PROJECT REPORT ON

SPEECH-CONTROLLED MUSIC PLAYER

Shubham Damkondwar

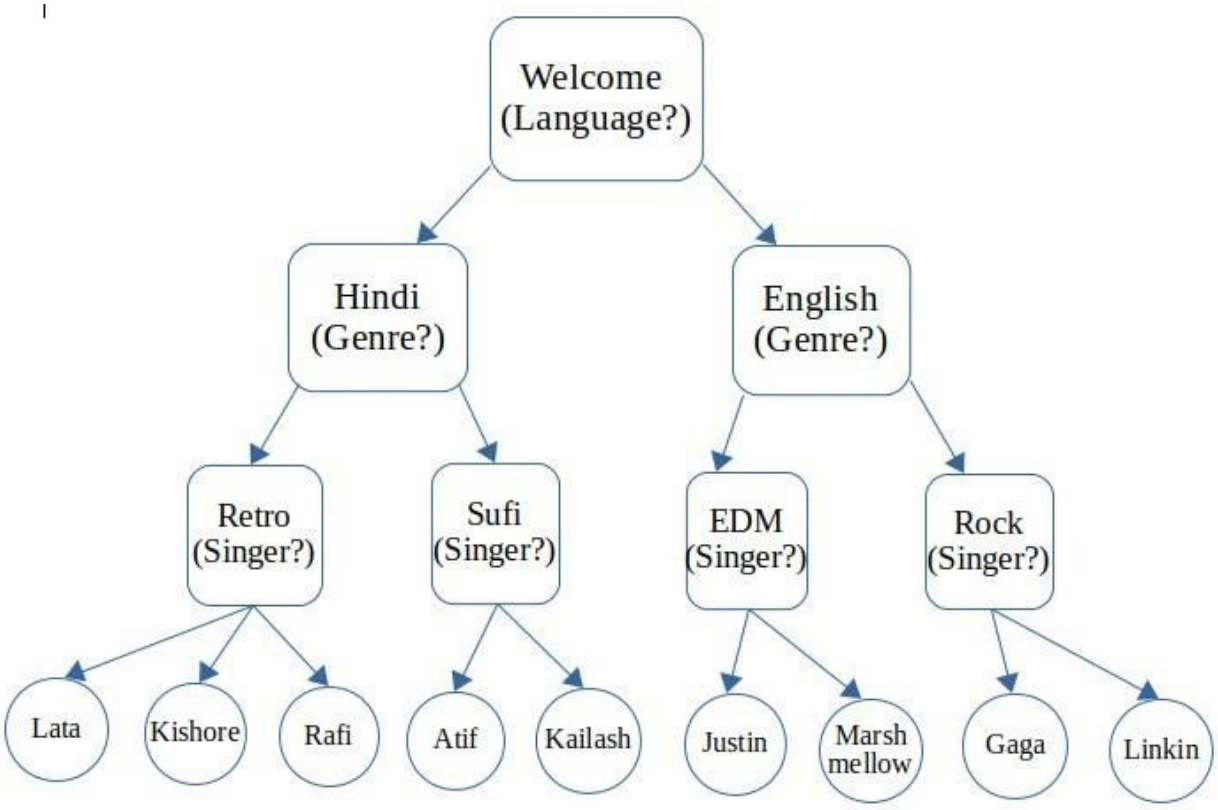
IIT Guwahati

Introduction

In this project, I have developed a “Speech Based Song Playlist application” for Desktop. This application provides users the functionality to play songs from various genres of the preferred language.
This project “Speech-based Music Player” is a Windows Forms based VC++ application that uses the SoundPlayer feature of Windows Forms to play a song of the language chosen by the user. The project records the choice of language (a word like “Hindi” or “English” spoken by the user, recognizes the word spoken, and plays a song of that language, on the basis of the recognized word. The recognition is done using HMM, and the project also provides speech based navigation around the GUI.

PROJECT EXECUTION

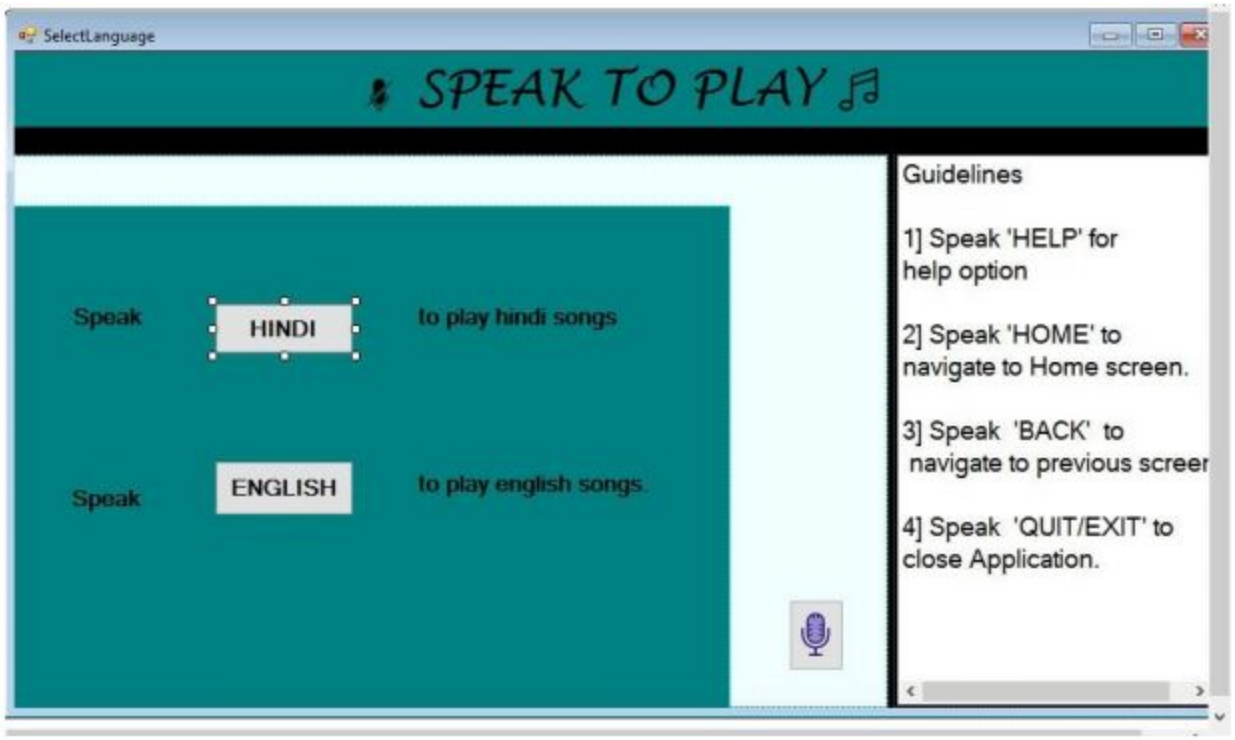
FLOWCHART



Frontend

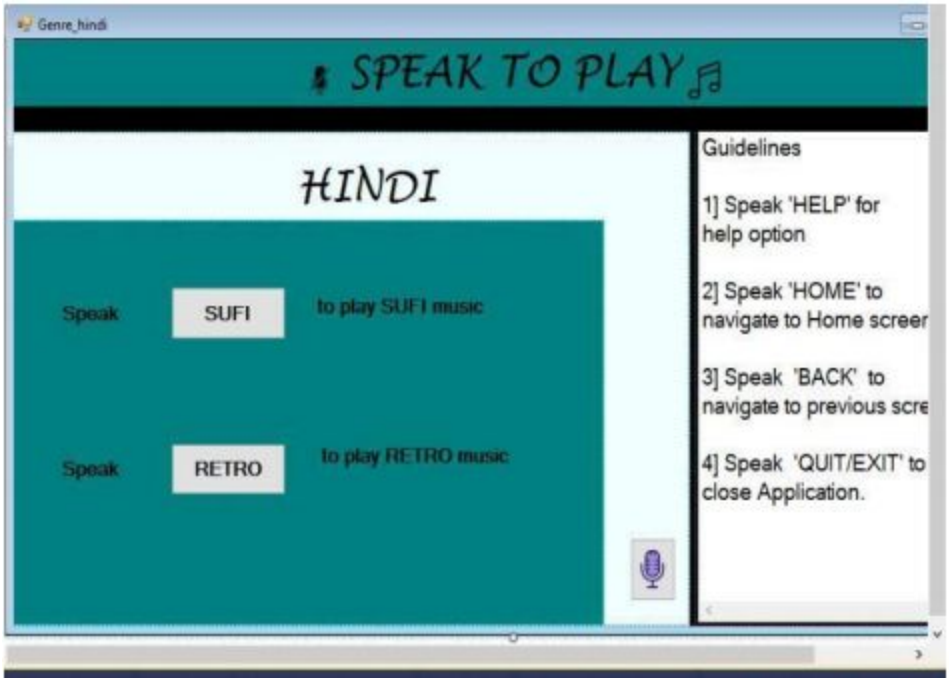
Frontend is developed in Visual Studio. There are 3 basic interfaces designed in the application.

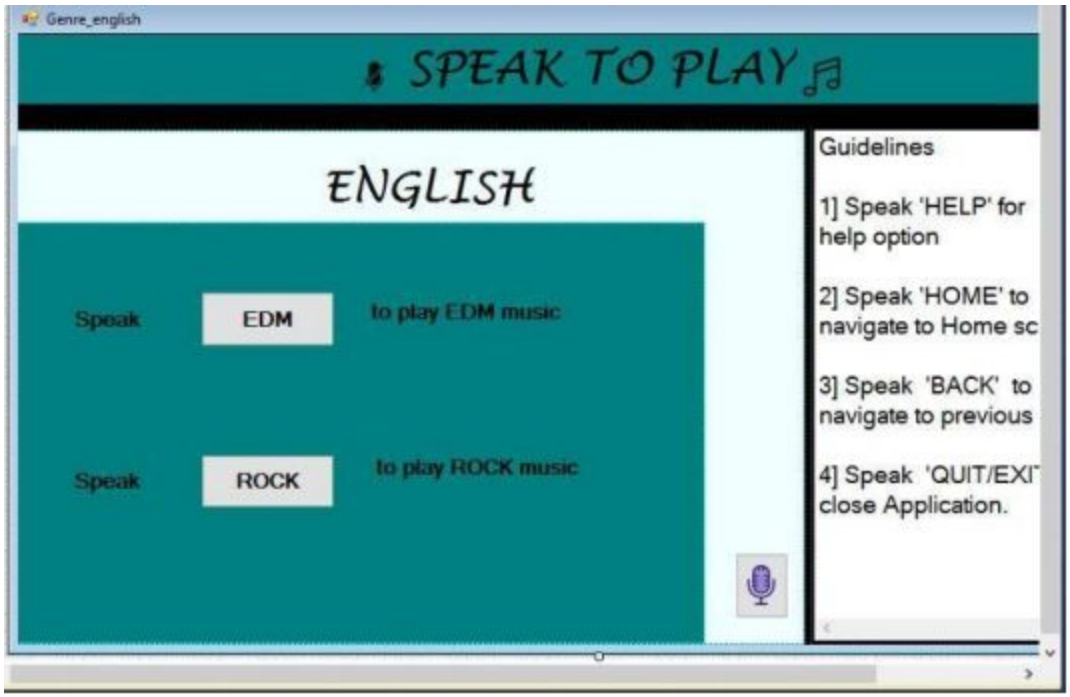
1. Languages: This interface will let the user to select between 2 languages. User can either select Hindi or English.



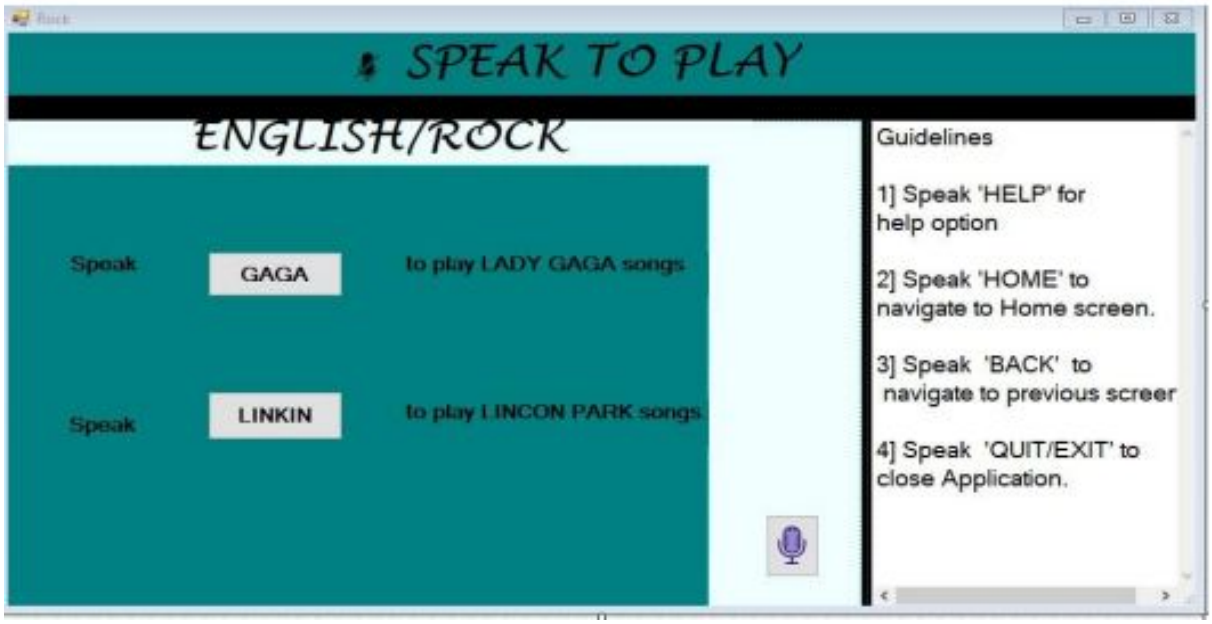
2. Genres : This interface will ask user to select among the 2 genres provided under each language.

   There are 2 options of genre under each category.

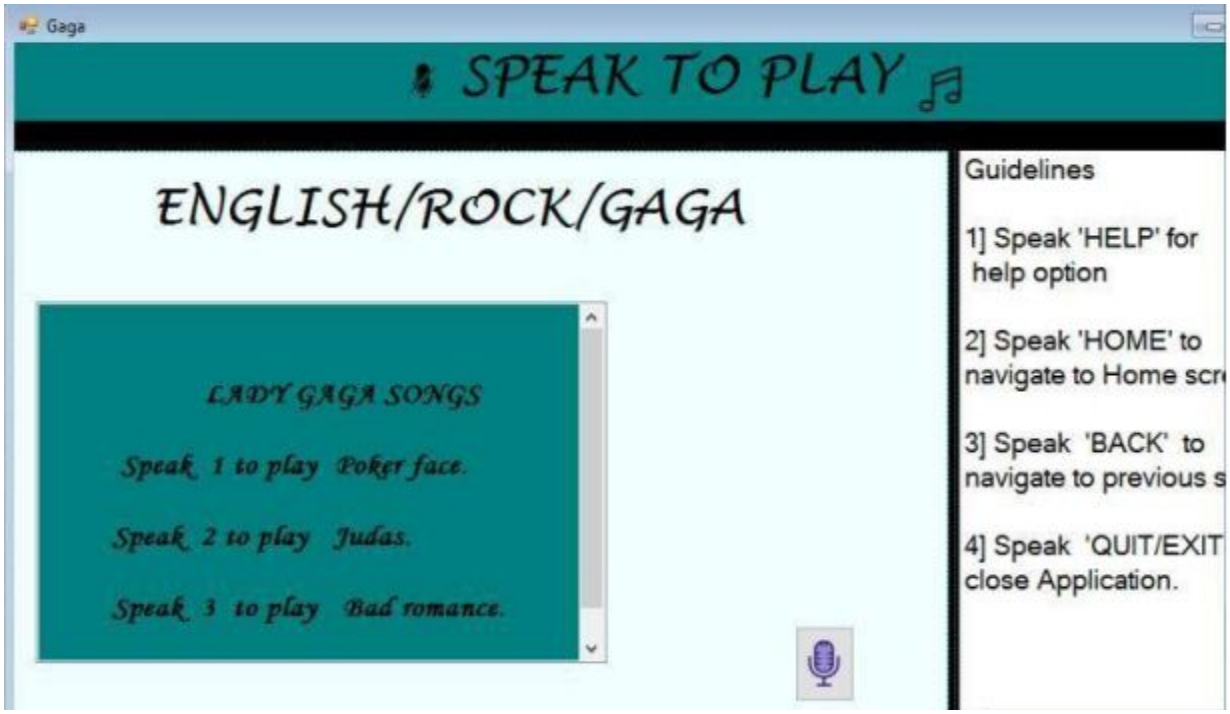

Genre_hindi
SPEAK TO PLAY
HINDI
Speak
SUFI
to play SUFI music
Speak
RETRO
to play RETRO music
Guidelines
1] Speak 'HELP' for help option
2] Speak 'HOME' to navigate to Home screer
3] Speak 'BACK' to navigate to previous scre
4] Speak 'QUIT/EXIT' to close Application.


Genre_english
SPEAK TO PLAY
ENGLISH
Speak
EDM
to play EDM music
Speak
ROCK
to play ROCK music
Guidelines
1] Speak 'HELP' for help option
2] Speak 'HOME' to navigate to Home sc
3] Speak 'BACK' to navigate to previous
4] Speak 'QUIT/EXI close Application.

3. Singer: This interface will give user, the choice to select the singer of their choice under each category



4. Songs : In this interface user has to select the preferred song. There is an index associated with each song and user has to speak the corresponding index to play the song



PROJECT FUNCTIONALITY (EXTRA)

DESCRIPTION

On any page, the user has the option to navigate to other pages using the following voice commands. These details are mentioned on every page on the right hand side panel.

“HOME”

From any page, if this command is spoken, the user is taken to the “WELCOME” page.

“BACK”

From any page, if this command is spoken, the user is taken to the previous page he/she had been on.

“HELP”

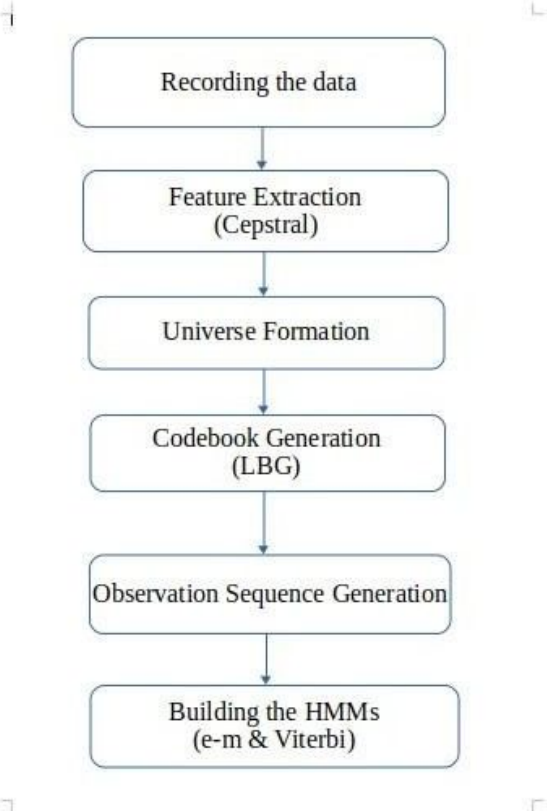
From any page, if this command is spoken, the user is taken to the “HELP” page. This page contains detailed information on how to use the project, and answers to common queries that a user might have.

“QUIT”OR**“EXIT”**

From any page, if this command is spoken, the project is terminated.
On any page, once the user has spoken his choice, a confirmation dialog box pops up, which asks the user if the word that is recognized is the same word he has spoken. The user answers “YES” or “NO” accordingly.

PREPROCESSING

FLOWCHART



DESCRIPTION

Recording the data:

Each command that will be spoken by the user is recorded a certain number of times (20 or 15), using the recording module. The recording module will store the recorded data in .txt and .wav file formats. The sampling frequency is 16kHz and the bit rate is 16bps.

Feature Extraction:

Each .txt file now has a number of samples. The data is then made to undergo the following operations

DC- Offsetting

Normalization

Frame Blocking

Application of Hamming Window

Calculation of Ri's

Calculation of LPC coefficients (Ai's)

Calculation of Cepstral coefficients (Ci's)

Application of Raised Sine Window on the Ci's

The windowed Cepstral coefficients are now stored in text files.

Universe Formation

All the cepstral coefficient files thus formed are used to create the universe of data.

Code Book Generation (LBG)

On the universe file thus formed, LBG algorithm is run to form the code book of size 32 (32 code vectors).

Observation Sequence Generation

Using the code book, each cepstral coefficient file is converted to an observation sequence file by comparing each cepstral vector to each code vector and noting the code book index of the code vector that gives the smallest Tokhura's distance.

Building the HMMs (E-M and Viterbi)

Using all the observation sequence files and an initial Bakis model as input, I run the E-M algorithm on the model till it converges. To check after each iteration the quality of the HMM formed, I've run the Viterbi algorithm on it. Now I got 20 HMMs for each word (as there are that many iterations per word). I averaged these models out and use this as my

initial model, to run E-M algorithm on it again. This process is repeated 3-4 times to get the best possible model.

Live Training Module :

5 utterance of the preferred word selected by the user (which are used in the program to train models) is recorded on the voice of new user and the model is trained again taking our models as base model, so if new user tries to operate the application then it easily recognizes the new speaker's voice.

Target User Base

This application can be used by anyone who does not like cluttered interfaces of applications he/she must use. On using this application, he/she may simply speak his choice depending on his/her mood to start playing a song.

Future Scope

This project can be expanded to include the new user training module as part of the GUI application itself.

The options at each stage can be increased.
There is scope to be able to stop the song that is playing using voice command itself, if there is some way to distinguish between the sound of the song and that of the command.

Instead of having to click the microphone button to start recording the word being spoken, some provision can be made to make the recording a continuous process, and the user need only speak whenever he/she is ready.