

Music Player with Speech Recognition Function Design Documentation

by
Anyun Li, Yilei Zhong

OBJECTIVE:

To design/develop a music player capable of functioning in an environment where speech is available as a mean of interfacing with the application.

APPLICATION:

Neal Alewine, Harvey Ruback and Sabine Deligne, in their paper “Persuasive Speech Recognition”, states:

Speech user interface design and validation is a critical step during application development. Because people are accustomed to speaking to other people, designers must account for this preconditioned input method. Commands should be natural and intuitive, resulting in less memorization for new users. Also, acoustically distinct alternative commands for the same task ensure high accuracy for users with different speaking styles.

Some applications are designed to relax such as music player. People are more desired to have more easy ways to interact with the application when they are enjoying the music. It is quite cool that they can remote control the application with their voice when they lying on the bed listening to the music.

OVERVIEW:

The speech recognition music player provides the user with the ability to add music files, delete music file, control the play of songs, and set the play mode just like the normal music player. But it also can use voice commands controlling this music player and in some circumstances it will give audio feedback.

The application begins by adding users’ favorite music. At first it is manual mouse-control just like normal application and all the functions can be controlled by mouse. After user manually adds some songs into this application, user can press the particular blue button to start speech recognition and voice control function. Just like you are talking with application, tell it what you want to do (play the music, pause the music, play the next music, etc.)

If user said “play mode”, the system will respond voice guide to tell the user what kind of play mode user can choose. Then the user determines which play mode he wants and can say it out. The system will recognize the mode. After the play mode is chosen, the system

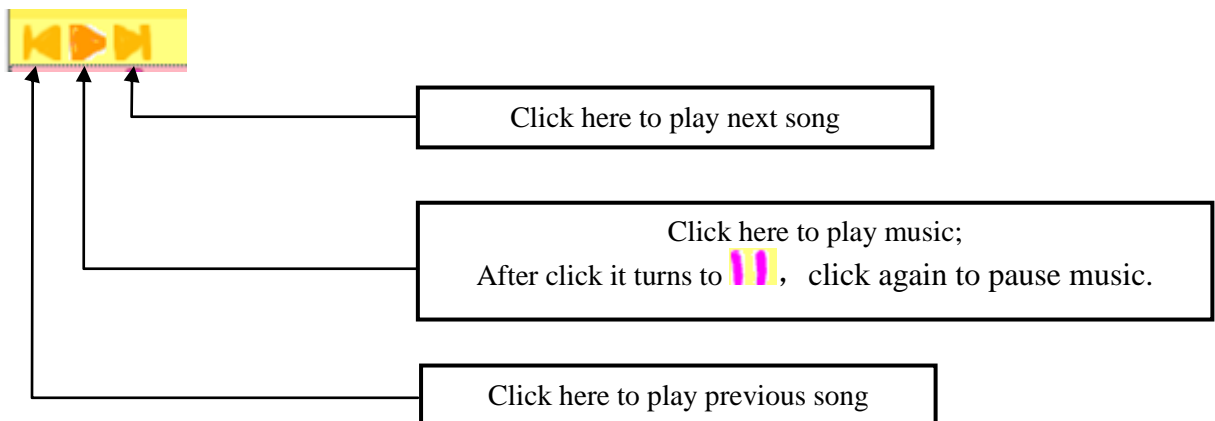
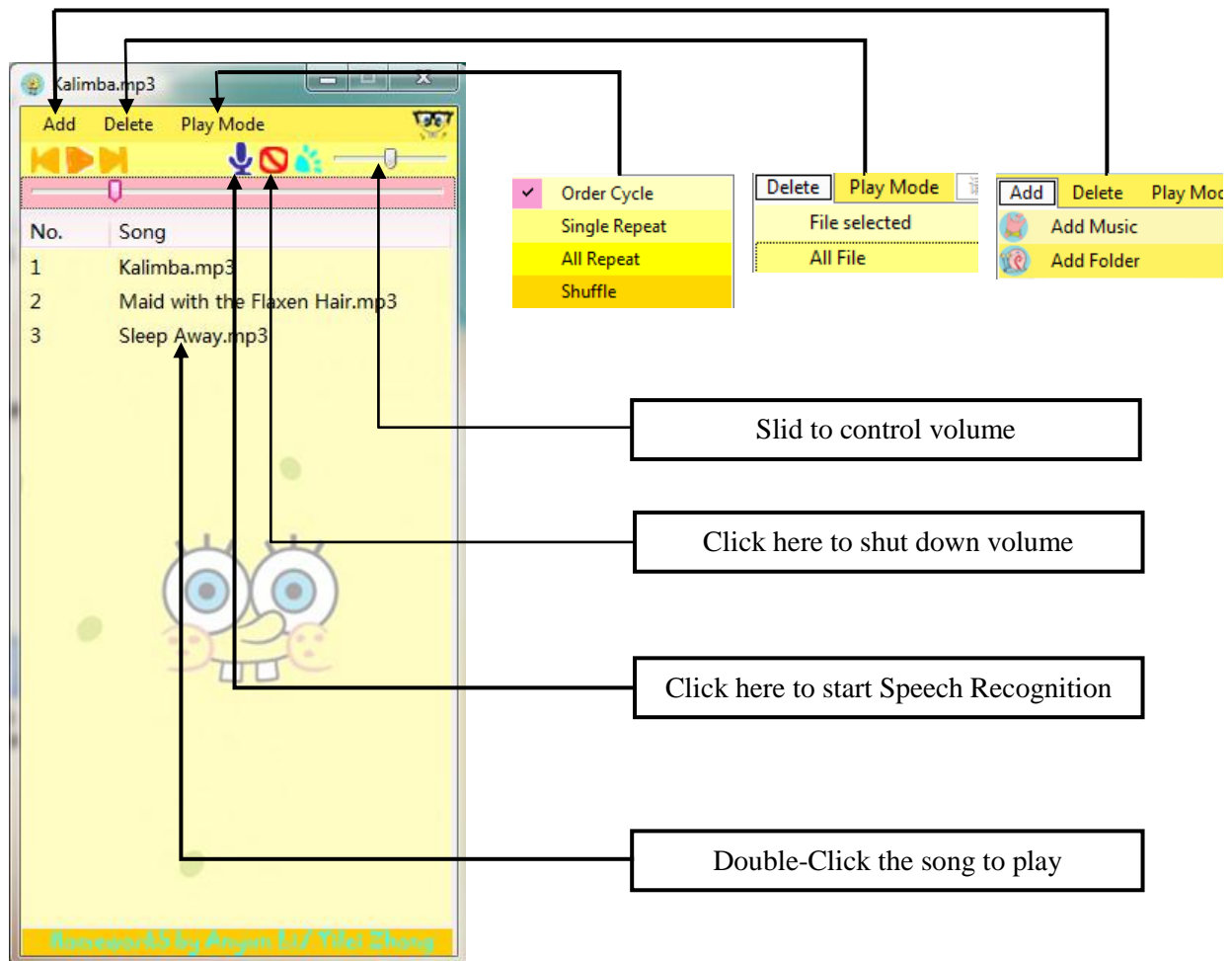
will use that mode to play the music. At any time throughout the course of the application the user may exit simply by saying “exit”, “exit system”, “close” or “close system”

DESIGN NOTES:

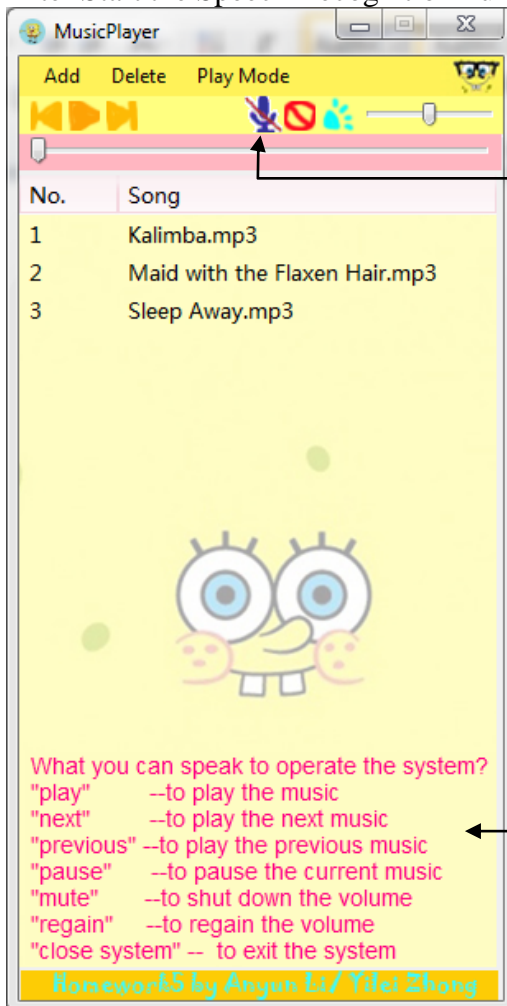
- The program was coded in C# by Visual Studio 2010 under windows 7 OS, using System.Speech in .net 4.0 framework utilizing the speech grammar written in the code.
- This application can be controlled by mouse, voice or both.
- The user begins the speech recognition part of the system by clicking the “start speech recognition” button.
- The system will give the brief guide text to help user what to say to operate the music player.
- The user can exit the application at any time by the verbal command ‘exit’. This option at all points during execution of the program to provide the user with maximum flexibility.
- Grammar is used in the application. It provides flexibility in the format of command recognition.

Function	Command
Play music	play, music, start, play music, play the music
Pause music	pause, stop
mute	Mute, silence, silent
Recover the volume	Regain, recover
Play next song	Next, the next, next one
Play previous song	Previous, last, the last
Close application	Close, exit, close system, exit system

INTERFACE AND FUNCTION INTRODUCTION:





After Start the Speech Recognition Function:





Click here to close speech function

Help Text

SPEECH APPLICATION TASK ANALYSIS:

TASK	INITIATING (signal or event)	ACTION	FEEDBACK	SPECIAL CONDITIONS	POTENTIAL ERRORS	ERROR RECOVERY
Start Speech Function	None	Press  . User listens	The system shows the command prompt label and reply 'Welcome to Music Player. It is starting Speech Recognition. Can I help you?'			
Play music	The system interface shows  , has music file on the list and no music is playing.	User commands the system orally with "play"	The system plays music		System does not recognize response at all. System recognizes incorrect command.	User repeats response. User uses other command to correct the previous command and speaks the correct command again.

Pause the music	The system interface shows  and music is playing.	User commands the system orally with “pause”	The system pauses the music		System does not recognize response at all.	User repeats response.
Play next/previous music file	None.	User commands the system orally with “next/previous”	The system plays the next/previous music file		System does not recognize response at all.	User repeats response.

Choose play mode: 1.Order cycle 2.single repeat 3.all repeat 4.shuffle	Voice prompt (s) from the system.	Respond with one of the four choices.	System repeats user’s choice.		System does not recognize response at all.	User repeats response.
Shut down/Start the volume	None	User responds orally with “mute/regain”	System execute the command		System does not recognize response at all.	User repeats response.
Shut down Speech Function	None	Press  User listens	The system reply ‘Closing Speech Recognition’			
Exit Application	User initiates at any time	User says “close system”	Application closes		System does not recognize response	User repeats “close system”

POSSIBLE CHANGES:

This application is design to have a speech recognition function in the music player so that we can remotely control the main functions of our application. We found it is very convenient that when we are cooking or lying on the bed we can remotely control the computer to play the music.

But when we finally finished programming and test the application. We find many problems needed to be improved. The computer cannot always recognize what you said even if you think your pronounce is right. And computer is harder to recognize the voice command with music playing. And the voice start is a little lag.

We have a test to test all the 20+ grammars in this application. And the result shows that 15 grammars are easy to be recognized by computer while more than 6 are hard to be recognized.

We think that if we add more ambiguous words which are similar to the word which is hard to be recognized to the grammar, may it will help the application to recognize the word easily.

For example, we found the word “regain” is hard to be recognized, we add the words like “regale”, “regard”, and “regal” to grammar to help solve the problem.