Application of Short Time Fourier Transform on Stationary and Non-Stationary Data

*[1]Gökçen Bombar and [2]Gülden Göktürk
*[1] Faculty of Engineering, Department of Civil Engineering, Ege University, Turkey
[2]Faculty of Engineering, Department of Electrical and Electronic Engineering Dokuz Eylül University, Turkey

Abstract:

The aim of the study is to compare the frequencies of stationary and non-stationary data by Short Time Fourier Transform (STFT). A rectangular window and a Gaussian window are used where the latter stands for the Gabor Transform. It is revealed that the for stationary data rectangular window and for non-stationary data Gaussian window is more appropriate to reveal the frequency content of a time series.

Key words: Short Time Fourier Transform (STFT), rectangular window, Gaussian window, Gabor Transform

1. Introduction

Non-stationary signals are usually the signals which the frequency distribution changes with time. The frequency content of this type of signals could not be observed by Fourier Transform. Fourier transform is an operation which decomposes the signal into its frequencies in other words transforms the signal from time domain to frequency domain. The handicap of this transformation is it suffers to distinguish the dominant frequencies with respect to time. The Short Time Fourier Transform (STFT) is used to analyze a signal by dividing the time series into many parts in the time domain and taking the Fourier Transform of each part. Traditionally a rectangular window is used to divide the time series. The introduction of Gaussian window improved the consequences of the sharp edges of the rectangular window. This is called the Gabor Transform [1].

2. The Set-up for Generation of the Data

The time series data is generated by the ultrasonic transducer of NXT Lego Robot (Fig. 1). The processing and data recording is done in the unit of NXT called brick and the ultrasonic sensor is mounted on the front of NXT which can measure the distance from its tip to an obstacle which is simply a card board in this case, as shown in Fig. 2. The code written by the NXT software (Fig. 3.) is transformed to the NXT brick via a cable. When the code is initiated, the NXT start oscillating 12.5 cm forward and 12.5 cm backward with an overall amplitude of 25 cm. By holding an obstacle in front of the ultrasonic sensor, it is possible for the NXT to measure the

*Corresponding author: Address: Faculty of Engineering, Department of Civil Engineering Ege University, 35100, İzmir TURKEY. E-mail address: gokcen.bombar@ege.edu.tr, Phone: +902323886026 Fax: +902323425629

distance and record at the meantime. The sampling period of the ultrasonic sensor is 0.2 s, which means 5 Hz sampling frequency. After the recording, the data is transferred in to the computer and analyzed.

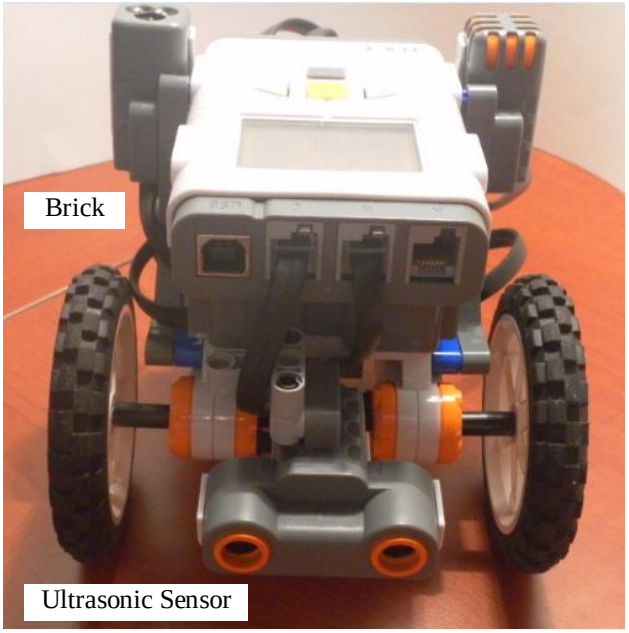

Figure 1. The NXT Lego Robot

Figure 2. The scheme of the set-up

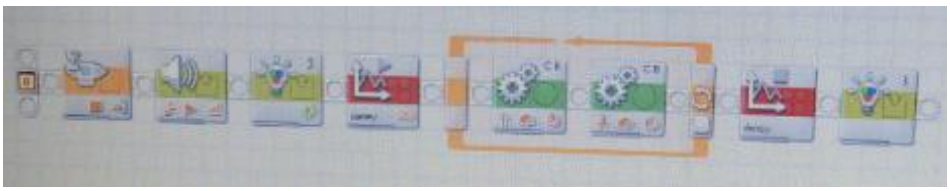

Figure 3. The code written in the NXT software

3. Generated Signal Data

Four runs are performed by using the set-up defined in the previous section. In Run-1, the NXT kept in rest and only the obstacle is oscillated manually, with a constant speed as much as possible. In Run-2, the obstacle is kept immobile and only the NXT was active. In Run-3 both NXT and the obstacle was let mobile, oscillating forward and backward with more or less the same frequencies in Run-1 and Run-2. The recorded time series which constitutes 119 data are depicted in Fig. 4.

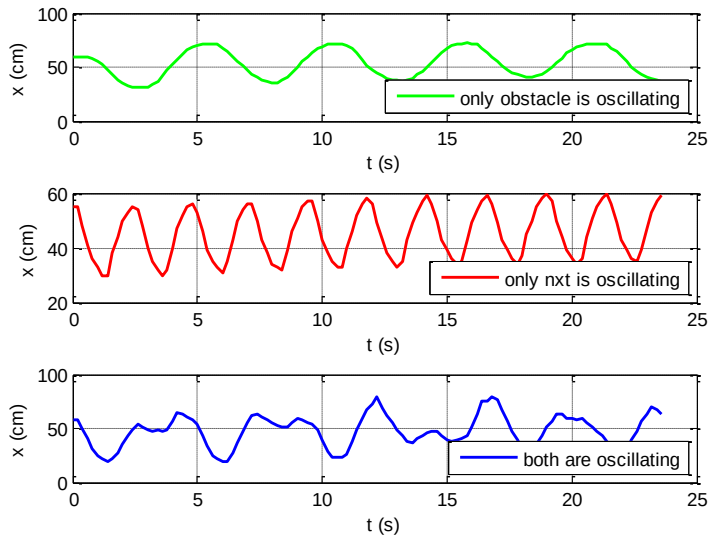

Figure 4. The recorded time series for (a) Run-1, (b) Run-2 and (c) Run-3.

Run-4 was performed by the constant frequency of NXT while a variable oscillation rate of the obstacle which is determined by manually, as depicted in Fig. 5.

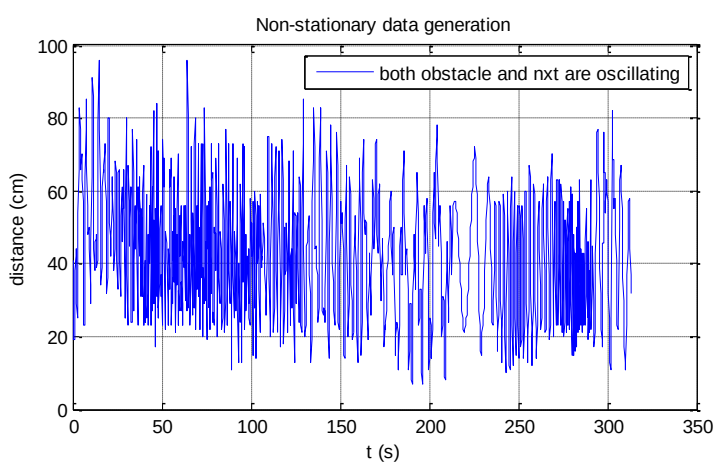

Figure 5. The recorded time series for Run-4

The characteristics of the runs are given in Table 1. Here the number of data in the time series of the runs is denoted by N. The period (s) and frequency (Hz) of the time series are named as T0 and fx, respectively. Fs is the sampling frequency of the ultrasonic sensor (Hz).

Table 1. The Characteristics of the Runs

Parameter	N	T_0	f_x	F_s
Run -1	119	5.2 s	0.19 Hz	5 Hz
Run -2	119	2.4 s	0.42 Hz	5 Hz
Run -3	119	-	-	5 Hz
Run -4	1559	-	-	5 Hz

The statistical properties of the time series are given in Table 2 and the auto-correlation coefficients calculated for lag of 1, 2, 3 and 4 are tabulated in Table 3. Minimum values are 31 cm, 30 cm, 19 cm and 7 cm for Run-1, Run-2, Run-3 and Run-4 and the maximum values are 73 cm, 60 cm, 80 cm and 96 cm for Run-1, Run-2, Run-3 and Run-4, respectively.

Table 2. Statistical Parameters of the Runs

Parameter	Mean	Std. Dev.	Skewness	Kurtosis
Run-1	53.87	13.05	-0.028	1.619
Run -2	45.24	9.06	-0.024	1.608
Run -3	47.86	14.68	-0.097	2.358
Run -4	42.65	17.08	0,238	2,338

Table 3. Auto-Correlation Coefficients

Parameter	r1	r2	r3	r4
Run -1	1.00	0.99	0.98	0.97
Run -2	1.00	0.98	0.96	0.93
Run -3	1.00	0.98	0.95	0.91
Run -4	1.00	0.96	0.91	0.88

The histogram fit for normal distribution is given in Fig. 6. It is observed that the data of Run-4 is similar to normal distribution when compared with other runs.

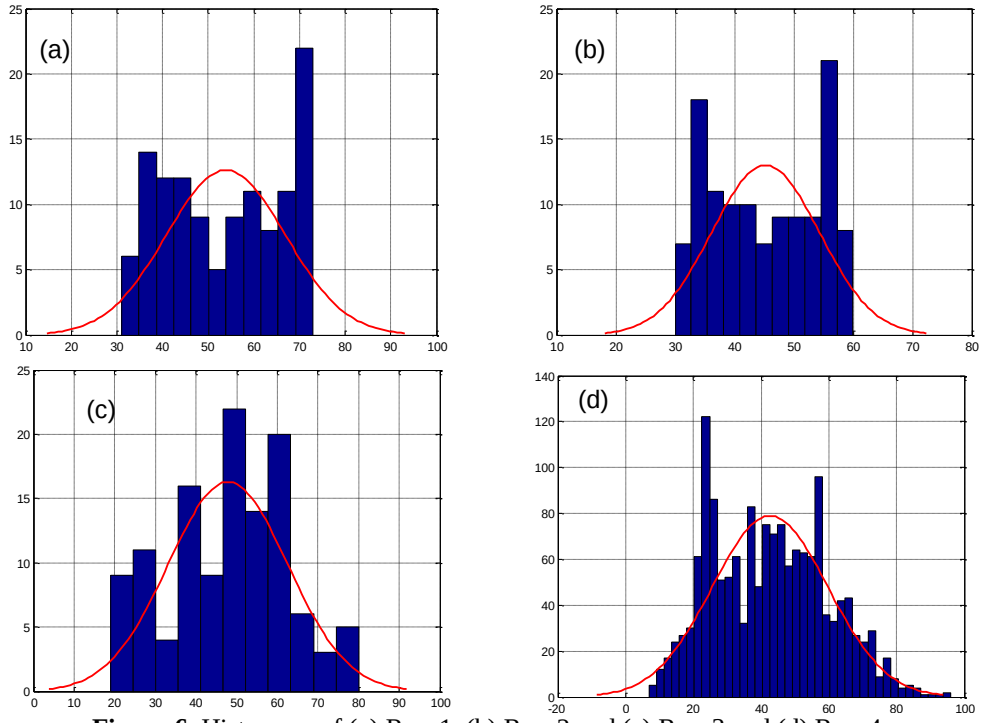

Figure 6. Histogram of (a) Run-1, (b) Run-2 and (c) Run-3 and (d) Run-4.

The function "`fft`" is provided by MATLAB which computes the discrete Fourier transform (DFT) with a fast Fourier transform (FFT) algorithm. The result is a vector of complex numbers, with real and imaginary parts. The magnitude of the complex numbers are calculated and depicted in Fig. 7 for Run-1, Run-2 and Run-3.

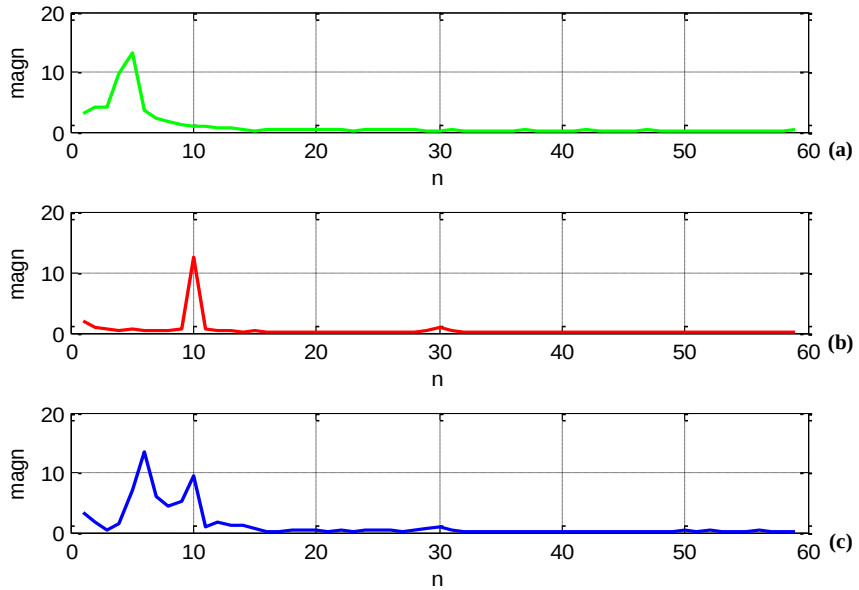

Figure 7. Magnitude of the coefficients of the Fourier Transform of (a) Run-1, (b) Run-2 and (c) Run-3.

The variation of the real, imaginary parts and the magnitude of the Fourier Transform of the Run-4 data is given in Fig. 7.

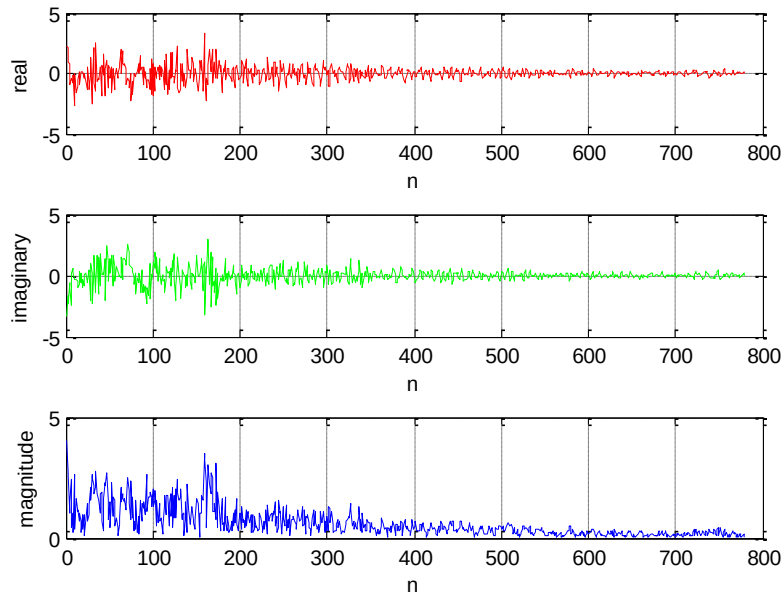

Figure 8. The real, imaginary parts and the magnitude of the Fourier transform of the time series of Run-4.

In Fig. 7 and 8, the parameter in the horizontal axis is n, which is defined as the frequency of the time series times the length of the time series divided by the sampling frequency, where the equation below holds.

$$f_x = \frac{n\ F_s}{N} \qquad (1)$$

The Frequency domain plots of Run-1 – Run-4 are plotted by the help of the Window Visualization Tool provided by MATLAB (Figure 9). Here the normalized frequency in the horizontal axis is the frequency (linear) of the periodic signal normalized to half the sample rate, denoted by f_N, (eqn. 2). The f_N values read and marked on the Fig. 9.

$$f_x = \frac{f_N\ F_s}{2} \qquad (2)$$

The f is the frequency (Hz) may be given in equation (3).

$$f_x = f\ F_s \qquad (3)$$

The n, f_N and f for the runs are tabulated as in Table 4.

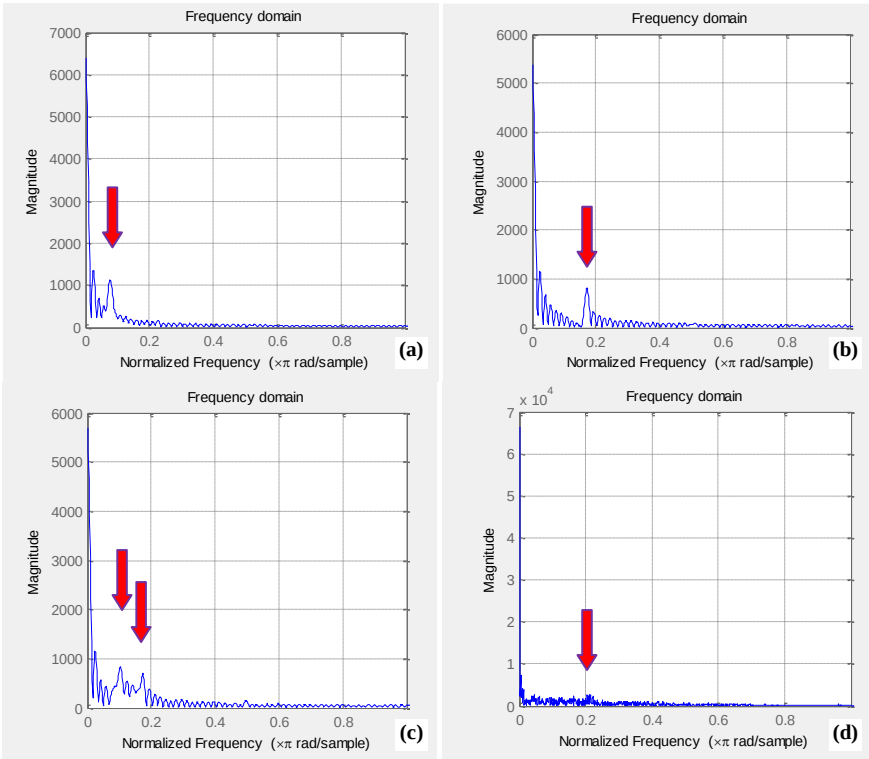

Figure 9. The frequency domain plots for (a) Run-1, (b) Run-2, (c) Run-3 and (d) Run-4.

Table 4. The Frequency Values of the Runs

Parameter	n	f_N	f
Run -1	4.5	0.0762	0.0381
Run -2	10.2	0.1718	0.0859
Run -3	4.5&10.2	0.1035&0.0518	0.1738&0.0868
Run -4	164	0.21	0.104

4. Results of the Short Time Fourier Transform

Rectangular window and Gaussian window are used in this study, where their frequency plots are depicted in Fig. 10 and Fig. 11, respectively. The leakage factor, relative sidelobe attenuation and the mainlobe width of both window types of length 64 are tabulated in Table 5.

Table 5. Properties of Rectangular and Gaussian Windows

Parameter	Leakage Factor	Relative sidelobe attenuation	Mainlobe width
Rectangular window	9.3 %	-13.1 dB	0.10938
Gaussian window	0 %	-46.5 dB	0.17969

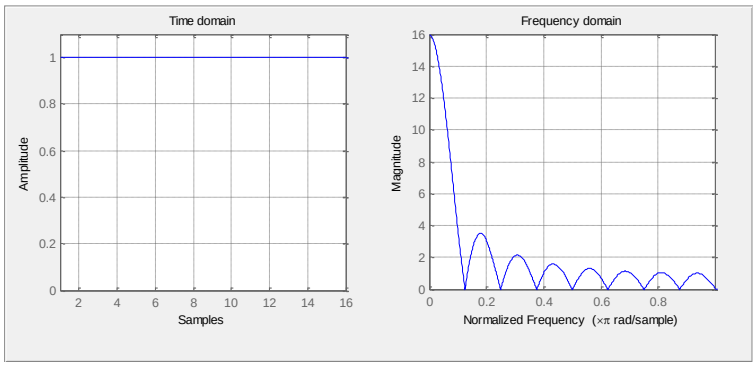

Figure 10. The time and frequency domain plots of rectangular window of length 64.

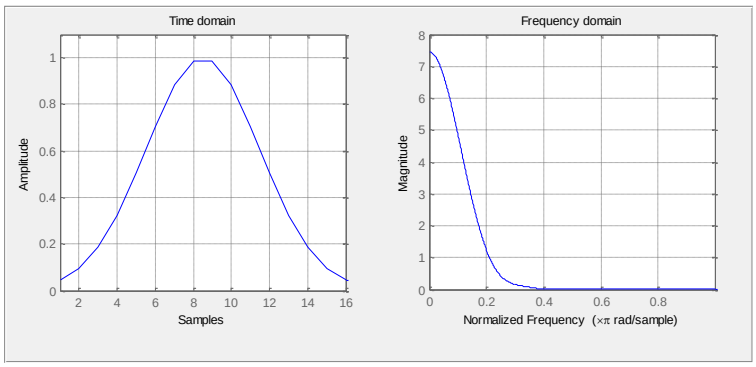

Figure 11. The time and frequency domain plots of Gaussian window of length 64.

The Signal Processing Toolbox product of MATLAB provides a function, "spectrogram" that returns the time-dependent Fourier transform for a sequence, or displays this information as a spectrogram. The time-dependent Fourier transform is the discrete-time Fourier transform for a sequence computed using a sliding window. These forms of the Fourier transform, also known as the short-time Fourier transform (STFT). The spectrogram of a sequence is the magnitude of the time-dependent Fourier transform versus time [1].

The short time Fourier transform of the input signal vector x is performed by the command "`spectogram`" by "`spectrogram(x,window,noverlap,nfft,fs)`". By default, x is divided into eight segments. If x cannot be divided exactly into eight segments, it is truncated. The "window" used here is a rectangular window and a Gaussian window, which stands for a STFT and Gabor Transform. The length of the window is determined by defining by "`window = rectwin(nfft);`" or "`window = gausswin(nfft);`" in a previous statement. Here nfft is the FFT length and is the maximum of 256 or the next power of 2 greater than the length of each segment of x. [1].

The STFT with rectangular window and Gaussian window are given in Fig. 12. Note that the "time" on the vertical axis stands for the length of the time series. It is observed that the f_N are revealed from these spectrograms are in accord with the ones given in Table IV, since they are stationary. It is also observed that the rectangular window detect the f_N values clearly with respect to the Gaussian window.

The spectrograms with window lengths from 1028 to 64 were depicted in Fig 13.

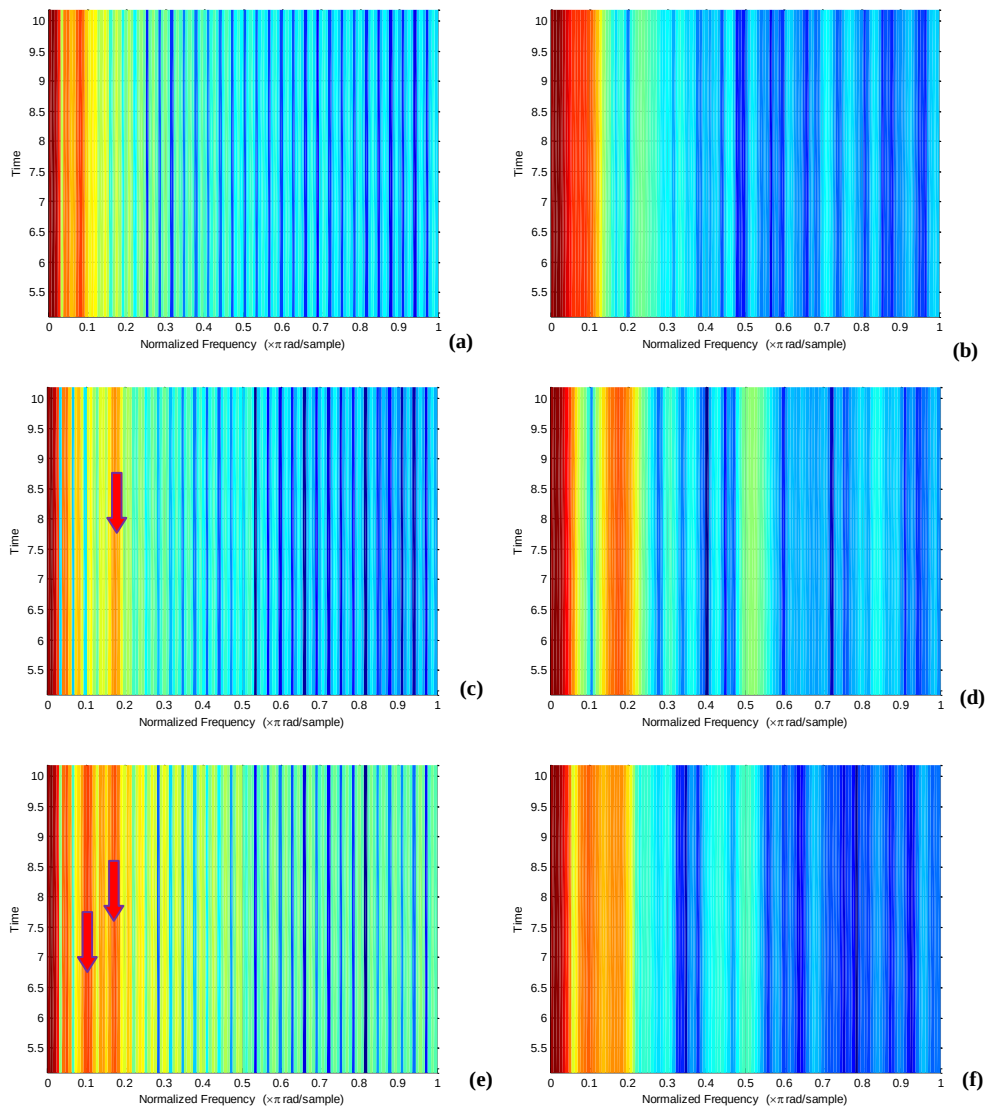

Figure 12. (a) & (b) Run-1, (c) & (d) Run-2, (e) & (f) Run-3, where (a), (c), (e) for rectangular and (b), (d), (f) for Gaussian window of length 64.

5. Conclusion

The frequencies of stationary and non-stationary data by using STFT with a rectangular window and a Gaussian window (Gabor Transform) are calculated and it is revealed that the for stationary data rectangular window and for non-stationary data Gaussian window is more appropriate to reveal the frequency content of a time series.

Acknowledgements
This work was conducted in collaboration with Mr. Ufuk BOMBAR to whom the author is grateful.

References

[1] MATLAB, User's Guide, Signal Processing Toolbox. 2010.

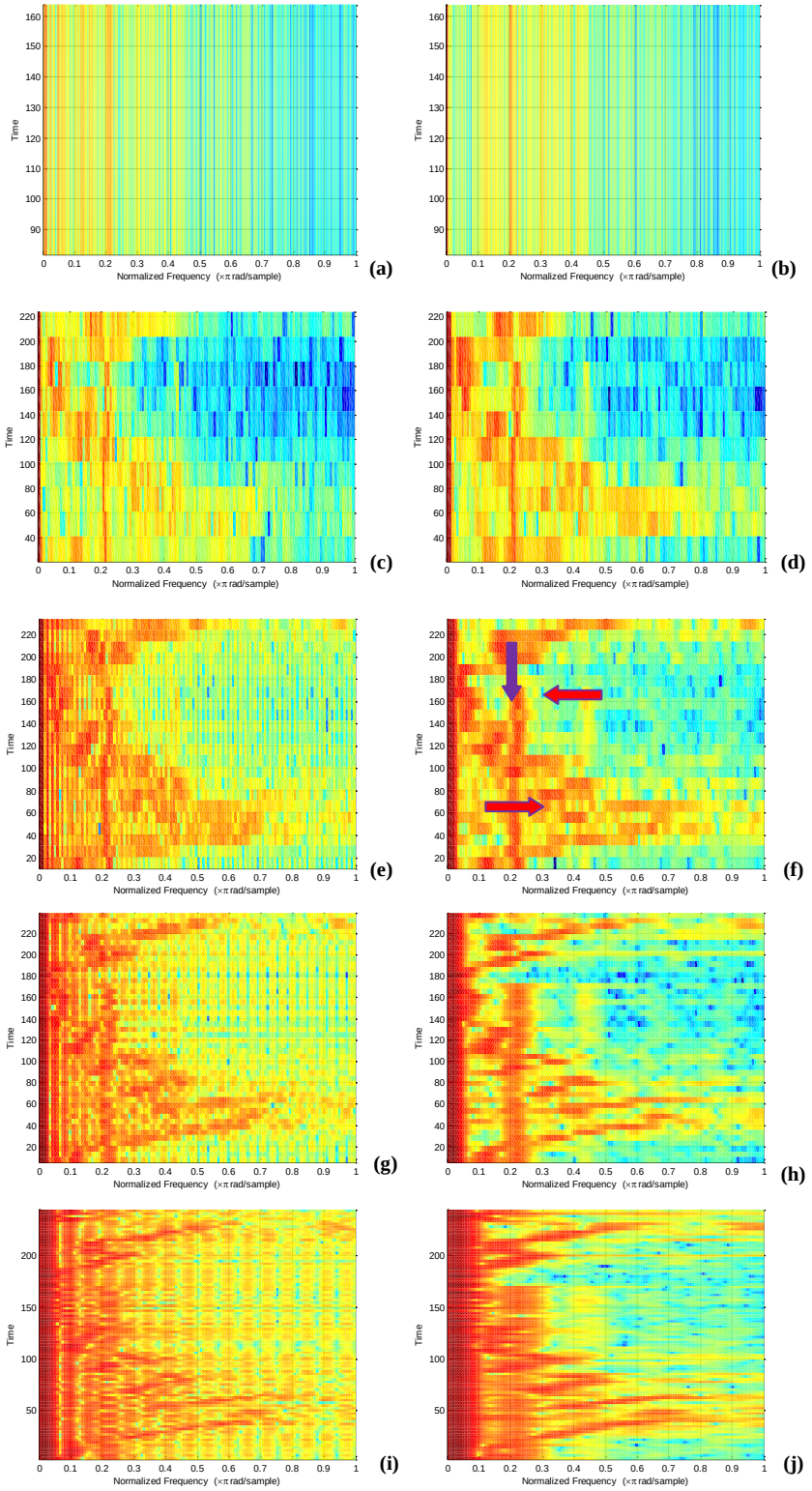

Figure 13. The length of the windows are (a) & (b) 1028, (c) & (d) 512, (e) & (f) 256, (g) & (h) 128, (I) & (j) 64 where (a), (c), (e), (g), (i) for rectangular and (b), (d), (f), (h), (j) for Gaussian window for Run-4.