ARDUINO-SPOTIFY CONTROLLER

- Ayal Moran 311148522 - moranayal@gmail.com
- Daniel Aviv 209228154 - dani6laviv@gmail.com

INTRODUCTION

The system employs an Arduino UNO R3 microcontroller, four Ultrasonic sensors (PING)))) to interpret human gestures and relay appropriate commands to Spotify's API.

- The project seamlessly integrates hardware simulation on TinkerCAD and data scraping from the serial monitor of TinkerCAD using Python, which facilitates the interaction with the Spotify API.

The purpose of this project is to create a more interactive and accessible user experience for Spotify, allowing users to control their music playback through simple hand movements.

System Design

The system design of our project can be broadly divided into three primary components: hardware design, software implementation, and the interface between hardware and software.

Hardware Design:

The hardware setup is designed around an Arduino UNO R3 microcontroller, the heart of our system. The UNO R3 was chosen for its ease of use, extensive community support, and extensive libraries, making it ideal for this project.

Four Ultrasonic Sensors (PING))) are connected to the Arduino, serving as the system's eyes, detecting the presence and motion of hand gestures. These sensors use sonar to determine the distance to an object, providing robust and reliable gesture recognition. They are connected to digital pins 9, 10, 11, and 12 on the Arduino board for trigger signals. The echo signals are hard-wired to the corresponding trigger pins, reducing the number of pins required on the Arduino.

The entire hardware setup is simulated on TinkerCAD, a web-based 3D modeling and simulation tool. This allows for a hassle-free and iterative design process, minimizing the risk of hardware damage and allowing for easy modifications and troubleshooting.

Software Implementation:

The software for our system is written in C/C++ and is designed to run on the Arduino UNO R3. The code continuously monitors the four ultrasonic sensors for any changes indicating a gesture. It then interprets the sequence of triggered sensors to determine the corresponding gesture and performs the appropriate action, such as play, pause, next track, previous track, volume up, and volume down.

Hardware-Software Interface:

One of the unique aspects of our project is the interface between the hardware and the Spotify API. This is accomplished by scraping the serial monitor of TinkerCAD using Python. The Python script listens to the serial port for the commands sent by the Arduino program. Once a command is detected, the script sends the corresponding request to the Spotify API, effectively controlling the Spotify playback based on the detected hand gestures.

Implementation

The Arduino is programmed in C/C++.
The program continually reads data from the sensors, interprets the sequence of triggered sensors as different gestures, and performs corresponding actions.
If a sensor is triggered, it records the sensor number and checks if a command should be executed.
The commands include play, pause, next track, previous track, volume up, and volume down.
The system also includes a condition to reset the sensor data if no action is detected for 15 seconds.
The system employs a Python script, executed within a Jupyter Notebook environment, to methodically extract data from TinkerCAD's serial monitor. The extracted data, which represents specific user commands, is then accurately relayed to the Spotify API, resulting in the desired actions being performed on the Spotify platform.

Conclusion

Looking ahead, there are several avenues for further improvement and expansion of this project.
Some possible improvements may include:

- Gesture Speed Recognition:
  Implement functionality to recognize the speed of gestures, enabling additional commands. For instance, a quick swipe might indicate skipping several tracks, while a slower swipe could suggest skipping a single track.

- Machine Learning Integration:
  Incorporate a machine learning algorithm to enhance gesture recognition accuracy and versatility. This could also help the system adapt to individual user's gesture style and improve over time.

- Feedback Mechanisms:
  Add additional feedback mechanisms such as audio prompts or visual feedback to confirm the recognition of gestures and the successful execution of commands.

- User Customization:
  Provide a feature that allows users to define their own gestures and link them with specific commands. This level of personalization would make the system more flexible and tailored to individual user preferences.

In conclusion, this project successfully demonstrates the potential of gesture-based interfaces in enhancing the way we interact with digital platforms. We have developed an intuitive and engaging system to control Spotify using hand gestures.
As a study project, it stands as an example to the innovative possibilities that emerge when hardware, software, and user experience design converge.