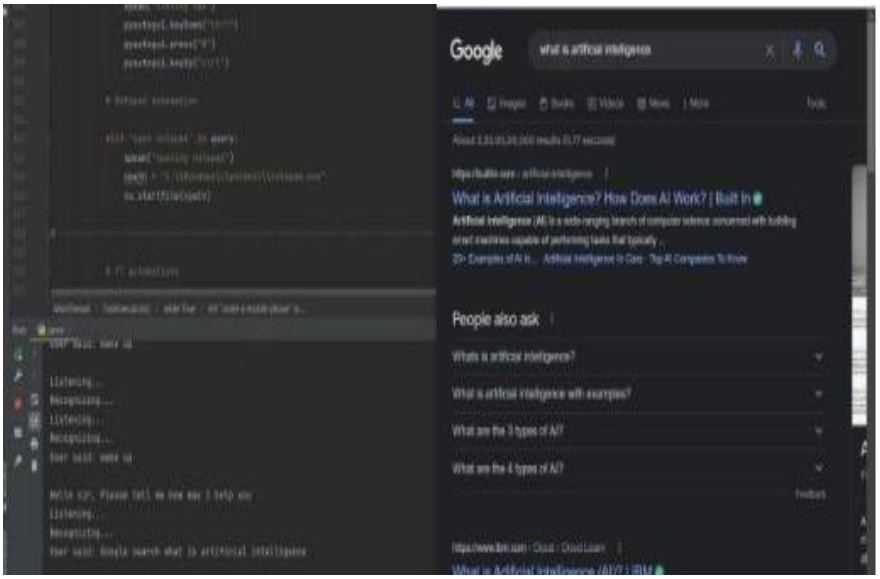



**Fig. 5.3: Assistant searching Google for data**

.

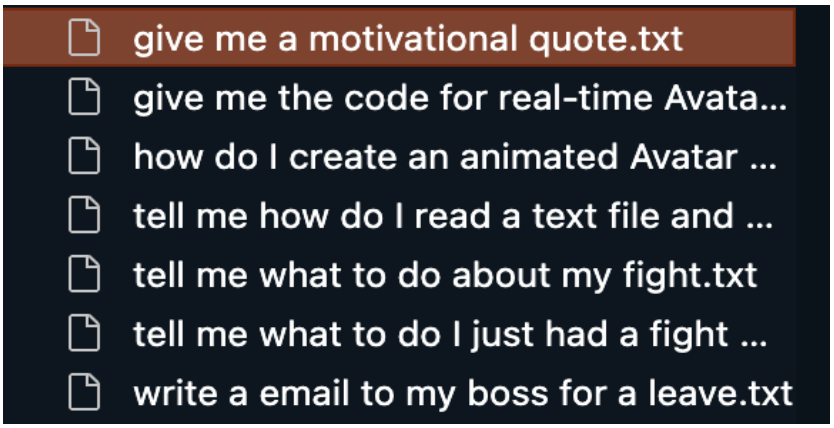



**Fig 5.4 To Show how the users prompt is stores as a Text file.**

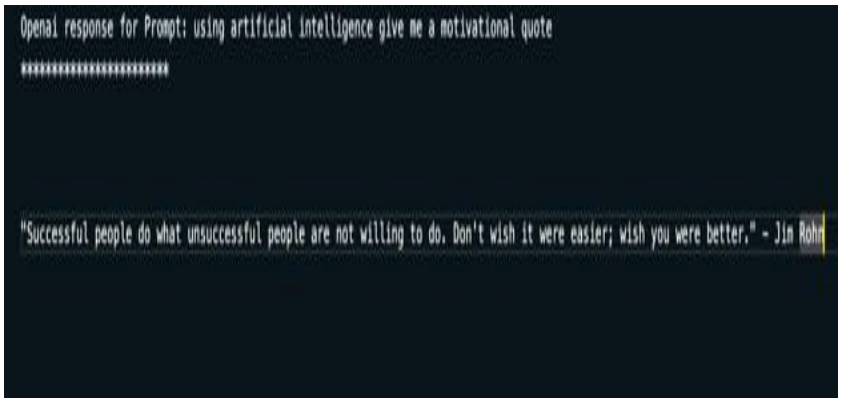



**Fig 5.5 To Show the response generated when the user asked for motivational quote.**

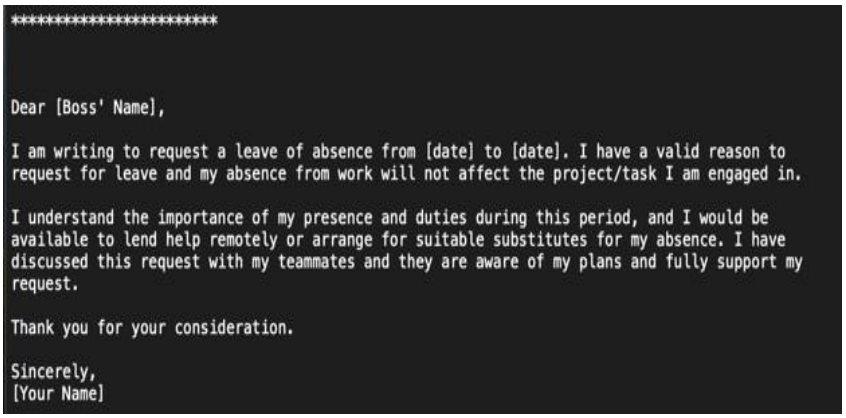



**Fig 5.6 Prompt generated when the user asked to give an e-mail for a leave to the boss.**

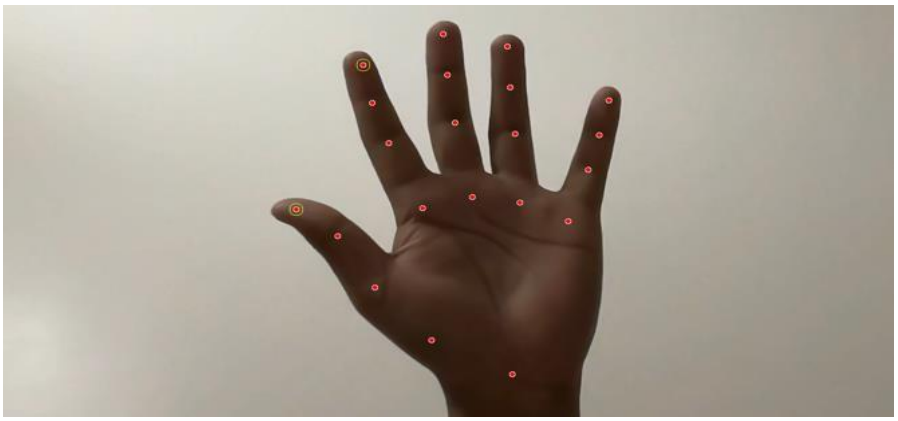


**Fig 5.7 This the image where the hand of the user is detected and ready to take the gestures.**

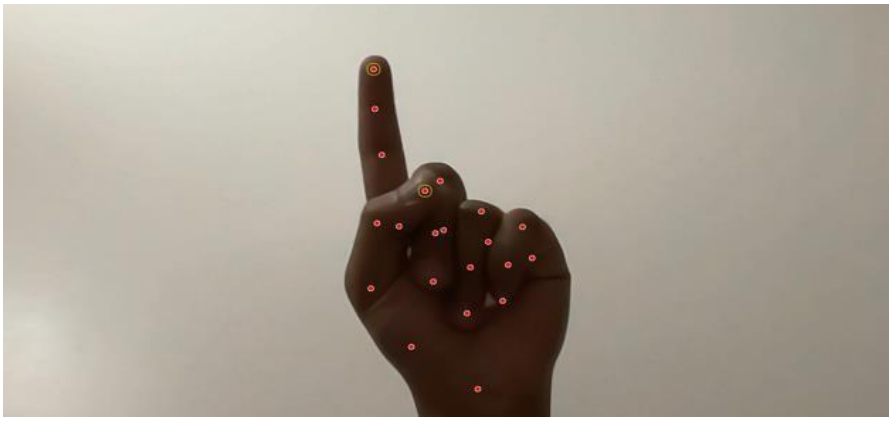


**Fig 5.8 This is image is when the cursor is moving with the hand gestures.**

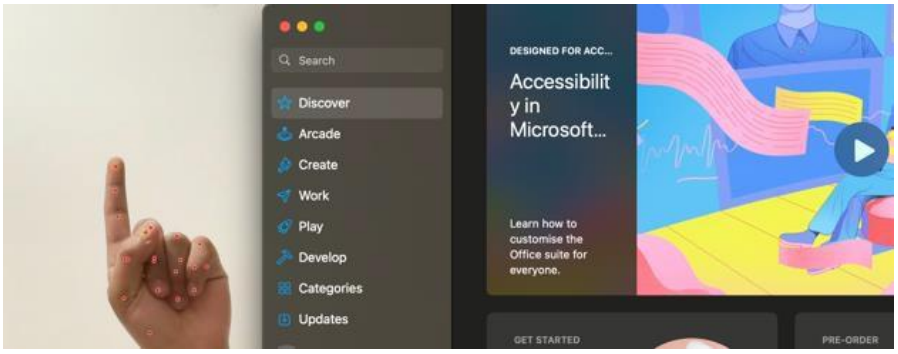



**Fig 5.9 To show when the cursor is moved on the close button to close an application using hand gesture.**

**Fig 5.10 to show the action of clicking using the hand gesture and the application closes.**

Figure 5.1: Helper Awaiting Awakening Order The screenshot shows the assistant interface in a standby mode, prepared to take voice commands from the user to wake it up. A prompt or listening icon are examples of visual cues that the interface might show to indicate that it is ready. Figure 5.2: Helper Launching Chrome: This picture shows the exact moment the user commands the assistant to launch Google Chrome and it does so successfully. A message requesting confirmation or a visual cue indicating the browser is being launched may be displayed on the assistant interface. Figure 5.4: Text file containing the user's prompt This picture shows how the user's prompt can be saved as a text file. By storing input commands or queries in a text document for future use or record-keeping, the assistant system logs user interactions. Figure 5.5: The generated response to the request for a motivational quote. This picture shows the assistant's response to a user request for a motivational quote. To give context, the assistant interface might show the quote along with extra details like the author or source. Figure 5.6: Email Leave Request Prompt Generated: This picture shows the prompt that appears from the assistant when a user asks to write their boss an email requesting a leave of absence. The assistant interface might show a form or email template and ask the user to fill it out with the necessary information, like the reason for the leave and the dates they would like.

Figure 5.7: Gesture-Based Hand DetectionThe user's hand is identified in this image, suggesting that they are ready to perform motions to interact with the assistance system. To indicate that the hand has been successfully detected and is ready for gesture input, the interface may highlight or outline it. Figure 5.8: Hand Gestures Associated with Cursor Movement: This image shows how the user's hand gestures cause the cursor to move. The interface shows how to control the position and movement of the cursor on the screen by tracking hand movements in real-time. Cursor Moving to Close Button (Figure 5.9): This illustration shows the cursor moving towards the close button on an application window and indicating with hand gestures that it is time to close the application. To indicate the intended action, the interface might show visual feedback, like highlighting the close button. Figure 5.10: To close the application, click the hand gesture action. This picture shows what happens when you use your hands to click an application to close it. The interface might show the result of the click action, like the application window closing, to validate that the gesture-based command was successfully executed.

## 5. Conclusion

In this research, the design and implementation of a Static Voice-enabled Personal Assistant for PC, created with the Python programming language, are thoroughly examined in this paper. Compared to earlier generations, these voice-activated assistants are incredibly helpful in saving time and are especially beneficial for people with disabilities. With just a few voice commands, this assistant can perform a wide range of tasks assigned to it, such as sending messages to a user's mobile device, automating YouTube functions, and

retrieving information from Google and Wikipedia. This voice assistant streamlines user tasks like web searches by automating multiple services with a single command. The ultimate objective is to transform this project into a full-featured server assistant that can take over from routine server administration duties. With the support of the PyCharm community and based on open-source software modules, the project is still flexible enough to accommodate new developments. Because of its modular design, flexibility is ensured and new features can be added without interfering with already-existing functionalities.

OpenCV and other computer vision concepts are integrated into the model to enhance its functionality. The assistant can use advanced color variation techniques to isolate and manipulate colors by creating masks through OpenCV. Packages like ‘mouse' are also used to facilitate mouse movement, allowing for accurate control of the cursor based on the detected color coordinates. This integration expands accessibility to a wide range of applications while also improving the user experience.

To put it simply, OpenCV is essential in giving users access to a wide range of models that greatly enhance functionality and usability. The voice assistant becomes a potent tool for streamlining processes and increasing productivity across a range of industries by utilizing these technologies.

## 6. Future Scope

This voice-activated personal assistant project has a bright future ahead of it and many applications. First off, advances in machine learning algorithms and natural language processing (NLP) can improve the assistant's comprehension and contextual response to user commands. To create interactions that are more engaging and natural, this involves enhancing the accuracy of voice recognition, adding more language support, and implementing advanced dialogue management systems.

Furthermore, the assistant's capabilities can be expanded to include smart home automation, IoT device control, and personalized user experiences through integration with cutting-edge technologies like artificial intelligence (AI) and the Internet of Things (IoT). The assistant can manage smart appliances, automate repetitive tasks, and offer proactive support based on user preferences and environmental circumstances by establishing connections with IoT devices. Adding multi-modal interaction support for voice, gesture, and visual inputs to the assistant's capabilities can improve accessibility and usability for a wider range of user needs and preferences. In order to facilitate more natural and engaging interactions, this involves implementing computer vision techniques for object detection, gesture recognition, and facial recognition. It is also possible to integrate the assistant with voice-activated VR and AR platforms, which could lead to new opportunities for immersive experiences and hands-free interaction in virtual environments. With its seamless integration of digital and real-world interactions, this could completely transform the gaming, education, training, and entertainment sectors.

Finally, by encouraging the development of third-party plugins, extensions, and integrations for particular use cases and domains, ongoing community-driven development and collaboration can further enhance the assistant's ecosystem. In light of changing user needs and technological advancements, this collaborative approach guarantees that the assistant will remain relevant, adaptive, and customizable. This will position it as a flexible and indispensable tool in the digital landscape of the future

## Refrences

[1] Shaughnessy, IEEE, “Interacting with Computers by Voice: Automatic Speech Recognition and Synthesis” proceedings of the IEEE, vol. 91, no. 9, september 2003.
[2] Patrick Nguyen, Georg Heigold, Geoffrey Zweig, “Speech Recognition with Flat Direct Models”, IEEE Journal of Selected Topics in Signal Processing, 2010.
[3] Mackworth (2019-2020), Python code for voice assistant: Foundations of Computational Agents- David L. Poole and Alan K. Mackworth.
[4] Keerthana S, Meghana H, Priyanka K, Sahana V. Rao, Ashwini B “Smart Home Using Internet of Things ”, proceedings of Perspectives in Communication , Embedded -systems and signal processing, 2017.

[5] Sutar Shekhar, P. Sameer, Kamad Neha, Prof. Devkate Laxman, "An Intelligent Voice Assistant Using Android Platform", IJARCSMS, ISSN: 232-7782, 2017.

[6] Rishabh Shah, Siddhant Lahoti, Prof. Lavanya. K, "An Intelligent Chatbot using Natural Language Processing", International Journal of Engineering Research , Vol.6 , pp.281-286, 2017.

[7] Ravivanshikumar ,Sangpal,Tanvee ,Gawand,SahilVaykar, "JARVIS: An interpretation of AIML with integration of gTTS and Python", proceedings of the 2019 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT), Kanpur, 2019.

[8] Luis Javier Rodríguez-Fuentes, Mikel Peñagarikano, AparoVarona, Germán Bordel, "GTTS-EHU Systems for the Albayzin 2018 Search on Speech Evaluation", proceedings of berSPEECH, Barcelona, Spain, 2018.

[9] Guoli Wang, (2010). Optical Mouse Sensor-Based Laser Spot Tracking for HCI Input, Proceedings of the 2015 Chinese Intelligent Systems Conference: Volume 2, pp.329-340.

[10] Anna De Liddo, Ágnes Sándor, et.al, (2012). Contested Collective Intelligence: Rationale, Technologies, and a Human-Machine Annotation. Computer Supported Cooperative Work (CSCW) Volume 21, Issue 4–5, pp 417–448.

[11] Rashmi Adatkar, Ronak Joshi, et.al, (2017). Virtual Mouse, Imperial Journal of Interdisciplinary Research (IJIR), Vol-3, Issue-4.

[12] Arul. V. H, Dr. Ramalatha Marimuthu, (2014). A Study on Speech Recognition Technology, Journal of Computing Technologies, Volume 3 Issue 7, pp 2278 – 3814.

[13] Aniwat Juhong, T. Treebupachatsakul, et.al, (2018). Smart eye-tracking system. 2018 International Workshop on Advanced Image Technology (IWAIT).

[14] Guojen Wen, Zhiwei Tong, et.al, (2009), Man machine interaction in machining center. International workshop on intelligent systems and applications. pp 1-4.