Sound Localization Block Validation

Block Owner: Abdulla Al-Ansari
Date: Feb-11-2021

Design Details

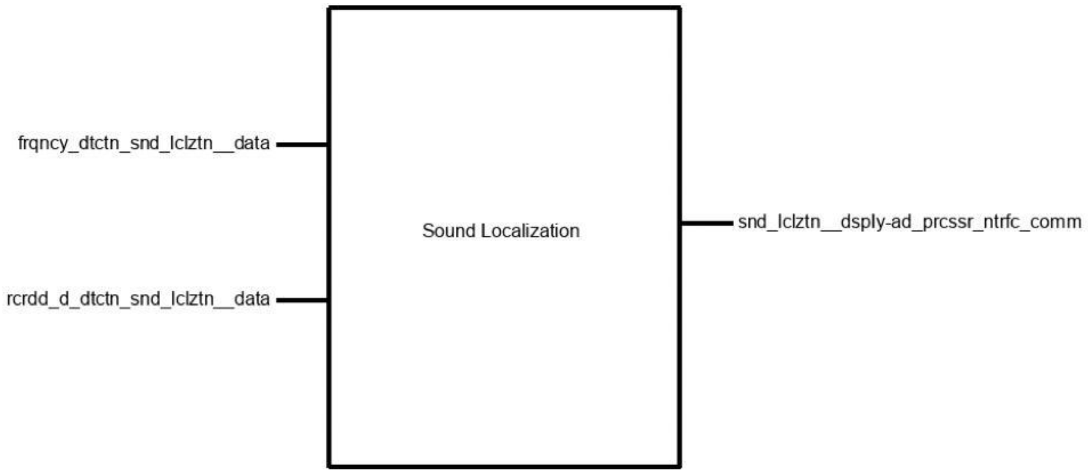
frqncy_dtctn_snd_lclztn__data
rcrdd_d_dtctn_snd_lclztn__data
Sound Localization
snd_lclztn__dsply-ad_prcssr_ntrfc_comm

Figure 1: Black Box Diagram of Sound Localization Block

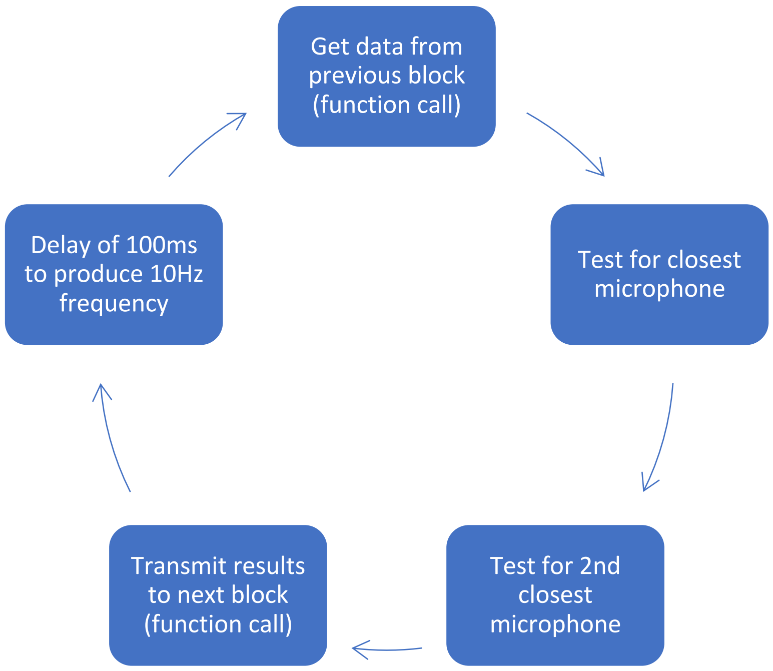
Get data from previous block (function call)
Test for closest microphone
Test for 2nd closest microphone
Transmit results to next block (function call)
Delay of 100ms to produce 10Hz frequency

Figure 2: Cyclic Flow of Sound Localization Block

Pseudo Code:

Task #	Task Details
1	Declare / Define necessary variables to store data temporarily
2	Get data from previous block stored array (digital 0-1023)

3	Test which is the closest microphone to sound
4	Test which microphone is 2nd closest to sound (if any)
5	Finalize results and store it to be able to transmit to next block
6	Function call to initialize USB serial communication
7	Give 100ms delay thus producing an iteration frequency of 10Hz
8	Loop back to step 2

Design Validation Overview

A little background about this project is necessary here to understand this block better. So the project will take sound signals from 4 microphones arranged in an array like 4 corners of square. Also, it is capable of taking sound from a sound file instead of microphones. The analog sound signals would be then processed by first digitizing then apply fast fourier transforms etc and then cleaned data for all 4 microphones would be stored in an integer array.

Now once the data is stored, this block comes in practice and it will take data from the array discussed above and apply test expressions to find out the distance of sound from the microphone which is getting the strongest signal.

Once the closest microphone is found, the 2nd closest microphone will be found similarly. Now once we get distance of sound from both of these microphones giving positive response to sound being closest, using their mutual distance, the distance and direction of sound can be inferred.

The guideline and help over this sound localization was taken from numerous papers primarily this and this where numerous techniques over this topic are discussed.

Once the sound is localized, the results are stored in temporary variables and then later on in next block would be transmitted to android using USB serial communication. However in scope of this block only a function call is done for USB serial communication.

Finally, this whole process will loop back with a frequency of 10Hz by producing a delay of 100ms after every iteration.

So, either the sound localization would be done on data from microphones or recorded file, in both of cases same set of instructions will be executed.

Design Validation Interface Table

Interface Property	**Why is this interface this value?**	**Why do you know that your design details for this block above meet or exceed each property?**
snd_lclztn__dsply-ad_prcssr_ntrfc_comm : Output		
Datarate: 57600 baud	We wanted fastest communication possible but at 57600 baud, the bottleneck will be the serial connection between the host and the cards see overview.	Tested through experimentation, on exceeding 57600 baud rate, data was getting damaged during transmission.
Messages: (x,y) coordinates	To locate sound in 2d plane, two dimensions were required to completely locate sound telling both the distance and direction data. Further guideline from here	The project requires sound localization in 2d plane so considering the microphone array as origin, the (x,y) coordinates will efficiently provide information about sound with reference to microphones / device.

Protocol: USB Serial	USB 2.0 offers greater transfer speeds then Bluetooth or other relevant protocols, however, it is limited by the length of the USB cable. Ref here	As the distance of android would not be far from controller or microphone array so, USB serial seems most suitable protocol for this project's requirement providing noise free fast communication. More info about idea here
frqncy_dtctn_snd_lclztn__data : Input		
Datarate: 10 Hz	Data rate for communication commands was lowered down on purpose to avoid false command transmission. 10 Hz communication means a delay of 100ms after every iteration of data intake More info taken from here	The frequency data would not be exactly real time as it has an adc in channel then fourier transforms and then data cleaning so a data rate higher than 10Hz would not be suitable here.
Messages: Time difference of arrival	Time difference of arrival (TDOA) is considered the base of sound localization giving information of distance of sound from a particular microphone This idea is clarified in depth in this paper.	Once TDOA information from all microphones would be taken, then using test experessions in code, it would be inferred that to which microphone the sound is closer thus giving distance from sound and by further comparing TDOA of adjacent microphones, direction data will be inferred thus finally extracting 2d coordinates out of it. Details in this paper
Protocol: Function call	This block is not directly involved with microphones and digitized clean data would be taken from previous block using a call to function which will take data stored by previous block.	As the project involves multiple code snippets so inorder to merge them all or communicating with another code part, the best way is to segregate code in functions doing specific tasks. This will improve code's readability and flexibility for future upgrades too. Guide taken on this topic here
rcrdd_d_dtctn_snd_lclztn__data : Input		
Datarate: 10 Hz	Data rate for communication commands was lowered down on purpose to avoid false command transmission. 10 Hz communication means a delay of 100ms after every iteration of data intake More info taken from here	Also for the recorded sounds, data would not be exactly real time as it would be analog data and would first be digitized to process and then fourier transforms and then cleaning which takes time so in order to avoid data loss a 10Hz frequency is considered
Messages: Time difference of arrival	Time difference of arrival (TDOA) is considered the base of sound localization giving information of distance of sound from a particular microphone This idea is clarified in depth in this paper.	Once TDOA information from all microphones would be taken, then using test expressions in code, it would be inferred that to which microphone the sound is closer thus giving distance from sound and by further comparing TDOA of adjacent microphones, direction data will be inferred thus finally extracting 2d coordinates out of it. Note: Here no actual microphones are involved and recorded sound is used but since 4 microphones are

		also our origin of 2d plane so will still process recorded sound data as per locations of all 4 microphones to maintain same scale. Details in this paper
Protocol: Function call	This block is not directly involved with microphones and digitized clean data would be taken from previous block using a call to function which will take data stored by previous block.	As the project involves multiple code snippets so inorder to merge them all or communicating with another code part, the best way is to segregate code in functions doing specific tasks. This will improve code's readability and flexibility for future upgrades too. Guide taken on this topic here

References

N-dimensional N-microphone sound source localization https://link.springer.com/article/10.1186/1687-4722-2013-27

USB 2.0 vs Blutooth: https://smallbusiness.chron.com/usb-20-vs-bluetooth-47408.html

FHT Arduino : https://create.arduino.cc/projecthub/janux/fht-audio-spectrum-visualizer-83bba0

Arduino to Android USB communication: https://www.digikey.be/en/maker/projects/how-to-use-your-android-to-communicate-with-your-arduino/aed1f8e3fa044264a4310c6b3b2a4364

Dividing code into functions: https://www.cs.utah.edu/~zachary/computing/lessons/uces-10/uces-10/node11.html

Simple solutions for hyperbolic and related position fixes :
http://jmargolin.com/sense/refs/ref26_fang.pdf