Computer Engineering Four Assignment One

Michael Still
964076

Table of Contents

Part A

Perform time-domain analysis with a frame size of 512

Input the file, play it back and state what the spoken phrase is

The spoken phrase is "Human Computer Communication".

Compute the energy contour in dB

The following Matlab script was used to generate the energy contour in dB, including using the first frame as a reference of 0 dB.

```
function [energy, edb] = econtour(filename)

framesize = 512;
numsamples = wavread(filename, 'size');

% Read the whole file into one big vector
% Check the wavread doco for a better way of doing this
[samples, fsample, nbits] = wavread(filename, numsamples(1));

% Compute the energy contour
fcount = 1;
while fcount < numsamples(1) / framesize
    temp = samples((fcount - 1) * framesize + 1: fcount * framesize);

    % Calculate the energy, and the reference if required. Apply that reference
    energy(fcount) = temp' * temp;
    edb(fcount) = 10 * log10(energy(fcount));
    if(fcount == 1)
        eref = edb(1);
    end
    edb(fcount) = edb(fcount) - eref;

    % Next frame
    fcount = fcount + 1;
end
```

Code: econtour.m

Compute the normalised autocorrelation contour

The following Matlab script was used to calculate the autocorrelation contour:

```
function [acorr, ffreq] = autocorr(filename, energy)

framesize = 512;
numsamples = wavread(filename, 'size');
vthresh = 0.375;

% Read the whole file into one big vector
% Check the wavread doco for a better way of doing this
[samples, fsample, nbits] = wavread(filename, numsamples(1));

% We need to know kmin and kmax before we can continue
kmin = fsample / 200;
kmax = fsample / 80;

% Compute the autocorrelation
fcount = 1;
while fcount < numsamples(1) / framesize
```

```
    temp = samples((fcount - 1) * framesize + 1: fcount * framesize);

    for count = kmin:kmax
        acalc(count) = temp(1:framesize - count)' * temp(1 + count: framesize);
    end

    % Now normalize
    [amax, indexmax] = max(acalc);
    acorr(fcount) = amax / energy(fcount);

    if (acorr(fcount) > vthresh)
        ffreq(fcount) = fsample / indexmax;
    end

    % Next frame
    fcount = fcount + 1;
end
```

Code: autocorr.m

Set a suitable autocorrelation value for the voiced/unvoiced threshold and compute the fundamental frequency contour

The fundamental frequency is calculated by the autocorrelation function in the code in the previous section.

Plot all three of the above contours with a common time axis and on that plot mark the phoneme boundaries and label the phonemes

Please refer the to next page for this graph, with annotation.

Documentation

The two scripts presented earlier in this section to calculate the values required are tied together by the following Matlab script:

```
function [] = questionOne(filename)

% Generate the energy contour (part b)
[en, edb] = econtour(filename);

% Generate the autocorrelation (part c)
[ac, ff] = autocorr(filename, en);



% Plot all of these
subplot(3, 1, 1), plot(edb)
grid on
ylabel('Energy (dB)');
title('Energy, with first frame as reference');

subplot(3, 1, 2), plot(ac)
grid on
title('Normalized autocorrelation');

subplot(3, 1, 3), plot(ff)
grid on
title('Fundamental frequency');
```

Code: questionOne.m

Perform frequency-domain analysis on a frame in the centre of the first vowel /u/ and on a frame in the centre of the consonant /sh/

The following scripts were used to perform these calculations:

```
function [ ] = questionTwo( filename )

% The framesize is 512
framesize = 512;

% Find the /u/ frame and perform a frequency domain analysis
[sdata, fsample] = ph_u(filename);
[lms, cep, smoothed] = fd_analyse(sdata, fsample, framesize);

% Plot all of this
figure(1);
subplot(3, 1, 1), plot(lms);
grid on
title('Log magnitude spectrum of /u/');

subplot(3, 1, 2), plot(cep);
grid on
title('Cepstrum of /u/');

subplot(3, 1, 3), plot(smoothed);
grid on
title('Smoothed cepstrum of /u/');


% Find the /sh/ frame and perform a frequency domain analysis
[sdata, fsample] = ph_sh(filename);
[lms, cep, smoothed] = fd_analyse(sdata, fsample, framesize);

% Plot all of this
figure(2);
subplot(3, 1, 1), plot(lms);
grid on
title('Log magnitude spectrum of /sh/');

subplot(3, 1, 2), plot(cep);
grid on
title('Cepstrum of /sh/');

subplot(3, 1, 3), plot(smoothed);
grid on
title('Smoothed cepstrum of /sh/');
```

Code: questionTwo.m

```
function [ sdata, fsample ] = ph_u( filename )

% The framesize is 512
framesize = 512;

% Read in the wav file again
numsamples = wavread(filename, 'size');
[samples, fsample, nbits] = wavread(filename, numsamples(1));

% I have imperically determined that the following frames are used for /u/
start = 6 * framesize;
finish = (12 * framesize) - 1;

% This can be verified by plotting the /u/ sound. The hints for the as-
signment say that for /u/
% we should have periodicity of about 100Hz -- therefore I generate a graph with a pe-
riod of 100Hz
```

```
%plot(samples(start:finish))
%grid on

% Therefore, a frame in the center is 8 * 512 to 9 * 512
sdata = samples(start:finish);
```

Code: ph_u.m

```
function [ sdata, fsample ] = ph_sh( filename )

% The framesize is 512
framesize = 512;

% Read in the wav file again
numsamples = wavread(filename, ’size’);
[samples, fsample, nbits] = wavread(filename, numsamples(1));

% I have imperically determined that the following frames are used for /sh/
start = 70 * framesize;
finish = (74 * framesize) - 1;

% This can be verified by plotting the /sh/ sound. The hints for the as-
signment say that for /sh/
% the plot exhibits non-periodicity and a high-frequency "ragged" waveform
%plot(samples(start:finish))
%grid on

% Therefore, a frame in the center is 8 * 512 to 9 * 512
sdata = samples(start:finish);
```

Code: ph_sh.m

```
function [ lms, cep, smoothed ] = fd_analyse( sdata, fsample, framesize )

% Generate a hamming window and then apply that window
hwind = hamming(framesize);
fcount = 1;
while fcount < framesize
    hammed(fcount) = sdata(fcount) * hwind(fcount);

    % Next frame
    fcount = fcount + 1;
end

% Perform a FFT
fdata = abs(fft(hammed));

% Square each value and then perform a log
fcount = 1;
while fcount < length(fdata)
    lms(fcount) = 20 * log10(fdata(fcount));

    % Next frame
    fcount = fcount + 1;
end

cep = real(ifft(lms));

% Now we need the smoothed cepstrum
tcut = fsample / 400;
scep = cep;
for i = tcut:(framesize + 2 - tcut)
    scep(i) = 0;
end

smoothed = real(fft(scep));
```

Code: fd_analyse.m

/u/ plot

Please refer to the next page for this plot.

/sh/ plot

Please refer to the page following that for the /sh/ plots.