GOERTZEL, MATCHED FILTER ALGORITHMS AND FFT BASED APPROACH FOR DTMF DETECTION

Zejun Yin

251388005

尹泽钧

Introduction

DTMF (Dual-Tone Multi-Frequency) signals are commonly used in modern telephone systems to represent dialed numbers. Each DTMF key (0-9, *, #, and others like A-D) is transmitted as the sum of two sinusoidal tones: one from a lower-frequency group (697 Hz, 770 Hz, 852 Hz, 941 Hz) and one from a higher-frequency group (1209 Hz, 1336 Hz, 1477 Hz, 1633 Hz). For instance, pressing the digit "1" generates tones at 697 Hz and 1209 Hz, which are added together. DTMF detection is important because it is widely used in telephony systems, voice response systems, and embedded devices needing to interpret keypad inputs. The pressed key must be accurately detected to identify the two frequencies present. Typically, it needs to balance computational efficiency and detection reliability. In application, it is common to use Goertzel algorithm to detect DTMF. In this report, I implemented and compared three methods for DTMF detection: Goertzel algorithm, Matched filter and FFT-based approach. I analyzed their respective advantages and disadvantages in terms of computational cost and detection accuracy.

Algorithm Descriptions

This section provides a more detailed description of each method, along with a theoretical background.

2.1 Goertzel Algorithm

The Goertzel algorithm is a technique for computing the Discrete Fourier Transform (DFT) at one or a small number of specific frequency bins without performing a complete FFT. Conceptually, it implements a recursive filter:

$$s[n]=x[n]+2\cos(\omega 0)s[n-1]-s[n-2]$$

Where $\omega 0=2\pi k/N$, k is the frequency bin index, and N is the signal length. By examining the final filter states, one obtains the magnitude of the signal at the chosen frequency. Goertzel algorithm is efficient for detecting a small set of frequencies and widely used in DTMF detection, where typically 8 distinct frequencies must be checked. There is lower computational overhead than a full FFT when only a few frequencies are needed.

2.2 Matched Filter

Matched filtering (or template matching) involves correlating the input signal with a reference signal (in this context, a sinusoid at the target frequency). For a frequency f_0 :

$$\text{corrVal}= \left| \sum_{n=0}^{N-1} x[n] \cdot \sin(2\pi f0 n/fs) \right|$$

A larger correlation value indicates that x[n] contains a strong component at f_0. DTMF detection simply requires repeating this process for each candidate frequency. It is conceptually straightforward: multiply the input by a reference sinusoid and sum. The implementation is easy, but detecting multiple frequencies requires repeated calculations.

2.3 FFT-Based Approach

The FFT (Fast Fourier Transform) has a form like this:

$$X[k]= \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad \text{,devided into}$$

$$X[k]= \sum_{r=0}^{N/2-1} x[2r] W_{N/2}^{kr} + W_N^k \sum_{r=0}^{N/2-1} x[2r+1] W_{N/2}^{kr}$$

It computes the frequency spectrum of a signal with complexity O(NlogN) after Butterfly Operation. Once the spectrum is computed, the amplitude at any frequency bin can be obtained by indexing the FFT result. It is a single transform reveals the entire spectrum and more efficient if you need to detect multiple frequencies at once.

Simulation Results

3.1 Signal Generation

- Example DTMF digit: “5483887086,” which is a phone number
- Sampling rate: f_s=8000 Hz
- Duration: 0.25 seconds
- Signal number N=1024
- A test signal by adding the two sinusoids (low-frequency + high-frequency).

I then defined a list of all DTMF frequencies (697, 770, 852, 941, 1209, 1336, 1477, 1633) for detection.

3.2 Amplitude Comparison

I measured how strongly each of the three methods (Goertzel, matched filter, FFT) responds to each of these 8 frequencies. To be brief, I only demonstrate a typical amplitude plot of the first digit “5”:

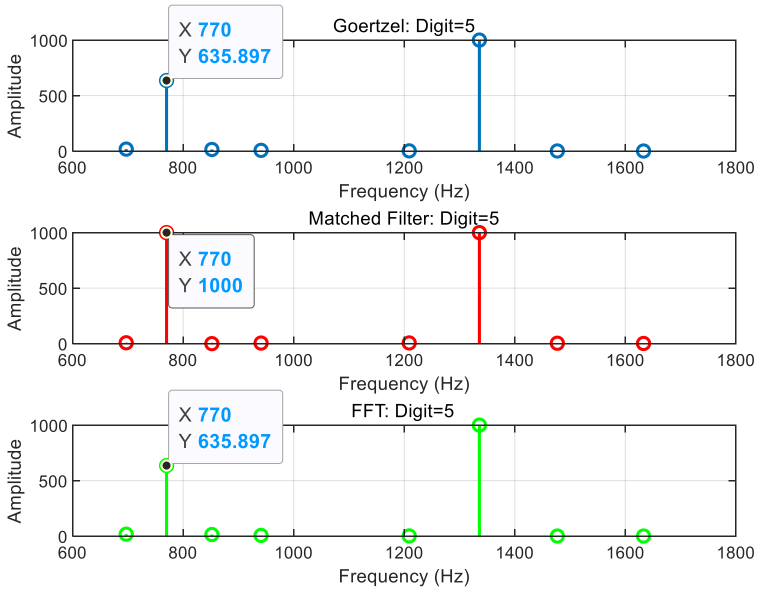

Figure 1. The amplitude comparison of three algorithms for digit 5

In figure 1 we can see that all three algorithms can distinguish the target frequency (high amplitude at the correct frequencies (770 Hz, 1336 Hz)), and significantly smaller amplitude at the other six frequencies. Although the matched filter has the highest amplitude, it does not affect the result. This confirms that all three methods successfully detect the two frequencies present in the test signal.

3.3 Computation Time Comparison

To assess efficiency, I compared the average runtime for each digit for all of three algorithms, and repeated many times to obtain stable averages. The calculated result is shown in the Figure 2 below.

```
>> testDTMFNumber
============= Detection Results Comparison =============
    Index    Original     Goertzel     Matched      FFT
    _____    ________     ________     _______     _____

      1         5          {'5'}        {'5'}      {'5'}
      2         4          {'4'}        {'4'}      {'4'}
      3         8          {'8'}        {'8'}      {'8'}
      4         3          {'3'}        {'3'}      {'3'}
      5         8          {'8'}        {'8'}      {'8'}
      6         8          {'8'}        {'8'}      {'8'}
      7         7          {'7'}        {'7'}      {'7'}
      8         0          {'0'}        {'0'}      {'0'}
      9         8          {'8'}        {'8'}      {'8'}
     10         6          {'6'}        {'6'}      {'6'}

============= Execution Time Comparison =============
    Goertzel_1digit    Matched_1digit    FFT_1digit
    _______________    ______________    __________

      3.1695e-05         0.00011906      5.3922e-06

fx >>
```

Figure 2. Each digit runtime for all three algorithm

The matched filter method processes the most time, which was consistent with our expectation due to its high-demanding. However, the FFT execution time was even less than Goertzel algorithm, surprisingly. Since the complexity for FFT is O(NlogN), and for Goertzel is O(K*N), and K is 8 here. After referring information, I learned that in practice FFT can often be faster than Goertzel due to its highly optimized implementations (e.g., FFTW) that leverage CPU caching, SIMD instructions, and parallelism, while Goertzel is only script iteration. To visually demonstrate the difference of complexity for the two algorithms, I wrote a FFT script manually, and compared it with the embedded FFT function.

```
myFFT result:
   3.8549 + 4.3206i   0.1483 - 0.0991i   0.6420 + 0.8806i  -0.5912 - 0.5778i  -0.3359 + 1.2839i   0.1189 + 0.2020i   1.2968 + 0.3590i   0.5411 + 1.3088i

MATLAB builtin fft result:
   3.8549 + 4.3206i   0.1483 - 0.0991i   0.6420 + 0.8806i  -0.5912 - 0.5778i  -0.3359 + 1.2839i   0.1189 + 0.2020i   1.2968 + 0.3590i   0.5411 + 1.3088i

Difference norm: 3.59753e-16
myFFT total time: 0.2482 s
built-in fft time: 0.0047 s
fx >>
```

Figrue 3. Comparison of FFT script and FFT function

In figure 3 it is evident that the FFT function in Matlab ran faster than executed FFT step by step. No wonder that FFT runs faster than Goertzel.

Conclusion

In this report, I have implemented and compared three common DTMF detection methods: Goertzel, matched filtering, and FFT-based detection. I measured in accuracy and computational cost. It turned out all three methods will correctly identify the two prominent frequencies in a DTMF tone. And for the computational cost, Matched Filter is straightforward but requires separate correlations for each frequency. Goertzel is efficient for a small set of frequencies and should be the fastest algorithm in dealing with DTMF theoretically. FFT yields the entire spectrum in one go, which is helpful for many frequencies but may be less optimal for one or two frequencies. However, in my project FFT is the best algorithm due to its highly integration in Matlab.

In many practical real systems of telephony, the Goertzel algorithm is used for DTMF detection because it is fixed for a specific set of frequencies and is computationally efficient. However, an FFT-based approach is also quite feasible, particularly in environments where you might check multiple frequencies simultaneously.

Furthermore, for the practical cnsiderations, in hardware-constrained environments, Goertzel is often preferred. FFT may be employed for analyzing multiple frequencies, given optimized FFT libraries. Matched filtering remains an easily understood approach but can grow more expensive as the number of frequencies increases.