Dual-Tone Multiple Frequency Detection and Estimation

Nicholas Kottenstette

March 18, 2009

1 Derivation of DTMF Generator

A dual-tone signal, $x(n)$, consists of two sinusoidal signals, $x_{\omega_l}(n)$ and $x_{\omega_h}(n)$, representing the respective low and high frequency component of the signal. Instead of resorting to solving a Taylor Series expansion of $sin(\omega n)$ to generate the tones we can use a marginally stable IIR filter. The z-transform of $sin(\omega n)$ is given in (1).

$$X_\omega(z) = \frac{z^{-1}sin(\omega)}{1 - 2z^{-1}\cos(\omega) + z^{-2}} \tag{1}$$

The corresponding difference equation, will have the form given in (2),

$$x_\omega(n) = 2\cos(\omega)x_\omega(n-1) - x_\omega(n-2); x_\omega(0) = 0, x_\omega(1) = \sin(\omega) \tag{2}$$

Similar results have been cited in works such as those by [1]. With these results, a dual-tone signal can be generated using (3), noting that (f_l, f_h) represent the respective dual-tone frequency pairs on a touch-tone phone [2].

$$x(n) = x_{\omega_l}(n) + x_{\omega_u}(n); \omega_l = 2\pi\frac{f_l}{Fs}, \omega_h = 2\pi\frac{f_h}{Fs} \tag{3}$$

The low-tone frequencies, for this project were specifically taken from the corresponding set $f_l = \{687, 770, 852, 941\}$ Hz. The high-tone frequencies, for this project were specifically taken from the corresponding set $f_h = \{1209, 1336, 1447, 1633\}$ Hz. Before, concluding this derivation, we would like to emphasize that the difference equation given in (2) lends itself for various parametric algorithms for PSD estimation of sinusoids in additive white noise, see [3] for additional details.

2 Derivation of DTMF Detector

The DTMF detector analyzes the input data in the following steps:

1. Filter input data, $x(n)$, with a band-bass filter with pass-bands centered at DTMF -tones frequencies. The output of the filtered tone data will be referred to as $xf(n)$.

2. After applying filter, scale outputs to have $+/-1.0$ peak-to-peak range.

3. Calculate PSD of $xf(n)$ using the Welch Method specifying the FFT length, $fftlen$, the data-segment step size, D, the data-window size, M, the segment averaging data length, L, and the window, $w(n)$. See [4] for additional details. The resulting output, $E(k,i)$ corresponds to the resulting Welch PSD estimate at index k and i. The k index corresponds to the frequency, $F_k = \frac{Fs \times k}{fftlen}$. The i index corresponds to the Welch PSD estimate of the windowed data starting at $xf(iD+m), m \in 0, 1, ..., M$. N.B. the exact indexing of i is slightly modified, there is an abuse in notation, in the software implementation. Since we are sliding a Welch PSD estimate along our sampled data. The Welch calculation will actually be the average of the Nw samples centered at index i. Refer to the code to fill in any missing details, specifically:

   ```
   calc_psd_welch(x,D,M,w,fft_len,Nw) in dtmf_det.m
   ```

4. Remove unnecessary PSD information, the powers at frequencies we are not interested in. Then determine the peak power within the frequency-bands of interest. This is to account for the variance in the DTMF tones. Refer to:

   ```
   freq_peaks(E,fmin,fmax,Fs) in dtmf_det.m
   ```

5. Apply maximum-likelihood detection, by eliminating possible tones, through a threshold-detection scheme. In which tones are only valid if a pair of low and high frequency tones exceed a given threshold level, $Thresh$, for Nt consecutive tones relative to index i. Refer to:

   ```
   tone_detect(E,Nt,...,f_name) in dtmf_det.m
   ```

The differential scheme is the same as the above, except that a second band-pass filter is applied to detect voice disturbances in the harmonics of the tones we are trying to detect, $xhf(n)$. We apply the same processing to $xhf(n)$ to generate a decimated-frequency-peak estimate of the harmonic tones. We then subtract the tone estimate, Ep, from the harmonic estimate, Eph, resulting in a differential estimate, $Edif$. Which we then apply at point 5 in the above algorithm. Refer to:

```
%dtmf_det.m
if en_di
  Edif = Ep - Eph;
else
  Edif = Ep;
end
```

The bandpass-filter was a linear-phase FIR filter, in order to keep a constant group delay of the data. The implementation is a slight modification of the audio equalizer developed and discussed by [5].

References

[1] John G. Proakis, Dimitris G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications*. Prentice-Hall, Inc., 1996. (pp. 352–354)

[2] Robert L. Stevenson, EE598D – Project #2

[3] John G. Proakis, Dimitris G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications*. Prentice-Hall, Inc., 1996. (pp. 948–956)

[4] John G. Proakis, Dimitris G. Manolakis, *Digital Signal Processing Principles, Algorithms, and Applications*. Prentice-Hall, Inc., 1996. (pp. 911–913)

[5] Nicholas E. Kottenstette, EE598D – Project #1 Report