CS-308-2016
“Do Not Disturb”
Team no. 10

Pratham Desai, 12D170001
Amal Dani, 120020019
Vishal Agarwal, 120110009
Darsh Shah, 120010010

Project Description - “Do Not Disturb”

“Do not disturb” is a smart sound device. The goal of this device is to alter the behaviour of a music system by receiving feedback from the surroundings about human voice and conversation.
It provides users independence from needing to reduce the volume of music while speaking, and to avoid them to bring the volume back when the conversation ends. It becomes the device’s role to notice that there is human conversation and to accordingly change the volume of the music system.
An analogous application and implementation of this device, is to change the radio frequency when the RJ is speaking and find a channel where there is music playing.

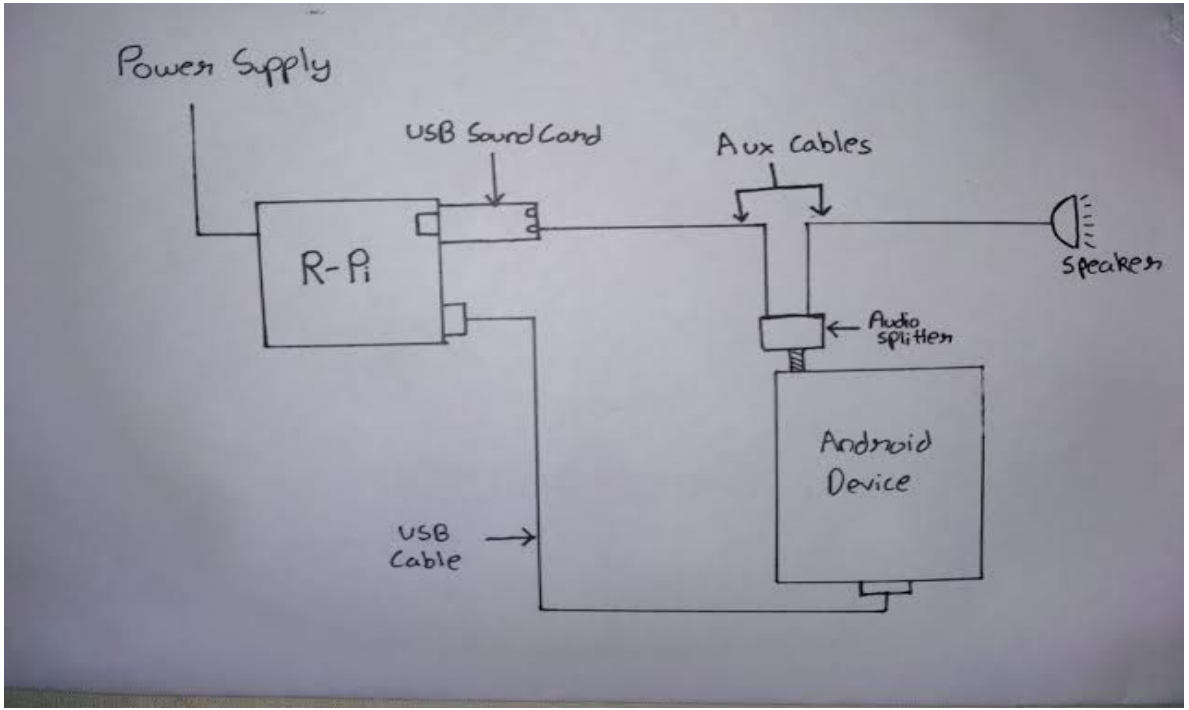
Requirements/Task Specifications Aimed For

- We targeted 2 modes of operation
- First mode: To enable changing of frequency automatically when the current radio channel is not playing music and the RJ is speaking or there is an advertisement playing
- Second Mode: To enable human voice dependent music volume change. The sound of music playing should reduce when there is some human voice in the surrounding and should go back to the original level when the volume reduces

Project Plan

- We decided to divide the work into 2, one would deal with the hardware side of implementation - setting up raspberry pi and sound card interfacing. Pratham Desai and Darsh Shah worked on this initially.
- The second group of Amal Dani and Vishal Agarwal, worked on creating a classifier that would distinguish between music and human voice. After recording the audio, the code would analyse it and using a trained classifier, give a confidence measure on whether it is just music or with some human voices.
- The first milestone was developing the ‘First Mode’ by 23rd March 2016
- By 6th April, we developed the ‘Second Mode’, all 4 of us worked equally for this part

Block Diagram



Description of States

- ‘First Mode’ : Changing Radio Frequency
  - Normal state: A channel is playing
    - On detecting a human voice signal with probability >= 0.95 => change radio frequency
    - On detecting a human voice signal with probability < 0.95 => stay in the same state
- ‘Second Mode’ : Volume control based on human voice
  - Normal state: Music plays at a given level
    - On receiving an external human voice signal with probability >=0.5 => go to reduced volume state
  - Reduced volume state: Music is playing at a low level
    - On receiving a signal that human voice signal has reduced with probability >=0.5 => go to normal state

Innovation and Challenges

- Being new to audio processing, it took us a lot of tries to interface the sound card with the raspberry-pi
- The classifier training required sufficient time. We tried various audio analysis libraries in order to discover the one which would work best for our use case (capture the burstiness property of the sound)

Task Completed (1/2)

- As mentioned, we completed the ‘First Mode’ by March 23rd and the ‘Second Mode’ by 6th April
- After getting familiar with raspberry-pi, and overcoming those initial glitches, we were comfortable with using it as the base for our device
- Fixing the entire device setup, and coding modules to control the phone on which music/radio was playing, required reading of a lot libraries and APIs
- This was figured during the completion of the ‘First Mode’

Task Completed (2/2)

- Positioning of the mic to get a reasonable estimate of the surrounding human sound was another challenge
- Dealing with music and music+human sound was a difficult task in terms of both software (finding a good classifier) and in terms of enabling proper input to the raspberry-pi device
- Finally through trial and error, and good judgement on how the mic is to be adjusted, we were able to ensure that even this mode worked just fine

Review

Review, Test Plan/Cases (test cases implemented), performance metrics

- We were very satisfied with the working of the device in both the modes
- The testing environment was a controlled environment inside our hostel room
- We tested several cases in both ‘First Mode’: radio frequency changing mode and the ‘Second Mode’: volume reduction mode
- In the first mode, we had no control over what the radio channel was playing. Yet there was a very good performance in changing the channel when there was only human voice being played/heard

Review

Review, Test Plan/Cases (test cases implemented), performance metrics

- In the second mode, we tried several songs
  - While we expected it to work well with instrumental music, which it did. It also worked well with rap music like DJ Bravo's popular 'Champion Song'. In fact in the final demo we showed its successful working to the TAs
- There were almost zero false positives, which should be the aim for such a device. There should not be any annoying/unnecessary changes
- One delay that we incur is owing to the processing time of the relatively weak raspberry-pi processor. On a stronger RAM this delay of 3 seconds that the device incurs to react to a change in environment, would be reduced

Reusability features

- Library pyAudioAnalysis can be reused to perform a multitude of tasks
- Code for recording audio via the soundcard, transferring files from the android device to the computer and sending tap events to the android device are some of the features which can be reused
- The entire hardware & software setup can be reproduced to do any kind of audio analysis. For instance, one could make a system which detects human mood based on his/her voice.

Future Enhancements

- We can make the product much more compact and pleasant
- The algorithm implemented does not respond very well to radio jockey with background music, we can work on the machine learning aspect of it and implement a better model
- The placement of mic to optimally absorb the music and conversation levels is an open ended question and requires some deeper thought
- Presently, the code has certain amount of latency, we can work on it to reduce the latency