TTT4110 - PROJECT

By
Kristoffer K. Larsen
Magnus Skeide
Martin Ervik

Spring 2014

PART 1 - THEORY

- Generating the DTMF-signals from the given table
- Play the generated numbers from an arbitrary number

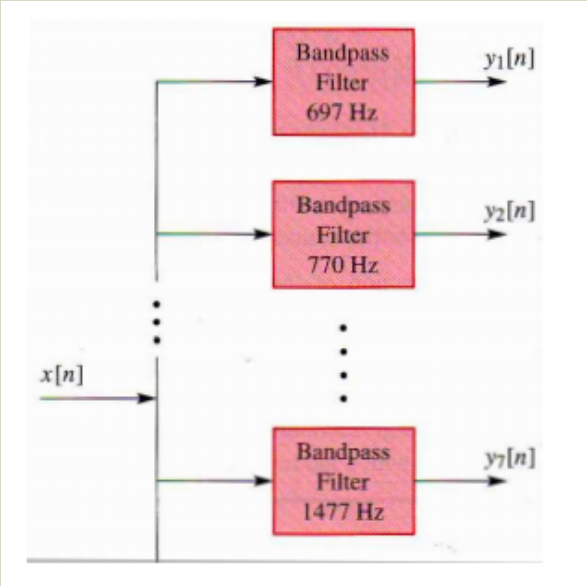
Freqs/Hz	1209	1336	1477
697	1	2	3
770	4	5	6
852	7	8	9
941	*	0	#

PART 1 - CODE

```
function returnArray = main_part1()
T = 0.2; % tid
Fs = 8000; % Hz Punktpr?vingsfrekven
t = 0:1/Fs:T; % Sample vektor
z = 0:1/Fs:.05; % Zero Vektor
returnArray = [];
number = input('Enter the characters you wish to generate DTMF dailing frequncies: ','s');
userInput = input('Do you wish to hear the frequencies genereted by your number? (y/n) ','s');
if userInput=='y';
    for x = 1:numel(number)
        switch number(x)
            case '1'
                returnArray = [returnArray (sin(2*pi*697*t) + sin(2*pi*1209*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
                case '2'
                returnArray = [returnArray (sin(2*pi*697*t) + sin(2*pi*1336*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '3'
                returnArray = [returnArray (sin(2*pi*697*t) + sin(2*pi*1477*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '4'
                returnArray = [returnArray (sin(2*pi*770*t) + sin(2*pi*1209*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '5'
                returnArray = [returnArray (sin(2*pi*770*t) + sin(2*pi*1336*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '6'
                returnArray = [returnArray (sin(2*pi*770*t) + sin(2*pi*1477*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '7'
                returnArray = [returnArray (sin(2*pi*852*t) + sin(2*pi*1209*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '8'
                returnArray = [returnArray (sin(2*pi*852*t) + sin(2*pi*1336*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '9'
                returnArray = [returnArray (sin(2*pi*852*t)+ sin(2*pi*1477*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '0'
                returnArray = [returnArray (sin(2*pi*941*t) + sin(2*pi*1336*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '*'
                returnArray = [returnArray (sin(2*pi*941*t) + sin(2*pi*1209*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            case '#'
                returnArray = [returnArray (sin(2*pi*941*t) + sin(2*pi*1477*t))];
                returnArray = [returnArray (0*sin(2*pi*697*z))];
            otherwise
                disp([number(x) ' is not a number'])
        end
    end
    sound(returnArray);
end
```

PART 2 – THEORY

- Decoding the DTMF-signals
- The input is now a signal
- We want the output to be the numbers
- To do so we must:
  - Create filters
  - Split up signal
  - Recognize the numbers in an iterative manner



PART 2 - CODE

- Variables for use throughout the code
- Checking length of signal

```
function decodedNumber = main_part2(ToneGeneratedPart1)
    %% Variabler som brukes gjennom hele programmet

    Fs = 8000; % samplingsfrekvensen
    decodedNumber = []; % Telefonnummeret som vil bli returnert

    %% Fjerner pause delen i signalet

    toneLenght = Fs*0.2; %Lengden av en tone er samplefrekvens * tid
    pauseLenght = Fs*0.05; %Lengden av pausen er samplefrekvens * tid
    numbersToFilter = round(length(ToneGeneratedPart1)/(toneLenght + pauseLenght)); %Finner ut hvor mange nummer signalet best?r av
    toFilter = [];
    start = 1;
    for n = 1:numbersToFilter
        toFilter = [toFilter (ToneGeneratedPart1(start:start + toneLenght - 1))];
        start = start + toneLenght + pauseLenght;
    end
```

CODE CONTINUES

- Creating the filters

```
%% Lager Båndpassfiltrene som blir brukt til å identifisere tallene

Frekvenser = [697 770 852 941 1209 1336 1477]; % De forskjellige frekvensene i bruk i DTMF
L = 300; % Bredden på båndet til båndpassfilteret
n = 0:L-1;
filters = [];
for i = 1:length(Frekvenser)
    f0 = Frekvenser(i)/Fs; %Digital frekvens
    w0 = f0 * 2 * pi; %Vinkelfrekvens
    bk = (2 * cos(w0 * n)/L); % Amplitude

    filters  = [filters; bk];
    [h, w] = freqz(bk, 1, L);
    plot((w * Fs / (2 * pi)), abs(h));
    hold on
end
title('Filter');
xlabel('Frekvens');
ylabel('|H(w)|');
```

CODE CONTINUES

- Separating and decoding each “piece” of the input signal

```
%% Deler opp signalet til hvert enkelt tall og Dekoder input signalet ved bruk av Båndpassfilterene tidligere laget

for k = 1:(length(toFilter)/(Fs*0.2))
    NumberToFilter = toFilter((k-1)*(Fs*0.2)+1:k*(Fs*0.2));
    threshold = 0.8;
    % Siden filteret ikke er perfekt, setter vi grensen på 0.85.
    % Hvis vi får et treff høyere enn 0.85 så antar vi derfor at
    % frekvensen finnes.
    temp = []; %Liste for å lagre frekvensene vi finner.
    for n=1:7
        y = filter(filters(n, 1:L), 1, NumberToFilter); %Filtrerer vektoren part gjennom filteret
        if(max(y) > threshold) % Hvis amplituden på signalet er høyere enn thresholdet, så registreres det.
            temp = [temp n]; % Legger til frekvenstallet i listen
        end
    end
```

Summing the frequencies for the final result

```
%% Summerer frekvensene og finner ut hvilke sum som hører til hvilket tall
sum = Frekvenser(temp(1)) + Frekvenser(temp(2)); %Henter ut de to dekodede frekvensene fra parten og Summerer de to frekvensene
%Finner ut hvilken sum som tilhører hvilket tall
switch sum
    case 1906
        decodedNumber = [decodedNumber '1'];
    case 2033
        decodedNumber = [decodedNumber '2'];
    case 2174
        decodedNumber = [decodedNumber '3'];
    case 1979
        decodedNumber = [decodedNumber '4'];
    case 2106
        decodedNumber = [decodedNumber '5'];
    case 2247
        decodedNumber = [decodedNumber '6'];
    case 2061
        decodedNumber = [decodedNumber '7'];
    case 2188
        decodedNumber = [decodedNumber '8'];
    case 2329
        decodedNumber = [decodedNumber '9'];
    case 2150
        decodedNumber = [decodedNumber '*'];
    case 2277
        decodedNumber = [decodedNumber '0'];
    case 2418
        decodedNumber = [decodedNumber '#'];
end
```

- Takk for oss!