**SPL-1 Project Report, 2019**

**Fourier Transform on Sound Processing**

**Course: Software Project Lab**

**Course No: SE 305**

Submitted by

**Md. Eashaque Rahman Sami**

**Roll No : BSSE1033**

**Session: 2016-2017**

Supervised by

**Dr. Ahmadul Kabir**

**Institute of Information Technology**

**University of Dhaka**

**Submission Date: 29-05-2019**

Introduction:

Fourier transform is a transform between time domain and frequency domain. It decomposes a signal into sinusoids. So, it is known as the frequencydomainrepresentation of the original signal. Now-a-days we use this transform in various fields for example – digital sound processing (DSP), Image Processing etc.

We know that sound is a combination of different sine and cosine wave. Every wave has its own frequency. By using Fourier transform we can transform the signal from time domain to frequency domain and get the frequencies.

**1.1 Background Study**

**Fourier transform (FT)**

Fourier transform is a transformation between time domain and frequency domain.

The Fourier transform of a function of time is itself a complex- valued function of frequency, whose magnitude represents the amount of that frequency present in the original function.

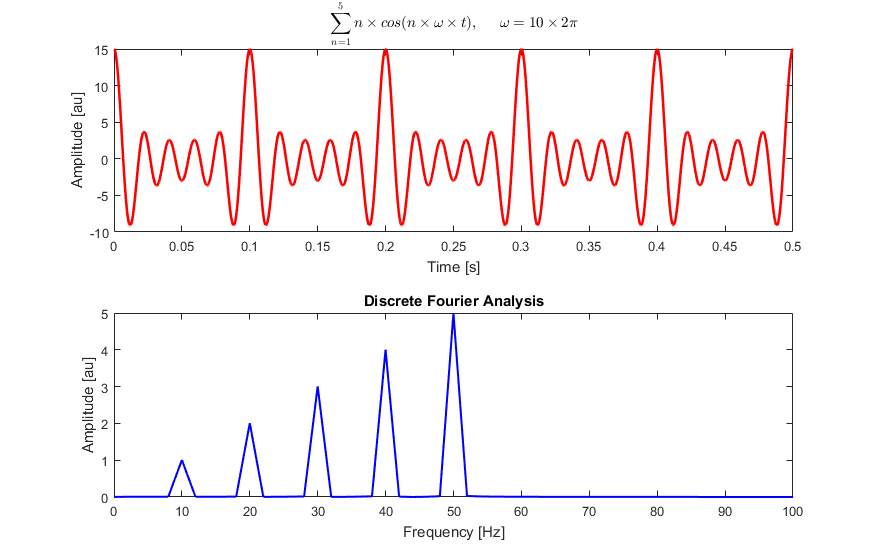


Figure 1: FT on a mixed frequency wave

The image is a sound wave of 10, 20, 30, 40 and 50 hz. When we do FT on it we can see that in frequency domain amplitude is high at 10, 20, 30, 40 and 50 hz. In

this way FT tells us about the frequency of a time domain signal.

**First Fourier Transform (FFI)**

A first Fourier transform (FFT) is an algorithm which computes the discrete Fourier Transform (DFT) of a sequence. FFT is based on divide and conquer algorithm where we divide the signal into two smaller signals, compute the DFT of the two smaller signals and join them to get the DFT of the larger signal. The complexity of DFT is O(n^2) while that of FFT is O (n log n). So FFT is faster than DFT.

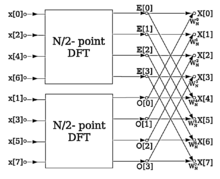


Figure 2: FFT algorithm structure

**Sampling Theorem**

For a DFT to represent a function accurately, the original function must be sampled at a sufficiently high rate. The appropriate rate for a uniformly sampled time series is determined by the Nyquist Sampling Theorem.

The Nyquist Sampling Theorem states that: A bandlimited continuous-time signal can be sampled and perfectly reconstructed from its samples if the waveform is sampled over twice as fast as its highest frequency component. We can describe this as bellow:

fs >= 2 fc

here, fs is sampling frequency

fc is highest frequency contained in the signal

**wav file handling**

1. Reading wav file header
2. Sampling wav file
3. Write a wav file

**Digital filter**

We use different types of digital filters to remove noise from a wav file. There are different types of digital filters. For example:

1. High pass filter:  allowing higher frequencies to pass through the filter.
2. Low pass filter:  allowing lower frequencies to pass through the filter.
3. Band pass filter: passes frequencies within a certain range and rejects frequencies outside that range.

1.2. Challenges

For implementing this project there are lot of challenges that I have faced. Some of them are:

1. Handling large code for the first time
2. Learning and understanding algorithm
3. Implement FFT and IFFT algorithm
4. Reading and sampling wav file
5. Implement low pass filter
6. Reconstruct new signal

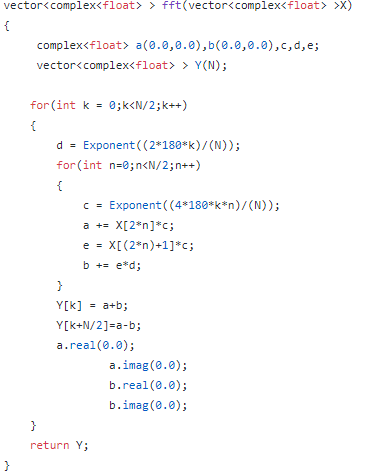
**2. Project Overview**

I have divided my whole project into three different parts. They are

1. Implement FFT and IFFT
2. Reading and sampling wav file
3. Removing noise from wav file
4. Reconstruct the wav file

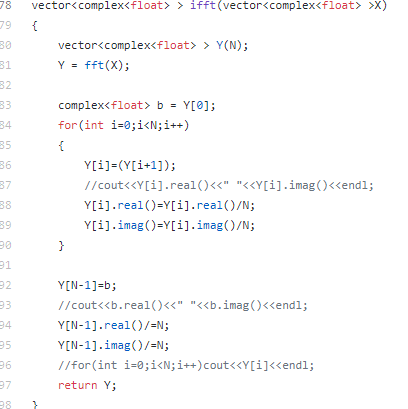
**2.1 Implementation of FFT algorithm**

This is the key part of my project. The input is an array of N samples where the samples can be real or complex. The output is a complex array contains real and imaginary part. For implementing FFT on sound file I have to handle a huge data. So I use vector instead of array. The sample code for FFT is given bellow:



**Implementation of IFFT Algorithm**

Here we call the fft ( ) function.



**2.2 Reading wav file header**

Here we read the header of a .wav file. By reading the header we can learn about different characteristics of the file. For example, we can learn about chunk size, sample rate, number of channels, sample rate etc. From this information we can sample the wav file. For example, if our sample rate is 44100 Hz and we have a 5 second wav file then the total number of samples will be 44100\*5 = 220500.

Again, we can learn if the wav file is stereo or mono by the number of channel(s).

A wav file format is given bellow:

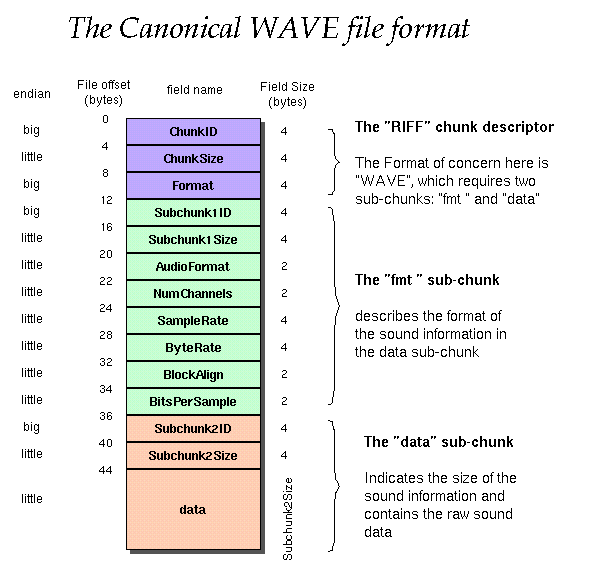


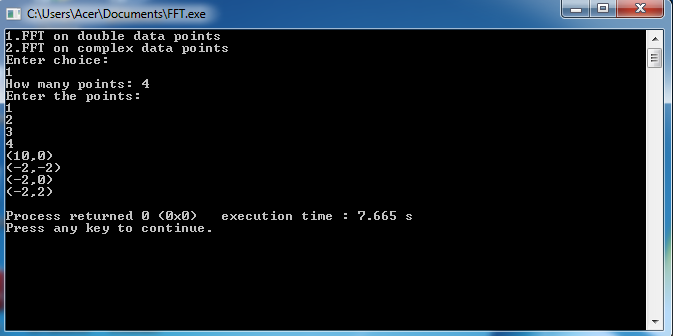
Figure: wav file format

**2.3 Implement digital filter**

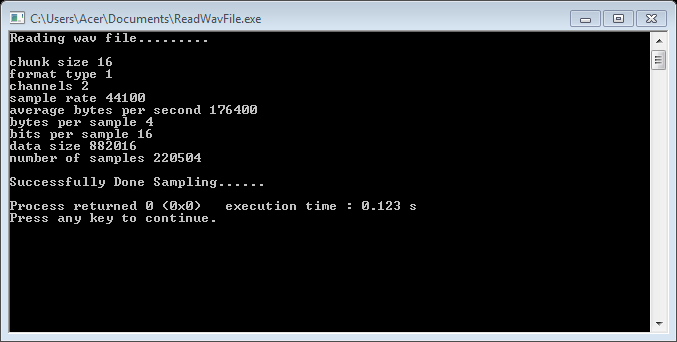
I implement 3 digital filters (HPF, LPF, BPF). They can remove noise from the signal.

**3.User manual**

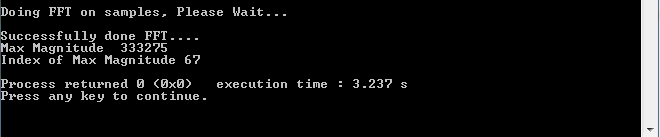
I have applied FFT for removing noise from sound. We can apply FFT on some double datapoints.



To do FFT first we read a wav file and do sampling.



After reading wav file and doing sampling we can do FFT on the samples.

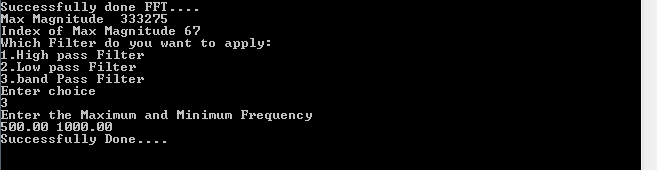


After doing FFT we can do filtering on the wav file data. Here we can apply

1.High Pass Filter

2.Low Pass Filter

3.Band Pass Filter



**4.Future Goal**

In this project I implement FFT for sound processing. In future I want to implement FFT on image processing.

**5.References**

1. <https://en.wikipedia.org/wiki/Fast_Fourier_transform>
2. <https://en.wikipedia.org/wiki/WAV>
3. <http://scistatcalc.blogspot.com/2013/12/fft-calculator.html>
4. <https://www.onlinehexeditor.com/>

**6.GitHub Link**

<https://github.com/eashaksami/SPL>