COMP9336 Assignment Report

Lesi Huang z5169900

Task 1 Single Tone Detection

In this task, I generated a sinusoidal wave, play it using AudioTrack Android API, and detect this tone with Goertzel Algorithm.

According to Nyquist frequency, the highest frequency a given sample rate can represent unambiguously is half the sample rate. In other words, to correctly generate and detect the frequency range up to 20kHz, a sample rate higher than 40kHz is needed. Thus I choose the sample rate of 44100/s, the highest one supported on my device.

On the transmitter side, after determining the sample rate, the following formula is used to generate a 16bit PCM sinusoidal wave:

$$sample[i] = sin\left(\frac{2\pi f i}{sample\ rate}\right) \cdot Short.MAX\_VALUE$$

On the receiver side, the accuracy of tone detection -- the frequency resolution, depends on the block size of the Goertzel Algorithm. To achieve a frequency resolution of 5Hz, a block size of 44100/5 = 8825 is chosen. Therefore, it takes 8825(block size)/44100(sample rate) = 0.2 seconds to gather sufficient sound information for the first block, which is acceptable.

To detect tone frequency, the Goertzel Algorithm is run for every 5Hz within the range of 0Hz – 22050Hz, then the frequency yields the largest magnitude would be the detection result.

Task 2. Extension of Single Tone Detection

In this task, I explore the limits of detectable frequencies on my device, then choose nine reliable frequencies respectively for audible and inaudible sound to encode number 1-9.

In theory, for the 44100 sample rate, the maximum frequency that works should be 22050 Hz, and the minimum frequency should be 0 Hz. My device can detect reliably from the frequency 100 up to the frequency of 20000. which is close enough to theoretical limits.

The nine audible frequencies chosen are: 400 420 440 460 480 500 520 540 560
The nine inaudible or nearly inaudible frequencies chosen are: 17000, 17300, 17600, 17900, 18200, 18500, 18800, 19100, 19400

Task 3. Dual Tone Detection

In this task, I combine and transfer two frequencies from standard DTMF.

On the transmitter side, I update my sampling formula to combine 2 frequencies:

$$sample[i] = \left[sin\left(\frac{2\pi i \cdot f_1}{sample\ rate}\right) + sin\left(\frac{2\pi i \cdot f_2}{sample\ rate}\right)\right] \cdot Short.MAX_{VALUE}/2$$

On the receiver side, I apply the Goertzel Algorithm to find the frequency with the largest magnitude among row tones, and that among column tones.

Task 4. Packetized Data Communication with Audio Tones

In this task, I develop an audio-based data transmission system.

For modulation, my system uses binary frequency shift keying(BFSK). Frequency 1209Hz represents bit 1 and frequency 697 represents bit 0. The receiver applies the Goertzel algorithm to detect frequencies.

To achieve a data rate of 100 bits/s, which means a 0.01s symbol duration, I set the Goertzel algorithm block size to 44100(sample rate) × 0.01(symbol duration) = 441. Under this block size, the frequency resolution is 100Hz, so the two frequencies can be readily distinguished.

The packets include a preamble byte 101011 for synchronization, followed by a byte indicating the payload length, and then the payload in ASCII format.

Task 5. Error correction

In this task, I implement (7, 4) Hamming code with additional even parity bit(SECDED). This code achieves single error correction and double error detection, at the cost of reducing the actual data rate by 50%. The hamming code only applies to the payload length byte and payload. The preamble byte is not encoded.