DSP MATLAB Project Report

노이즈 캔슬링, 주파수 대역 분석을 이용한

“재난 현장에서 사람 목소리 탐지를 통한 구조 보조 프로그램”

2019 년 12 월 08 일

과제 수행팀 :32151648, 박동학

32172572, 오준희

목차

1. Project 개요

1.1 Project 개발 필요성

사고 현장에서는 인명 피해가 발생하고 이를 최대한 줄이기 위해서 구조 인력이 투입된다. 이러한 화재, 재난 현장에서는 많은 소음이 발생한다. 인명 구조 작업을 진행하는데 있어서 소방관과 구조 인력이 투입되기 전 현장에 구조 되어야 할 사람의 존재를 판단하는 것은 현장에 뛰어 들어가는 소방관과 구조인력에게 정보를 제공할 수 있다.

1.2 Project 개발 배경

부산 해운대의 101층 빌딩 엘시티에서 건물 꼭대기에 화재가 발생했을 때 외부 사다리가 닿지 않고, 비상 발전기도 모두 먹통이 되었을 때를 가정한 화재 진화 훈련이 진행 됐다. 소방관들은 20kg 이 넘는 장비를 착용하고 계단을 오르며 층마다 일일이 확인해야 했고, 꼭대기까지 올라가는 데 걸린 시간은 평균 41분이 소요되었다. 따라서 재난 현장에서 본 프로젝트의 결과물을 이용한다면 층마다 일일이 확인 하지 않아도 되기 때문에 사람을 구조하는데 있어서 소요되는 시간을 줄일 수 있고, 위험한 현장에 뛰어드는 소방관과 구조인력의 안전 또한 확보할 수 있게 될 것이라고 판단된다.

1.3 기본적인 동작 및 기능

소리가 입력되며 Noise Filtering을 통해서 사람의 소리에서 발생가능한 주파수 대역인 50~4000Hz 대역을 계산한다. 처리된 Time-Domain의 소리를 FFT를 통해 Frequency Domain 으로 옮기고 이를 분석하여 사람이 재난 현장 속에 있는지 유무를 파악한다.

1.4 Project 개발 방법

a. 참고 MATLAB project 출처

1. 음성 처리 : 음성 파일(.wav) 녹음 및 정보 확인, 노이즈 제거, 높낮이, 발음 구분하기 (https://honeyjamtech.tistory.com/category/Matlab)
2. Signal Processing Toolbox Guide, Audio Toolbox Guide (https://kr.mathworks.com/help/)

b. 수정 및 개선 필요성, 범위 및 방법

우리 팀은 현재 매트랩에서 제공하는 기본적인 기능들을 조합하여 실제로 필요한 서비스를 구현하는데 초점을 맞추었다. 전체적으로 완성된 프로그램을 수정하는 것이 아닌 제공되는 여러 기능들을 통해서 새로운 프로그램을 작성한다.

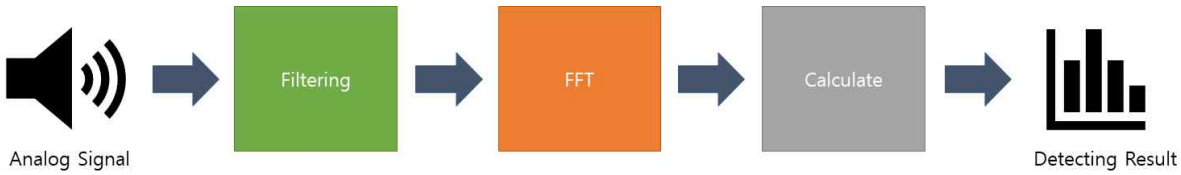
c. Project에서 본인들의 기여도 %로 측정 및 근거 제시

박동학 : 프로젝트의 알고리즘 구성을 주로 담당, 프로젝트에 사용되는 기술적인 부분을 탐색하고 매트랩 구현에 기여하였으며, 기여도는 60%로 판단된다.

오준희 : 본 프로젝트의 주제 결정 및 배경, 개요작성 및 매트랩 프로그램 탐색, 구현 등에 기여하였으며, 기여도는 40%로 판단된다.

2. Project 수행 내용 (5페이지)

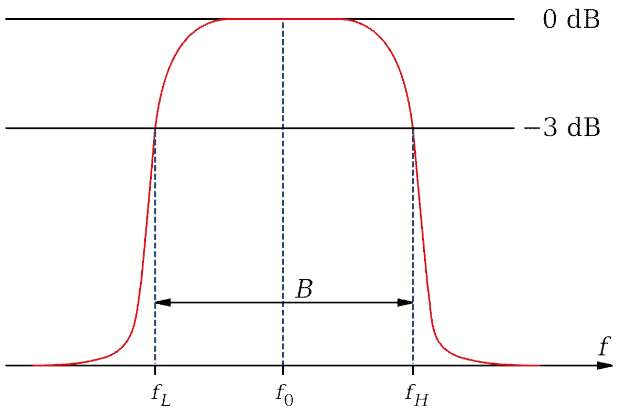
2.1 알고리즘 설명



<Figure 1>

이 프로젝트에서는 Analog 신호를 마이크로 입력 받으면 이를 사람이 내는 소리의 주파수인 50~4000Hz로 Band Pass Filter를 통해서 필터링을 하여 나온 결과물을 FFT를 통해서 주파수 영역으로 변환시킨다. Frequency-Domain으로 변환된 신호 계산을 통해서 사람 소리 주파수 대역의 신호가 얼마나 많이 발생했는지를 통해서 사람이 있는지 가능성을 알려준다. 주요하게 사용 되는 알고리즘은 아래와 같이 Band Pass Filter와 FFT이며 이를 통해서 나온 결과 값을 counting 하여 사람의 존재 유무를 알려주는 간단한 Calculate.m이 있다.

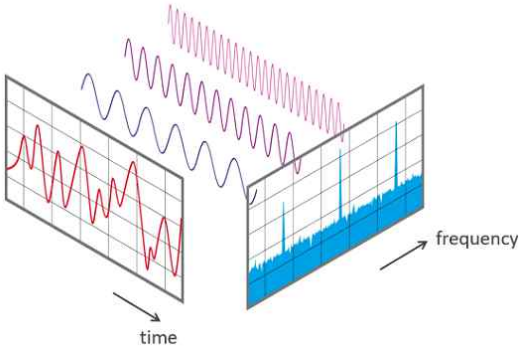
2.1.1 Band Pass Filter



< Figure 2 >

Band Pass Filter란 특정 영역의 주파수만 통과시키고 나머지 주파수는 없애는 것을 이야기 한다. 이러한 특정 주파수 영역을 Band라고 지칭하고 이 Band만 통과시키겠다는 것이다. Digital Signal Processing에서는 이러한 BPF를 간단한 연산으로 수행 가능하다. 필요한 주파수 영역만을 원래 값이나 증폭된 값으로 남기고 나머지 값들을 제거하여 저장하면 된다. 하지만 이러한 필터는 완전하게 그 부분만 통과 시키는 것은 불가능하기 때문에 어느 정도의 영역은 남아 있게 된다.

2.1.2 Frequency Domain, FFT

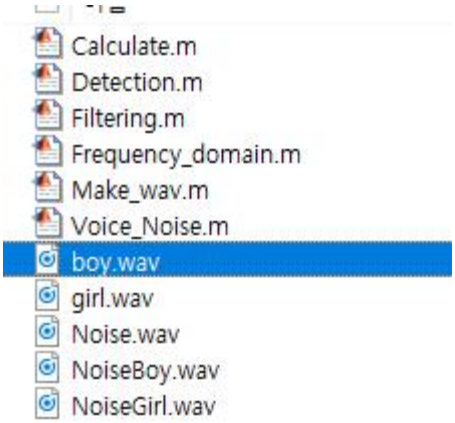


< Figure 3 >

아날로그 신호는 시간과 진폭으로 표현될 수 있다. 하지만 실제 발생하는 신호의 경우 높은 주파수 성분을 가지고 있기 때문에 시간과 진폭만으로는 유의미한 정보를 얻을 수 없는 경우가 많고 사람이 직관적으로 이해하기 쉽지 않다. 따라서 주파수 영역 (Frequency Domain)으로 변환하여 활용하는 경우가 많다. 이러한 변환은 특정 신호의 진동수를 기준으로 특정 진동수의 Magnitude를 나타낸다. 주파수영역으로 이동한 신호는 다양한 정보를 얻을 수 있으며 이 프로젝트에서는 사람의 음성 주파수 대역이 50 ~ 4000Hz 임을 이용한다.

2.2 MATLAB 구현

이번 프로젝트에서는 재난 현장에 소리를 직접 수집하는 것과 여러 가지 채널을 가지고 있는 마이크를 활용하기 힘들었기 때문에 사람의 목소리와 노이즈의 경우를 따로 녹음해서 직접 합성해서 프로젝트를 수행할 신호를 생성했다. 이 프로젝트에 사용된 MATLAB 함수들은 다음과 같다.



< Figure 4 >

Make_wav.m

- 매트랩의 기능을 이용하여 남자, 여자, 소음을 각 5초간 16bit 48000Hz로 녹음하여 wav 파일로 저장한다.

Voice_noise.m

– 생성된 파일들을 합성하여 노음이 있는 음성 파일을 생성한다.

Filtering.m

– 사람의 음성을 특정 주파수만 통과 할 수 있도록 Filtering 한다.

Frequency_Domain.m

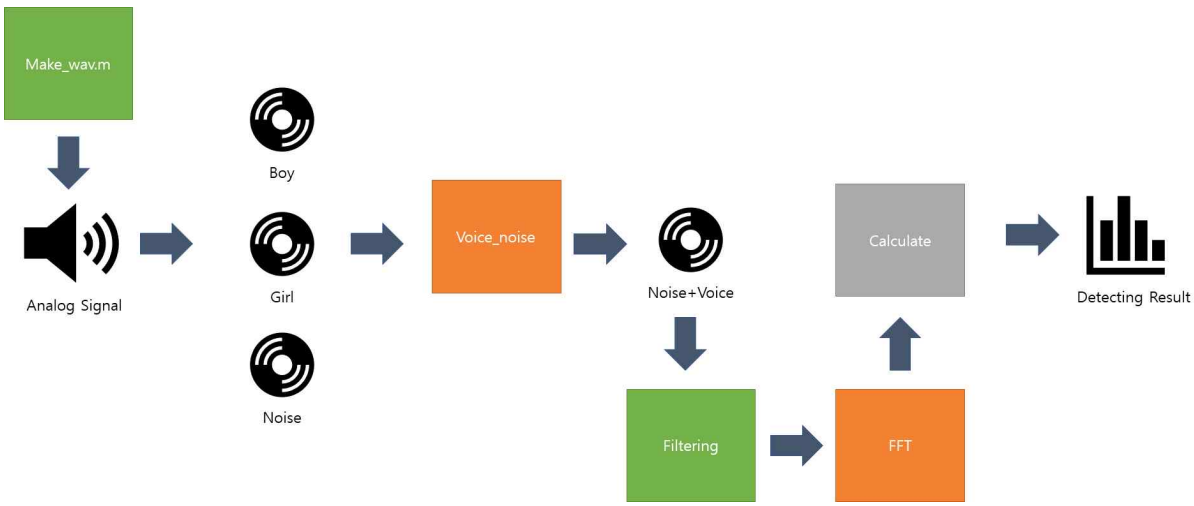
– 전처리된 데이터를 FFT를 통해서 주파수 영역으로 변환한다.

Calculate.m

– FFT로 주파수 영역으로 변경 된 신호 중에 사람 음성 주파수 대역에 해당하는 주파수의 Magnitude가 일정 수치 이상인 신호를 count 하여 일정 수준 이상이라면 사람이 있을 가능성을 알려준다.

Detection.m

– 작성한 기능들을 한번에 수행 할 수 있도록 명령어들을 모아 놓았다.

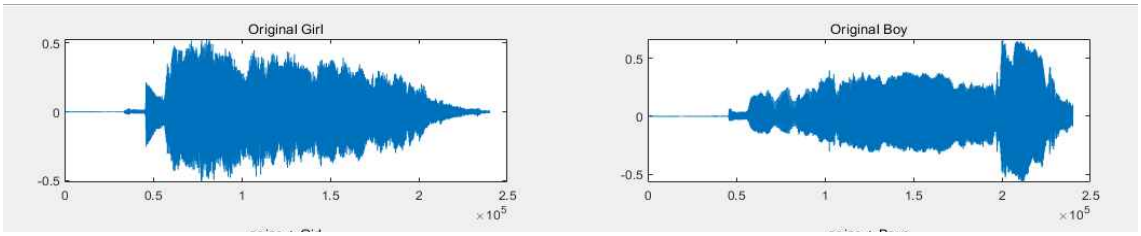


< Figure 5 >

입력으로 소리가 들어가면 출력으로 사람이 있는지 유무가 나오게 되는 프로그램 구성도 이다.

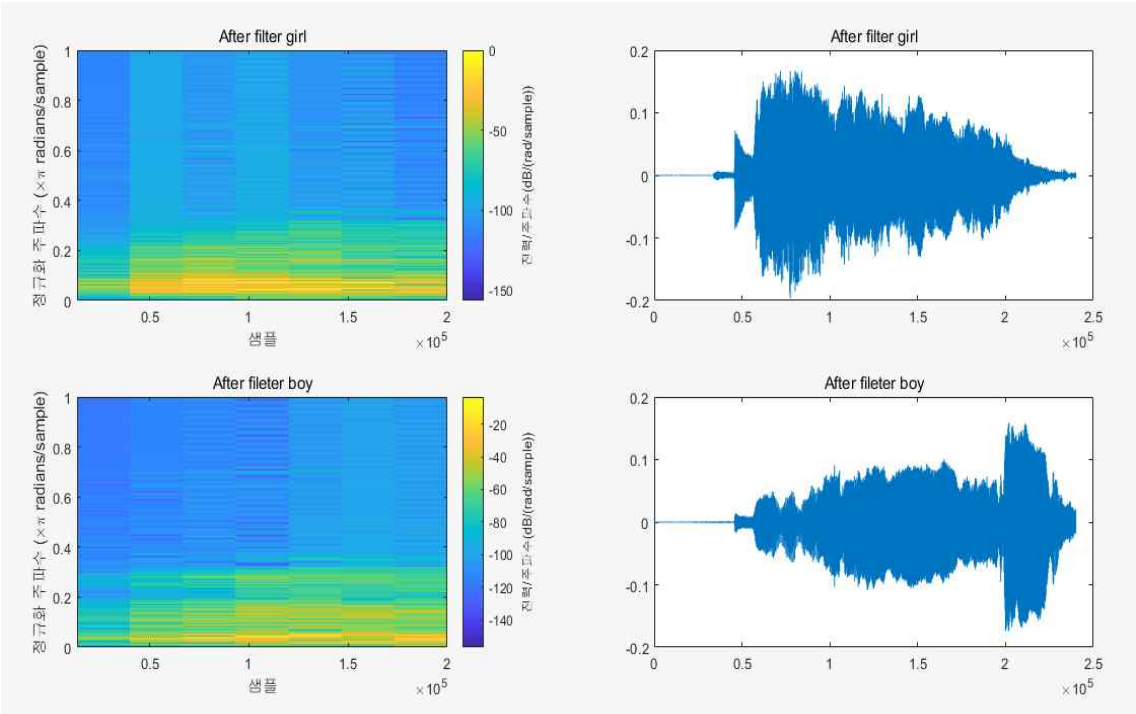
2.3 입력 및 출력

입력 : 남자와 여자의 음성 신호



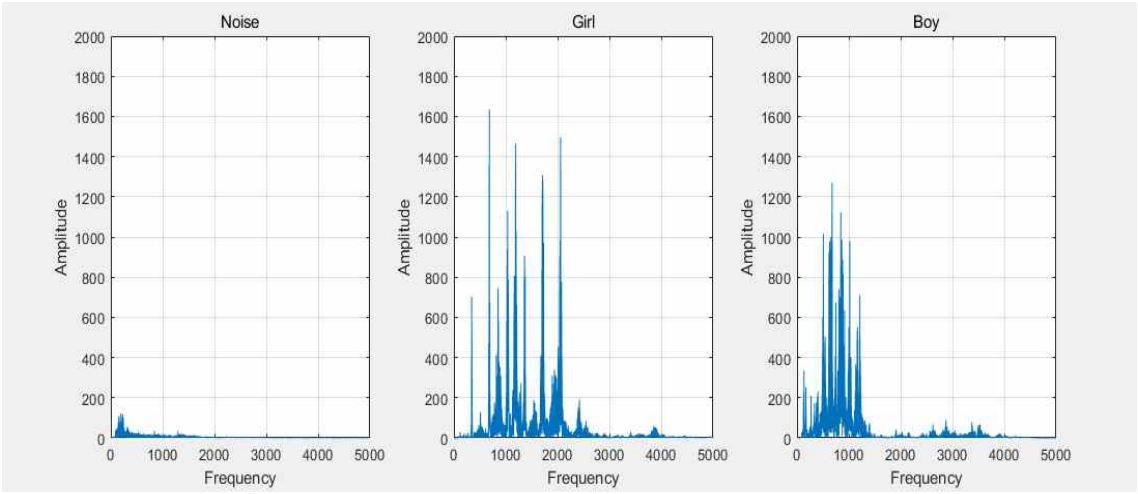
< Figure 6 >

노이즈와 합성한 음성 신호



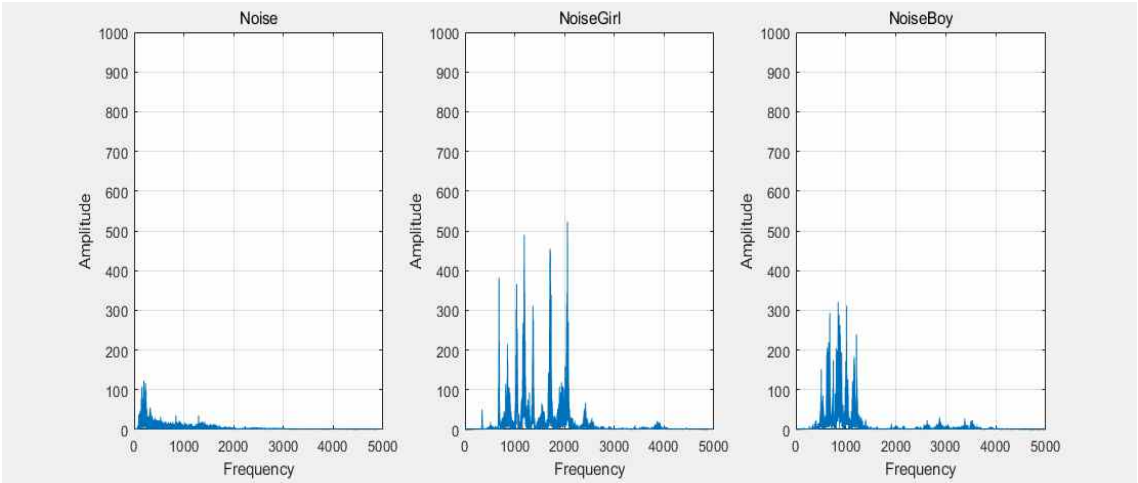
< Figure 7 >

필터링 되기 전 후의 Frequency Domain Plot



< Figure 8 >

필터링 되고 난 후의 Frequency Domain Plot



< Figure 9 >

출력 : 사람이 있는지 없는지 유무

```
>> Detection

ans =

    "여성의 음성으로 추정되는 주파수 대역의 소리 감지"


ans =

    "남성의 음성으로 추정되는 주파수 대역의 소리 감지"

경과 시간은 7.249783초입니다.
```

< Figure 10 >

사람의 음성이 같이 있는 소리에 대해서 사람이 있음을 알려줌

3. 참여 인원별 역할 및 수행 소감

박동학 : 처음 매트랩을 접해서 특정 기능을 구현하는 프로그램을 작성하는데 있어서 익숙하지 않아 빠르게 할 수 없었지만 시간이 지남에 따라 다른 프로그래밍 언어에 비해서 특정 기능들을 편리하게 사용 할 수 있는 많은 tool들이 준비되어 있어 아이디어를 생각하는 즉시 적용해 볼 수 있어서 좋았다. 시간이 부족해서 Localization에 필요한 장비를 구하지 못하고 실제로 재난 현장 소리를 구해 기본적인 동작만 하는 프로그램을 만들었지만 차후에 계획서에 제출한 대로 Localization을 적용하여 위치까지 파악할 수 있는 프로그램을 작성해 보고 싶다.

오준희 : 사람의 목소리를 인식하는 프로그램을 매트랩을 이용해 직접 구현할 수 있다는 점이 흥미로웠고, 관련된 내용의 자료를 검색했을 때 많은 자료들을 찾아볼 수 있기 때문에 구현하는데 많은 도움이 되었다. 또한 본 프로젝트를 구현하는 데 있어서 더 많은 시간이 주어졌더라면 더 완성도 있는 결과물이 나올 수 있었을 것이라고 생각한다. 본 프로젝트의 주제는 재난 현장에서 사람의 목소리를 인식해 사람을 구조하는데 도움이 되는 프로그램이었는데 이처럼 사람이 살아가는데 도움이 되는 많은 프로그램들이 개발 되었으면 한다.

4. Project 후기

우리 팀은 이번 프로젝트의 목표를 매트랩을 이용한 실질적인 응용프로그램 작성으로 설정하고 프로젝트를 진행했다. 이러한 목표를 설정한 것은 매트랩은 다양한 테스트를 짧은 시간 내에 수행할 수 있고 시각화 하는 다양한 방법을 제시해 주며 이미 MathWork에서 다양한 예시와 라이브러리를 제공하기 때문에 직접 프로그램을 작성하기 보다는 원리를 이해하고 이를 조합해서 서비스를 작성하는 것이 의미 있는 프로젝트라고 생각해서이다. 프로젝트를 수행하는데 있어서 음성 신호를 사용하면서 수업시간에 배운 다양한 내용들을 직접 수행해보고 이를 적용해서 일종의 결과를 도입함으로 우리가 배우고 있는 이론적인 내용이 실제 작업에서 어떻게 활용되는지 왜 필요한지에 대해서 느낄 수 있는 프로젝트였다.

다만 시간이 부족해서 처음에 기획했던 Sound_Localization과 소리 증폭을 통한 세심한 신호를 수집하는 기능 등 모든 기능을 구현하지 못하고 최소 2개 이상의 채널이 필요로 하는 하드웨어적인 제약과 실제 사고현장의 소리를 구할 수 없어 임의로 신호를 생성해서 이론적인 내용을 중심으로 프로그램을 작성하고 실험을 했기 때문에 실제적인 실험을 통해서 이 프로그램이 실제 필드에서는 어떻게 작용하는지를 확인하지 못한 것이 아쉽다.

5. Appendix (Project MATLAB source code)

Make_wav.m : 임의로 신호를 생성

```
% 현장의 소리 수집 및 신호로의 변환

recorder1 = audiorecorder(48000,16,1); % sampling rate = 48000Hz, Quantization
Bit = 16 bit, 1개의 채널

disp("Start");
recordblocking(recorder1,5); %5초간 녹음 하겠다는 의미
disp("End");

filename1 = 'boy.wav';
y1 = getaudiodata(recorder1);

audiowrite(filename1, y1, 48000);
[readY, Hz] = audioread(filename1);

figure(1);
subplot(211); plot(readY)
subplot(212); spectrogram(readY, 'yaxis')
// 실제 현장 소리 대신에 임의로 생성한 소리를 사용하여 실험 하였음
```

Voice_Noise.m : 음성과 잡음을 합성하여 신호 생성

```
clear
[x1, Fs1] = audioread('girl.wav'); % 여자 음성
[x2, Fs2] = audioread('boy.wav'); % 남자 음성
[x3, Fs3] = audioread('Noise.wav'); % 소음 ( 현장 소음 )

k = 0.65; //0.65로 하였을 때 가장 적합한 신호를 얻을 수 있었음
new1 = zeros(240000,1);
new2 = zeros(240000,1);

for i = 1:1:240000
    new1(i) = ( (1-k)*( x1(i)+k*(x3(i)) ) );
    new2(i) = ( (1-k)*( x2(i)+k*(x3(i)) ) );
end //잡음을 추가하여 신호를 새로 생성

filename1 = 'NoiseGirl.wav';
filename2 = 'NoiseBoy.wav';
audiowrite(filename1, new1, 48000);
audiowrite(filename2, new2, 48000);
```

```
%%%%%%%%%%%%%%%%%% 시각화 부분 %%%%%%%%%%%%%%%%%%%%
figure(1)
subplot(221)
spectrogram(x1, 'yaxis')
title('Original Girl')

subplot(222)
spectrogram(x2, 'yaxis')
title('Original Boy')

subplot(223)
spectrogram(new1, 'yaxis')
title('noise + Girl')

subplot(224)
spectrogram(new2, 'yaxis')
title('noise + Boys');
%%%%%%%%%%%%%%%%%%%%
figure(2)

subplot(221)
plot(x1)
title('Original Girl')

subplot(222)
plot(x2)
title('Original Boy')

subplot(223)
plot(new1)
title('noise + Girl')

subplot(224)
plot(new2)
title('noise + Boys');
```

Filtering.m : 50~4000HZ 만 통과시키는 필터링

```
[b,a] = butter(2, [0.05, 0.4], 'bandpass'); //50~4000Hz 사이만 통과
freqz(b,a)

aftfil1 = filter(b,a,new1);
aftfil2 = filter(b,a,new2);
```

```
figure(1)
subplot(221); spectrogram(aftfil1, 'yaxis')
title('After filter girl')
subplot(222); plot(aftfil1)
title('After filter girl')

subplot(223); spectrogram(aftfil2, 'yaxis')
title('After fileter boy')
subplot(224); plot(aftfil2)
title('After fileter boy')
```

Frequency_domian.m : 신호를 주파수 영역으로 옮긴다.

```
fs = 48000;

Noise = audioread('Noise.wav');
girl = aftfil1;
%audioread('NoiseGirl.wav');
boy = aftfil2;
%audioread('NoiseBoy.wav');

%Noise : 소음을 FFT
N_N = length(Noise);
F1=0:fs/N_N:fs-fs/N_N;
FFT_Noise =fft(Noise);

%Girl : 여자 목소리를 FFT
N_G = length(girl);
F2=0:fs/N_G:fs-fs/N_G;
FFT_Girl =fft(girl);

%Boy : 남자 목소리를 FFT
N_B = length(boy);
F3=0:fs/N_B:fs-fs/N_B;
FFT_Boy =fft(boy);

figure;

subplot(131);
plot(F1(1:N_N),abs(FFT_Noise(1:N_N)));
xlabel("Frequency"); ylabel("Amplitude");
title('Noise');grid on; axis([0 5000 0 1000]);
```

```
subplot(132);
plot(F2(1:N_G),abs(FFT_Girl(1:N_G)));
xlabel("Frequency"); ylabel("Amplitude");
title('NoiseGirl');grid on; axis([0 5000 0 1000]);

subplot(133);
plot(F3(1:N_B),abs(FFT_Boy(1:N_B)));
xlabel("Frequency"); ylabel("Amplitude");
title('NoiseBoy');grid on; axis([0 5000 0 1000]);
```

Calculate.m : 주파수영역의 신호를 카운트해서 사람이 있는지 추측

```
girl_cal = abs(FFT_Girl(1:N_G));
boy_cal = abs(FFT_Boy(1:N_B));
counter_G = 0;
counter_B = 0;

for i = 1:1:4000
    if(girl_cal(i) > 100 )
        counter_G = counter_G+1;
    end
end

for j = 1:1:4000
    if(boy_cal(j) > 100 )
        counter_B = counter_B+1;
    end
end

if( counter_G > 50 )
    sprintf("여성의 음성으로 추정되는 주파수 대역의 소리 감지")
end

if ( counter_B > 50)
    sprintf("남성의 음성으로 추정되는 주파수 대역의 소리 감지")
end
```

Detection.m : 수행해야 하는 기능들을 Detection을 통해서 한번에 실행

```
Voice_Noise;
Filtering;
Frequency_domain;
Calculate;
```