SJST
sjst.scst.edu.ly

Surman Journal for Science and Technology
ISSN: Online (2790-5721) - Print (2790-5713)

مجلة صرمان للعلوم والتقنية
Vol 7, No.2. Jun – Dec. 2025
Pages: 392 ~ 400

DESIGN AND IMPLEMENTATION OF A VOICE-CONTROL SYSTEM FOR COMPUTER APPLICATIONS USING PYTHON

Eenas Suwayd[1] , Mahmuod saad[2]

Surman College of Since and Technology, Surman, Libya.
*Corresponding author, e-mail: suwayd.e@scst.edu.ly

الملخص

أصبحت أنظمة التحكم الصوتي أحد المكونات الأساسية للتفاعل الحديث بين الإنسان والحاسوب، حيث تتيح للمستخدمين تشغيل الحواسيب والأجهزة الإلكترونية من خلال الأوامر الصوتية بالاعتماد على تقنيات التعرف على الكلام ومعالجة اللغة الطبيعية. تهدف هذه الدراسة إلى تصميم وتنفيذ نظام للتحكم بالحاسوب باستخدام الأوامر الصوتية اعتمادًا على لغة البرمجة بايثون. وقد تم استخدام عدد من مكتبات معالجة الكلام، بما في ذلك SpeechRecognition وPyAudio وPyWhatKit، لتطوير نموذج عملي قادر على استقبال الأوامر الصوتية وتنفيذ مهام محددة، مثل البحث وتشغيل المحتوى متعدد الوسائط على منصة يوتيوب.

أظهرت النتائج أن التطبيقات المعتمدة على الأوامر الصوتية يمكن أن تسهم في تحسين الإنتاجية، وتقليل الجهد البشري، وخفض تكاليف التشغيل، بالإضافة إلى توفير وسيلة أكثر سهولة وراحة للتفاعل مع الحواسيب. كما أبرزت الدراسة مرونة وبساطة لغة البرمجة بايثون في تطوير تطبيقات التحكم الصوتي، وأوصت بتوسيع نطاق استخدام هذه الأنظمة لدعم الأشخاص ذوي الإعاقة وتطوير تطبيقات ذكية أكثر تقدمًا.

الكلمات المفتاحية: نظام التحكم الصوتي، التعرف على الكلام، بايثون، التفاعل بين الإنسان والحاسوب، الذكاء الاصطناعي، معالجة اللغة الطبيعية.

Abstract

Voice control systems have become an essential component of modern human–computer interaction. These systems enable users to operate computers and electronic devices through spoken commands using speech recognition and natural language processing technologies. This study investigates the design and implementation of a voice-controlled computer system using Python programming language. Several speech-processing libraries, including SpeechRecognition, PyAudio, and PyWhatKit, were utilized to develop a practical prototype capable of receiving voice commands and executing specific tasks such as searching and playing multimedia content on YouTube. The results demonstrate that voice-based applications can improve productivity, reduce human effort, minimize operational costs, and provide more convenient interaction with computers. The study also highlights the flexibility and simplicity of Python for developing speech-controlled applications and recommends extending such systems to support people with disabilities and more advanced intelligent applications.

Keywords: *Voice Control System, Speech Recognition, Python, Human–Computer Interaction, Artificial Intelligence, Natural Language Processing.*

Submitted: *23/11/2025* ***Accepted****: 30/12/2025*

1. Introduction

Recent advances in artificial intelligence (AI), machine learning, and speech processing technologies have transformed the way humans interact with computers and electronic devices. Human computer interaction (HCI) has evolved from traditional input methods, such as keyboards and mice, toward more natural and intuitive interfaces. Among these technologies, voice control systems have emerged as one of the most promising approaches for enabling hands-free and user-

friendly interaction with computers and smart devices. Voice-controlled systems utilize speech recognition and natural language processing (NLP) techniques to convert spoken words into machine-understandable commands [1]. These technologies allow users to perform various tasks, including opening applications, controlling multimedia content, searching the Internet, sending messages, and operating smart home devices through voice commands. As a result, voice interaction has become an essential component of modern intelligent systems and digital assistants. The rapid growth of Industry and the Internet of Things (IoT) has accelerated the development and deployment of voice-based technologies. Intelligent voice assistants such as Apple Siri, Google Assistant, Amazon Alexa, and Microsoft Cortana have demonstrated the practical importance of speech-based interaction in everyday life. These systems provide users with convenient access to information and services while reducing the need for physical interaction with devices. Moreover, voice control technologies play an increasingly important role in healthcare, education, robotics, automotive systems, and assistive technologies for individuals with disabilities [2].

Speech recognition systems have benefited significantly from recent advances in machine learning and deep learning algorithms, which have improved recognition accuracy and system performance. Convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer-based architectures have enhanced the ability of speech recognition systems to understand natural language and distinguish between different speakers and accents. Consequently, voice-controlled applications have become more reliable and widely adopted across various domains [4]. Python has become one of the most popular programming languages for developing intelligent applications because of its simplicity, readability, and extensive ecosystem of libraries. Python provides powerful tools for speech recognition, natural language processing, machine learning, and multimedia applications. Libraries such as SpeechRecognition, PyAudio, PyWhatKit, TensorFlow, and PyTorch enable developers to build sophisticated voice-controlled systems with relatively simple programming structures. Furthermore, Python's portability and cross-platform compatibility make it suitable for implementing speech-based applications on different operating systems [5].

Despite the remarkable progress achieved in voice recognition technologies, several challenges remain, including environmental noise, variations in pronunciation, language diversity, and recognition accuracy. Therefore, continuous research and development are required to enhance system robustness and improve user experience. Developing lightweight and efficient voice-control applications represents an important research area with significant practical implications. This study aims to investigate the design and implementation of a voice-controlled computer system using Python programming language. The proposed system employs speech recognition techniques to capture spoken commands and execute specific tasks automatically. A practical prototype is developed using several Python libraries, including SpeechRecognition, PyAudio, and PyWhatKit. The system is capable of recognizing voice commands and performing multimedia-related tasks such as searching and playing videos on YouTube. The significance of this study lies in demonstrating the feasibility of implementing intelligent voice-control applications using open-

source tools and Python libraries. In addition, the proposed approach provides a simple and cost-effective framework that can be extended to support more advanced applications, including smart home automation, robotic systems, and assistive technologies for people with disabilities. The outcomes of this research contribute to enhancing human–computer interaction and highlight the potential of speech recognition technologies in modern computing environments. Although traditional input devices are widely used, they may not provide convenient interaction in many situations, especially for users with disabilities or in environments requiring hands-free operation. Consequently, there is a need for efficient voice-control systems capable of understanding spoken commands and executing computer tasks effectively. The objectives of this study are:

1. To investigate the principles of voice-control systems.
2. To highlight the role of artificial intelligence and speech recognition technologies.
3. To develop a practical voice-control application using Python.
4. To evaluate the performance and advantages of speech-based interaction.

2. Literature Survey

Voice recognition and voice-controlled systems have attracted considerable attention in recent years due to their growing importance in human–computer interaction and intelligent applications. Numerous studies have investigated speech recognition technologies, machine learning approaches, and practical implementations of voice-based systems.

Lu and Hu (2013) developed a speech control system for robotic applications using Raspberry Pi technology. Their work demonstrated the feasibility of employing speech recognition techniques for controlling robotic movements and executing commands through voice input. The results indicated that speech interfaces could provide a convenient and effective means of interaction with embedded systems [1]. Parthasarathy et al. (2018) proposed a voice-enabled code editor that allows programmers to write source code using speech recognition. The system was designed to improve accessibility and productivity, especially for individuals with physical disabilities. Their study showed that voice recognition can be integrated into software development environments with acceptable accuracy and performance [5]. Ali (2020) investigated the applications of Python programming language in Geographic Information Systems (GIS) and ArcMap environments. The study emphasized Python's flexibility and its extensive library support, making it suitable for implementing intelligent applications, including speech-processing systems [6].

Rouina (2023) introduced a voice-guided electronic translation system based on Python programming. The developed system translated spoken English commands into Arabic and highlighted the capability of Python libraries in handling speech recognition and natural language processing tasks. The study demonstrated the practical applicability of speech-based technologies in multilingual environments [7]. Sartiukova et al. (2023) proposed a remote computer voice-control system based on convolutional neural networks (CNNs). Their work employed deep learning techniques to improve recognition accuracy and reduce noise sensitivity. Experimental results showed that CNN-based architectures significantly enhance the performance of speech

recognition systems compared with conventional approaches [8]. Hinton et al. (2012) presented deep neural networks (DNNs) for acoustic modeling in speech recognition. Their pioneering work demonstrated substantial improvements in recognition accuracy and established deep learning as a dominant approach in modern speech-processing applications [9]. Graves et al. (2013) investigated recurrent neural networks (RNNs) for speech recognition tasks. Their study demonstrated that recurrent architectures are capable of learning temporal dependencies in speech signals and achieving high recognition performance, particularly for continuous speech processing [10].

Amodei et al. (2016) introduced Deep Speech, an end-to-end speech recognition system based on deep learning techniques. The proposed architecture provided high accuracy and robustness against environmental noise, making it suitable for practical applications involving voice-controlled interfaces [11]. Vaswani et al. (2017) proposed the Transformer architecture, which revolutionized natural language processing and speech-related tasks. The self-attention mechanism introduced in their work significantly improved sequence modeling and laid the foundation for many modern speech-recognition systems [12]. Radford et al. (2023) developed Whisper, a large-scale speech recognition model trained on extensive multilingual datasets. The model achieved remarkable robustness and accuracy across different languages and acoustic conditions, representing one of the most advanced speech recognition systems currently available [13].

Overall, previous studies demonstrate that speech recognition technologies have evolved from conventional acoustic models to advanced deep learning architectures. Python has emerged as a preferred platform for implementing voice-controlled applications because of its simplicity, extensive libraries, and compatibility with machine learning frameworks. Nevertheless, there remains a need for lightweight and practical voice-control systems that can be easily implemented and customized for everyday computer applications. Therefore, the present study focuses on developing a Python-based voice control system capable of executing computer commands efficiently and providing a user-friendly interface.

3. Methodology

The proposed system was implemented using Python programming language. The following libraries were installed:

1. SpeechRecognition
2. PyAudio
3. PyWhatKit

The system workflow consists of:

1. Capturing voice input through a microphone.
2. Converting speech into text using Google Speech Recognition.
3. Processing and analyzing the command.
4. Executing the required task.
5. Opening YouTube and playing requested content.

The implementation procedure includes importing libraries, defining the microphone source, receiving speech signals, converting them to text, and executing commands based on predefined keywords.

4. Experimental Results

Development Environment Setup: The implementation of the proposed voice-control system was carried out using the Python programming language. Initially, the Python Integrated Development Environment (IDE) was launched, and a new project file named project1 was created. To ensure the proper operation of the voice-control application, several Python libraries were installed. These libraries provide functionalities for speech recognition, microphone interfacing, and automatic execution of multimedia commands. The required packages were installed using the Command Prompt (CMD) as follows:

- pip install SpeechRecognition
- pip install pipwin
- pipwin install pyaudio
- pip install pywhatkit

System Methodology and Architecture: The proposed voice-control system employs a modular, five-tier architecture to translate vocal inputs into automated system actions, as illustrated in Figure 1. Figure architectural framework of the proposed Python-based voice-control system showing structural data flow.

1. Hardware Interfacing and Signal Capture

The process begins at the User Audio Interface, where spoken commands are captured by a physical microphone. At the Hardware Interfacing Layer, the PyAudio library interfaces with the operating system sound drivers to sample and digitize the analog signal into pulse-code modulation (PCM) data streams. To counter environmental interference, an ambient noise calibration routine adjusts the energy threshold before processing.

2. Speech Recognition and Cloud Processing

The digitized audio is handled by the Speech Processing Layer via the SpeechRecognition API, which formats the raw buffer into standardized web requests. These requests are sent to the cloud-based Google Speech Engine within the Cloud Processing Layer. This engine utilizes deep learning language models to perform acoustic decoding, converting the audio stream into an editable text string and returning it asynchronously to the local script.

3. Command Logic and Action Execution

The returned text string is processed at the Application Logic Layer using Python tokenization to parse strings and isolate key trigger verbs (e.g., "play" or "search"). Once validated, control shifts to the Action Execution Layer, where the PyWhatKit Framework interacts with the system's default web browser. It formats the parsed parameters into an explicit target URL, resulting in the automated launch and playback of the requested video at the System Output node. Built-in

sjst.scst.edu.ly
Surman Journal for Science and Technology
ISSN: Online (2790-5721) - Print (2790-5713)
مجلة صرمان للعلوم والتقنية
Vol 7, No.2. Jun – Dec. 2025
Pages: 392 ~ 400

exception handling filters any network timeout (RequestError) or transcription failure (UnknownValueError) to ensure runtime stability.

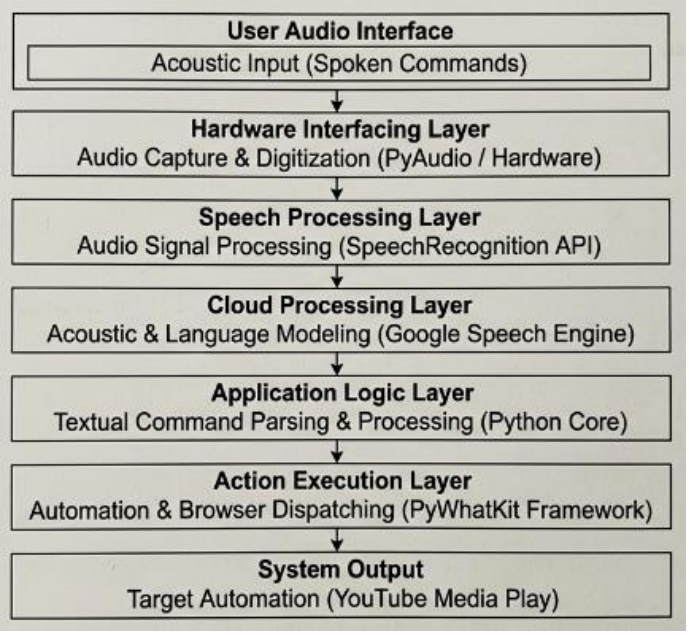

Figure 1. Architectural framework of the proposed Python-based voice-control system showing structural data flow.

The Command Prompt window used for package was installed, whereas Figure 2 illustrates the installation process of the SpeechRecognition library. Similar procedures were followed for installing the remaining libraries.

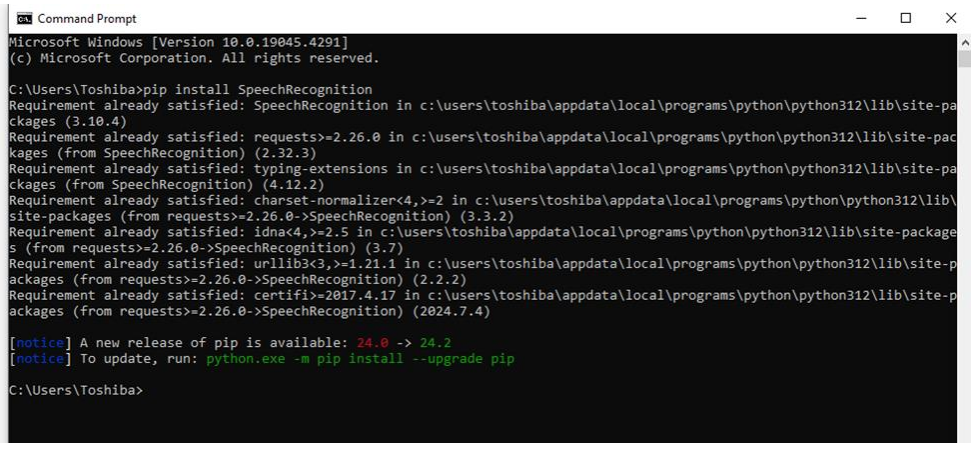

Figure 2. Installation of the SpeechRecognition package.

After successfully installing the required packages, the libraries were imported into the Python program to provide speech recognition and command execution functionalities.

sjst.scst.edu.ly
Surman Journal for Science and Technology
ISSN: Online (2790-5721) - Print (2790-5713)
مجلة صرمان للعلوم والتقنية
Vol 7, No.2. Jun – Dec. 2025
Pages: 392 ~ 400

Source Code Implementation: The program was developed to capture voice commands through the microphone and process them automatically. The following operations were performed:

- The SpeechRecognition library was initialized using the alias sr.
- The microphone source was defined and configured.
- The system was instructed to listen to the user's speech input.
- A variable named voice was used to capture the audio signal.
- The captured speech was converted into text using Google's speech recognition service.
- All recognized words were converted into lowercase letters for efficient processing.
- The recognized command was displayed on the console.

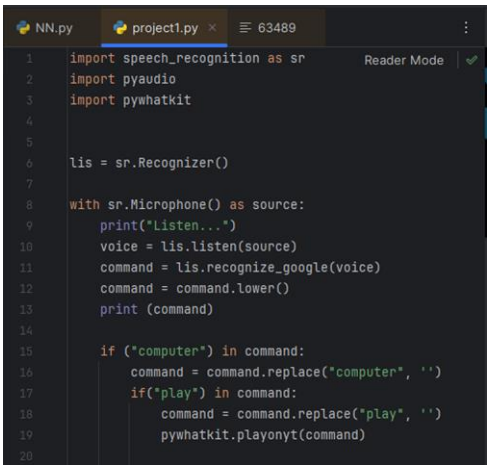

Figure 3. Source code of the voice recognition program.

To enhance command processing, additional instructions were included in the program. If the keyword computer was detected, it was removed from the command string. Similarly, when the keyword play was identified, it was excluded from the command, and the remaining text was used as the search query. Subsequently, the program automatically searched YouTube and played the requested content. Overall, the implementation results confirm the feasibility of developing voice-controlled applications using Python and demonstrate the effectiveness of integrating speech recognition technologies with multimedia automation tools.

5. Discussion

The experimental results demonstrate that the proposed voice-control system provides an effective and user-friendly approach for human–computer interaction. By integrating speech recognition technologies with Python libraries, the system successfully converts spoken commands into executable actions and performs multimedia-related tasks automatically. Several advantages of the developed system were observed during the implementation and testing phases. First, the system reduces the physical effort required for interacting with computers by providing hands-free operation. Second, it enhances productivity and efficiency by enabling users to execute

tasks quickly and conveniently. Furthermore, the use of voice commands minimizes operational costs and reduces the possibility of human errors associated with conventional input methods. Another significant advantage is the improvement in accessibility, particularly for elderly individuals and users with physical disabilities who may experience difficulties using traditional input devices. In addition, the proposed approach offers a convenient and intuitive interaction mechanism that contributes to enhancing the overall user experience.

From a software development perspective, Python proved to be an efficient platform for implementing voice-controlled applications due to its simplicity, portability, and extensive collection of libraries. Packages such as SpeechRecognition, PyAudio, and PyWhatKit significantly simplified the development process and enabled rapid prototyping of intelligent speech applications. Despite the promising results, several limitations were identified. The performance of the system depends on Internet connectivity because Google's speech recognition service is used for voice processing. In addition, environmental noise and pronunciation variations may affect recognition accuracy and increase response time. Therefore, future improvements may involve integrating offline speech recognition models and advanced deep-learning techniques to enhance robustness and reliability.

6. Conclusion

This study presented the design and implementation of a voice-controlled computer system using Python programming language. The developed system employs speech recognition techniques to capture spoken commands and translate them into executable actions. The implementation utilized several Python libraries, including SpeechRecognition, PyAudio, and PyWhatKit, to provide an efficient framework for voice-based interaction. Experimental results demonstrated that the proposed system is capable of recognizing voice commands and performing multimedia tasks successfully. The system exhibited satisfactory performance in terms of response time, ease of implementation, and usability. Moreover, the results confirmed that Python provides a flexible and powerful platform for developing speech-based applications because of its rich ecosystem of libraries and development tools. The proposed system contributes to improving human–computer interaction by enabling natural and hands-free communication with computers. In addition, it provides a low-cost and scalable solution that can be extended to support more advanced intelligent applications, including smart home automation, virtual assistants, robotic systems, and assistive technologies for individuals with disabilities. Overall, the findings of this study demonstrate the feasibility and effectiveness of Python-based voice-control systems and highlight their potential for future intelligent and interactive computing applications.

References

1. Lu, C. H., & Hu, Y. L. (2013). *Speech Control System for Robot Based on Raspberry Pi*. Advanced Materials Research, 791, 663–667.
2. Parthasarathy, G., Rangeesh, A., Kishowr, V. S., Sriram, R., & Vijay, S. (2018). *An Approach to Accept Voice in Code Editor through Speech Recognition*. International Journal of Engineering Research and Technology.

3. Rouina, A. (2023). *Voice-Guided Electronic Translator Based on Speech Commands Using Python*. ALTRALANG Journal, 5(3), 485–495.
4. Sartiukova, A., Markiv, O., Vysotska, V., Shakleina, I., Sokulska, N., & Romanets, I. (2023). *Remote Voice Control of Computer Based on Convolutional Neural Network*. IEEE IDAACS.
5. Parthasarathy, G., Rangeesh, A., Kishowr, V. S., Sriram, R., & Vijay, S. (2018). *An Approach to Accept Voice in Code Editor through Speech Recognition*. International Journal of Engineering Research and Technology.
6. Ali, A. R. (2020). *Python Programming in GIS Systems and ArcMap Environment*. Journal of Humanities and Sciences, 64, 1–14.
7. Rouina, A. (2023). *Voice-Guided Electronic Translator Based on Speech Commands Using Python*. ALTRALANG Journal, 5(3), 485–495.
8. Sartiukova, A., Markiv, O., Vysotska, V., Shakleina, N., Sokulska, N., & Romanets, I. (2023). *Remote Voice Control of Computer Based on Convolutional Neural Network*. IEEE IDAACS.
9. Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., et al. (2012). *Deep Neural Networks for Acoustic Modeling in Speech Recognition*. IEEE Signal Processing Magazine, 29(6), 82–97.
10. Graves, A., Mohamed, A., & Hinton, G. (2013). *Speech Recognition with Deep Recurrent Neural Networks*. IEEE ICASSP, 6645–6649.
11. Amodei, D., et al. (2016). *Deep Speech 2: End-to-End Speech Recognition in English and Mandarin*. Proceedings of ICML, 173–182.
12. Vaswani, A., et al. (2017). *Attention Is All You Need*. Advances in Neural Information Processing Systems (NeurIPS), 5998–6008.
13. Radford, A., et al. (2023). *Robust Speech Recognition via Large-Scale Weak Supervision*. Proceedings of ICML.