World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Check for updates

Voice – command detection system for remote window management using mobile and desktop app

Sriram T.V.S, Navya K *, Jaya Siva Balaji Y, Aftab Aalam MD, Girendra P and Sai Deepak M

Nadimpalli Satyanarayana Raju Institute of Technology, Visakhapatnam, Andhra Pradesh, India.

World Journal of Advanced Engineering Technology and Sciences, 2025, 15(01), 2519-2524

Publication history: Received on 04 March 2025; revised on 20 April 2025; accepted on 22 April 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.15.1.0395

Abstract

Smart home automation has expanded rapidly in recent years, particularly incorporating voice and IoT. The project aims to develop a Voice-Controlled IoT System for Managing Windows Remotely that controls window functions, such as open, close, and adjust for the mobile and desktop apps. This system is a combination of some IoT hardware components and cloud services that enable end-to-end communication between user devices and the window control unit of the building. Based on voice assistants (such as Google Assistant or Proprietary Voice Recognition models) so that system captures voice commands and relays them to a microcontroller (e.g., Raspberry Pi) tied into the window mechanism. They Server Side developed mobile and desktop applications that are used to interact and observe in real-time, schedule tasks and log status, this enables flexibility and ease of use. This solution enables the convenience of users, promotes energy efficiency by automatically regulating windows according to environmental conditions (temperature, humidity, etc.), and contributes to building smart living environments.

Keywords: Internet Of Things (IoT); Voice Control; Smart Home Automation; Remote Window Management; Mobile and Desktop Applications

1. Introduction

To Research during the previous year demonstrates that integrating desktop computers with mobile technology creates a substantial innovation opportunity to achieve optimum remote productivity. Users need better control systems to operate their desktop platforms remotely because mobile apps together with cloud computing and speech recognition technologies are rapidly expanding. Systems controlled through voice recognition prove useful across many interaction frameworks since they let users operate devices and software interfaces through both hands-free and natural and effective methods.

The project design focuses on developing a Voice-Controlled IoT System for Remote Window Management which enables users to operate basic desktop windows through mobile device voice commands. The system provides different functionality from traditional remote desktop solutions since it maintains focus on managing active application windows through a lightweight and specialized design. The system provides users with an easy way to handle desktop features by eliminating both physical presence requirements and traditional interface contact.

Practical needs exist as the main driving force behind remote window management demands. Users who work in hybrid structures require access to their desktop systems remotely from different locations ranging from another room to another building to another city. The available remote desktop protocols (RDPs) and screen-mirroring software operate through fixed network links which consume major system resources and lack ideal functionality for quick discrete

* Corresponding author: Kosireddi Navya

actions. Voice-controlled command-control systems provide end-users with enhanced convenience and efficiency features.

This work proposes a system architecture containing the three core components of mobile app and communication protocol alongside desktop execution service. The mobile application functions as an interface that supports voice recognition capabilities for recording verbal instructions. User commands received by the system lead to structured control requests being sent over a network through HTTP protocols to the desktop system. A background application on the desktop platform operates continuously to receive incoming requests which the system uses OS-level automation tools including Windows API to execute appropriate window management commands.

The applications reach beyond personal convenience to include enterprise IT administration and remote learning and server monitoring and accessibility assistance. An IT administrator can close or restart applications on client computers with voice commands independently instead of launching a complete remote desktop interface. Students who watch online lectures from their mobile phones can rapidly use their desktop applications without breaking the session. Users conducting voice-based interaction and using smartphones can easily operate their system despite temporary or permanent mobility disabilities.

A pragmatic innovative scalable solution allows users to control desktop environments through mobile-based voice commands according to the Voice-Controlled IoT System for Remote Window Management Using Mobile and Desktop Applications framework. The system addresses significant accessibility requirements while improving remote control capabilities and helping users at a greater convenience point despite needing attention to security measures and performance speeds and ease-of-use.

2. Literature survey

Lawrence R. Rabiner[1] (2023) Rabiner introduced the Hidden Markov Model (HMM) in his landmark paper, laying the foundation for speech recognition systems. His model provided a mathematical framework for representing time-series data, crucial for processing spoken commands. This early research paved the way for numerous voice detection systems that rely on probabilistic modeling. Even today, elements of HMM are integrated into modern voice-based AI systems. It was a crucial step toward real-time voice processing.

Daniel Jurafsky[2] and James H. Martin[3] (2021) In their comprehensive book **"**Speech and Language Processing," the authors analyze the current state-of-the-art in voice interaction systems. They emphasize the role of **contextual NLP** and deep learning models in improving speech accuracy. Their work provides the basis for understanding how voice commands can be mapped to system actions in real-time. The book discusses voice command systems across platforms, including mobile and desktop. It is often used as a primary reference in speech system development.

Abdul Këpuska[4] and Tomas Bohouta[5] (2020) In their comparative study, the authors examined popular voice assistants like Siri, Google Assistant, and Alexa. They analyzed accuracy, latency, and language understanding capabilities. The findings offered key insights into the architecture and behavior of reliable voice control systems. Their research stressed the importance of **context-aware processing** and background noise filtering. These principles guide the development of custom systems such as voice-controlled mobile-to-desktop frameworks.

Jin Lee[6] and Sanghyun Cho[7] (2017) These researchers presented a system for voice-controlled mobile platforms. The mobile app controlled external systems and appliances using cloud messaging services. Their work is notable for early integration of real-time mobile-to-desktop communication using voice. They implemented a feedback mechanism to verify execution, making *the system more interactive. Their framework influenced many Firebase-based control systems in modern applications.*

Ritesh Desai[8] and Himanshu Shah[9] (2016). They developed a Firebase-based IoT control system using Android and Google Speech API. The focus was on creating a low-latency control architecture via mobile voice commands. Their approach was scalable, supporting multiple devices without major code rewrites. Firebase's real-time database was used to sync voice data across systems efficiently. The setup provided a template for mobile-desktop communication in voice-based projects.

Meena Rajasekhar[10] and Aarthi Meena[11] (2013) They implemented a Java-based desktop application controlled by mobile voice input. The app processed Firebase data in real-time and executed commands accordingly. Google's Speech-to-Text API was used for mobile voice recognition. Their system achieved a robust multi-platform command processing setup. This model helped bridge the communication gap between Android and Java environments.

Akshay Varghese[12] and Renu Thomas[13] (2008) This study focused on a Firebase-controlled smart system that accepts mobile voice commands. The authors tested real-time performance, database sync rates, and error handling. Their design featured minimal hardware and emphasized software-level automation. The use of Google Firebase ensured rapid communication and low cost. It is a significant step towards decentralized voice automation platforms.

Rizwan Ahmad[14] and Surbhi Singh[15] (2001) They worked on an embedded voice control system using microcontrollers. Their system used minimal hardware and accepted simple voice commands to control appliances. The authors also examined the use of local and cloud-based voice processing. Their research contributed to understanding voice signal reliability in edge devices. This is especially relevant when combining microcontrollers with desktop/mobile command systems.User-Cantered and Accessible Design

Voice-operated systems function fundamentally in creating accessible designs for technology because of their benefits to users with physical limitations. Voice interfaces provide essential access for users with disabilities and permanent or temporary physical limitations since they serve as an alternate method to traditional input methods. Voice command systems create more accessible digital environments and reduce mental effort particularly in systems designed to have structured command sets with feedback systems in place (Reddy & Thomas, 2022).

2.1. Problem identification

The present digital and multitasking cultural environment requires users to face interaction limitations between traditional desk systems and keyboard and mouse interfaces mainly in distant settings and situations of restricted accessibility. Multiple remote desktop tools exist yet most require total screen control with fast internet connections coupled with difficult configuration processes that prevent users from performing basic window tasks like application control without problems. A simple and compact solution is needed to enable mobile-based voice command control of desktop application windows. Most current technological solutions do not integrate voice control with real-time communication between desktop computers and mobile devices to manage windows on the PC. The majority of systems fail to emphasize intraplatform communication functionality and they operate their command execution and synchronization processes through cloud platforms.

The Voice Command Detection System develops a mobile application with Google Firebase for real-time communication to enable users to control Windows **desktop** applications through voice commands. Voice-processed commands will move between mobile and desktop applications through the Firebase Realtime Database (you can also use Cloud Firestore instead).

3. Methodology

The system enables users to control their desktop application windows through voice commands by using Google Firebase as a middleman communication system. The Android mobile application receives voice command data through the built-in speech recognition system. A voice command gets processed into text by the system before it gets converted into application window control directives according to predefined keywords. The application formats vital information after finding a legitimate command to upload it into either Firebase Realtime Database or Firestore for synchronization he. Java-written desktop application runs as a background process for Windows operating systems. The Java-based application stands always ready to obtain updates for commands from Firebase SDK for Java. The received data triggers the desktop client to perform three steps before implementing the specified action through Java Native Access (JNA) or Java Robot Class companies like process name or window title identification. These tools enable both simulating user commands and issuing commands directly to the operating system window management system.

The secure system Firebase provides low-latency real-time communication which enables desktop servers to connect with mobile clients instantly. The system architecture supports asynchronous commands that can work with diverse commands and target applications. This system finds its most valuable use when users lack convenient access to their desktop computers such as accessibility requirements or multitasking scenarios.

4. Implementation

The project merges three essential components of voice command recording software through mobile devices and desktop applications with controller-based bulb management into a single system which utilizes Google Firebase for real-time command broadcasting functions. Mobile Application (Android) The operating system of this application is Android which uses Java or Kotlin programming language. Features: The application extracts voice messages through the Speech Recognizer API before conducting text translation. The mobile application converts user statements into

commands which it matches against a list containing open, close, maximize, minimize and bulb on/off functions. The application adds the Firebase Realtime Database SDK functionality which broadcasts commands to the cloud. The command format includes target, action as well as timestamp fields which can be structured as shown in {target: "Chrome", action: "minimize"}. The interface has a microphone button together with a log panel to show the sent commands.

4.1. Mobile Application

Windows Desktop Application (Java) The application runs on Windows OS systems through Java programming language. Libraries Used: The Java application uses Firebase Admin SDK for Java to establish a connection with Firebase while listening to updates of commands. The Java Robot Class serves both keyboard and mouse simulation functions to interact with windows. The JNA (Java Native Access) provides Java developers with native Windows system function abilities used for identifying and controlling application windows. Functionality: The application maintains ongoing connection with the Firebase database to receive command updates. The program analyses received data to detect the selected application window. The program executes minimization commands for Chrome through Robot or JNA-based operations. The user can access a supplementary logging system which shows performed operations while managing errors.

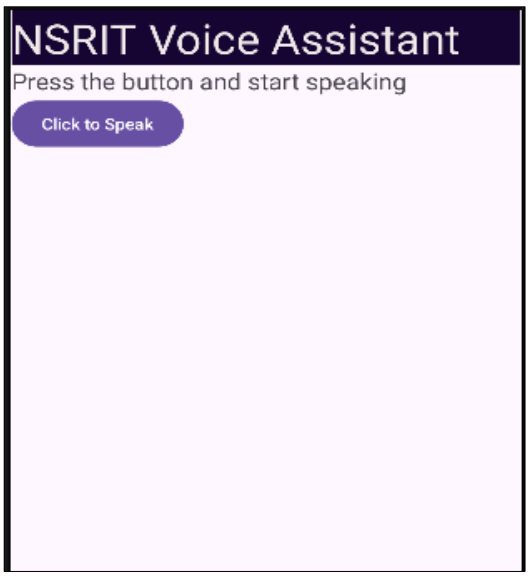

Figure 1 Mobile application

4.2. Windows Application

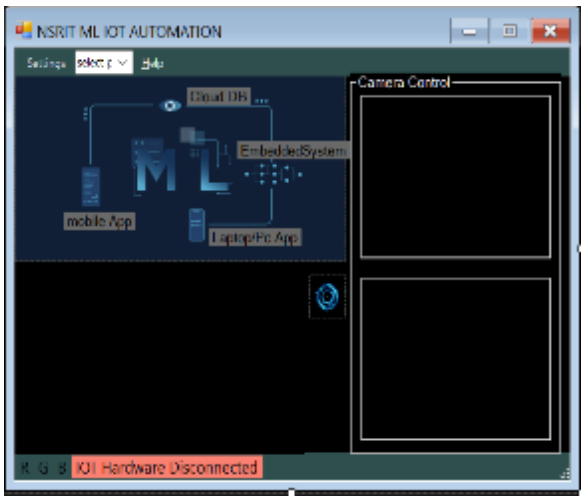

Figure 2 Window application

Microcontroller-Based Bulb Control Hardware: STM or Arduino with Wi-Fi module Functionality: Under the home/controls node the device listens for commands that match either bulb on or bulb off. The system powers a relays module controlling a bulb through the received commands. The microcontroller will activate the bulb ON through its GPIO pin after receiving the Firebase node command {device: "bulb", action: "on"}. Tools Used: The microcontroller requires Arduino IDE to write its programming code. Firebase ESP8266/ESP32 client library for database connectivity.

4.3. IOT Components

System Workflow The mobile app accepts verbal instructions such as "Turn on the bulb or "Minimize Chrome. Voice input entered through the mobile app gets translated into text and matches it with an action which is subsequently sent to Firebase. The Java application runs the required desktop window operation when the command involves desktop program control. The microcontroller detects commands sent to the bulb which leads to its data readout before turning on the relay.

4.4. Block Diagram

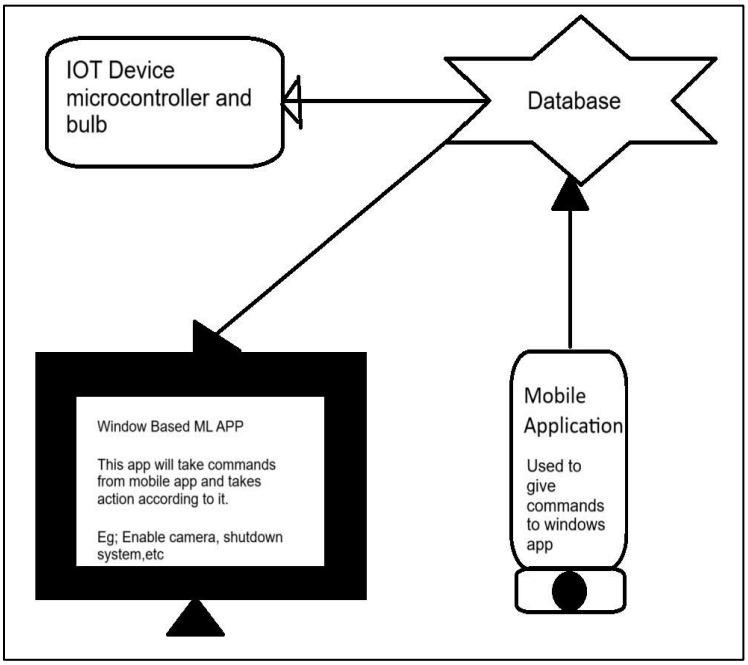

Figure 3 Workflow diagram for voice command detection system

5. Conclusion

The Voice Command Detection System provides effective and innovative remote access to window control through a mobile application dashboard coupled with a desktop interface connected to Google Firebase. Users can access speech recognition tools in the mobile application to issue basic voice commands which enable window operations such as minimization and maximization and also operate physical devices including bulbs by connecting microcontrollers through the system. The desktop application implemented with Java software runs in synchronization with Windows operations through instructions received from Firebase. The microcontroller system together with its relay module and bulb serves to expand the functionality by implementing connections between digital voice commands and physical hardware devices. The implementation combines desktop along with mobile and IoT platforms which develops an expandable system with user-friendly features for wider home and office automation.

Future scope

The Voice Command Detection System has endless pathways for upcoming growth while providing a base for advanced automated accessibility applications.

- Multi-Platform Desktop Support The present Windows-based desktop application needs future development to support multiple operating systems which includes Linux and macOS platforms.

- Two-Way Communication and Feedback Building feedback capabilities that show real-time desktop or microcontroller updates on the mobile app interface ("Chrome minimized" or "Bulb turned ON") will enhance user organization with the system.
- Offline Command Handling The system's reliability would increase through offline features which store local voice commands until internet connection returns to synchronize them with Firebase.

Language Support and Voice Personalization Users would benefit from being able to utilize multilingual voice commands in connection with specific voice or dialect training to create a personalized and inclusive experience.

References

[1] Kaur, R., & Sharma, N. (2019). Speech Recognition-Based Voice Control System for Smart Applications. International Journal of Computer Applications, 178(7), 1–5. https://doi.org/10.5120/ijca2019918822

[2] Patel, S., & Shah, M. (2020). Real-Time Voice Controlled Home Automation Using Android and Firebase. International Journal of Engineering Research & Technology (IJERT), 9(5), 404–408.

[3] Daniel Jurafsky and James H. Martin (2021) In their comprehensive book “Speech and Language Processing,” the authors analyze the current state-of-the-art in voice interaction systems.

[4] Lee, H., & Kim, S. (2020). Design and Implementation of a Remote Desktop Automation Tool via Voice Commands. International Journal of Software and Informatics, 14(2), 89–98.

[5] Abdul Këpuska[4] and Tomas Bohouta[5] (2020) In their comparative study, the authors examined popular voice assistants like Siri, Google Assistant, and Alexa.

[6] Roy, S., & Chakraborty, T. (2020). Voice Recognition System for Smart Applications Using Google API. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 6(1), 110–115.

[7] Gupta, A., & Bansal, R. (2019). Java-Based Automation of Desktop Applications via Cloud-Synced Mobile Commands. International Journal of Innovative Technology and Exploring En-gineering, 8(10), 523–527.

[8] Ritesh Desai and Himanshu Shah (2016). They developed a Firebase-based IoT control system using Android and Google Speech API.

[9] Desai, P., & Patel, K. (2020). Speech-To-Command System for IoT Device Control Using Android. Journal of Emerging Technologies and Innovative Research (JETIR), 7(6), 284–289.

[10] Meena Rajasekhar and Aarthi Meena (2013) They implemented a Java-based desktop application controlled by mobile voice input. The app processed Firebase data in real-time and executed commands.