Real-Time Spectrum Analyzer Light

Liu Zijia & Gao Min

June 2011

Bachelor’s Thesis in Electronics

Study Program for a Degree of Bachelor of Science in Electronics
Examiner: Niclas Björsell
Supervisor: Charles Nader
ACKNOWLEDGEMENT

The authors really appreciate Charles Nader for his professional, effective and enthusiastic supervision and providing useful matlabcode as reference. Without his help, this thesis project would not have been possible. The authors would also like to thank Niclas Björsell who gave us the general introduction to this project topic and provided useful suggestions.
ABSTRACT

Real-time spectrum analyzer is a hot instrument for signal analysis and spectrum monitoring. It has wide applications in wireless communication and high potential for smart reading. However, it is characterized with a high cost. Hence, a designed light version based on already available lab instruments is constructed for use in school lab. In this thesis, a simple oscilloscope and a computer were used in the process. Both instruments communicated through GPIB cable and MATLAB software was used for the processing tasks. In order to test the designed one, a simple signal generator was used. The signal was analyzed in both time domain and frequency domain. Then measurements of the signal over time were updated to a matrix of FFT and were plotted in 3D, showing spectrum variation over time. The application of this project to radio frequency analysis was also discussed.
TABLE OF CONTENTS

ACKNOWLEDGEMENT ............................................................................................................. i

ABSTRACT .............................................................................................................................. ii

TABLE OF CONENTS ............................................................................................................. iii

1. Introduction ......................................................................................................................... 1
   1.1. Background .................................................................................................................. 1
       1.1.1. Spectrum analyzer ............................................................................................... 1
       1.1.2. Real-time spectrum analyzer ............................................................................... 2
   1.2. Introduction to the project ........................................................................................... 3

2. Sampling and Fourier transform ......................................................................................... 4
   2.1. Sampling theorem ........................................................................................................ 4
   2.2. Coherent sampling ...................................................................................................... 5
   2.3. Discrete and fast Fourier transform ............................................................................. 5

3. Signal analysis .................................................................................................................... 7
   3.1. Principles of waveform analysis .................................................................................. 7
       3.1.1. Signal frequency ................................................................................................... 7
       3.1.2. Signal amplitude .................................................................................................. 7
   3.2. Principles of spectrum analysis .................................................................................... 8
       3.2.1. Amplitude spectrum ............................................................................................ 8
       3.2.2. Power spectrum .................................................................................................... 9

4. MATLAB review ................................................................................................................. 10
   4.1. FFT in MATLAB .......................................................................................................... 10
   4.2. Plots in MATLAB ........................................................................................................ 10
   4.3. TMtool in MATLAB ................................................................................................... 11

5. Process and results .............................................................................................................. 12
   5.1. Measurement setup ..................................................................................................... 12
5.2. Single tone analysis ................................................................. 13
  5.2.1. Time domain ................................................................. 13
  5.2.2. Frequency domain ......................................................... 14
5.3. Spectrum monitoring (3D-plot) ............................................. 16
  5.3.1. Fixed frequency ............................................................ 17
  5.3.2. Variable frequency ....................................................... 19
6. Discussion ........................................................................... 21
  6.1. Discussion of methods ....................................................... 21
  6.2. Analysis of the results ....................................................... 21
    6.2.1. Time domain ............................................................. 21
    6.2.2. Frequency domain ...................................................... 21
    6.2.3. Multiple domain ......................................................... 22
  6.3. Advantages and disadvantages ........................................... 23
7. Conclusion ............................................................................. 24
REFERENCE ................................................................................. 25
APPENDIX .................................................................................. A1
1. Introduction

1.1. Background

Nowadays, the radio frequency (RF) spectrum becomes more crowded and with more interference. Hence, the designers are endeavoring to improve the performance and resiliency of the wireless communication. These solutions are especially needed by those developing military and civil defense radio applications [1]. Spectrum monitoring is important for observing the frequency. It can be used to monitor the signal more directly and conveniently, so spectrum analyzer is widely used in modern radio testing. Future wireless communication analysis will do smart frequency hopping over wide bandwidth.

1.1.1. Spectrum Analyzer

Spectrum analyzer is an instrument for signal monitoring. It studies the spectrum structure of the signal and is used to measure the distortion degree, modulation degree, spectral purity, frequency stability and inter modulation distortion. It can also be used to measure the parameters of amplifier and filter system.

There are two types of spectrum analyzers. They are sweep spectrum analyzer and real-time spectrum analyzer (RTSA). The sweep spectrum analyzer [2] is widely used in the RF testing and is capable of observing the signal in time domain, in which way this kind of spectrum analyzer can reveal the invisible signals, which are measured using other type of signal-testing equipments. The advantages of the sweep spectrum analyzer are that it is cheap, able to be used in wide frequency range and have very wide scan spans. The disadvantages are that it cannot measure phase and transient events.

The RTSA [2] [3] is not as common as the former one, but it can trigger on the RF signal, capture, store it into the memory, and test the signal that changes over time. It will be introduced in detail in the following section.
1.1.2. Real-Time Spectrum Analyzer

Real-time spectrum analyzer [3] is the spectrum analyzer that uses the real-time signal processing to achieve and analysis the signal. Its working principle is that it transmits the measured signal through scanner to Cathode Ray Tube (CRT) or liquid crystal display (LCD), and the screen can display all the frequency components of the signal at an instant time. It can analysis the signal in time-domain, frequency-domain and modulation-domain. As shown in Fig.1, the transmitted periodic signal is transformed from time domain to frequency domain by using fast Fourier transform (FFT) inside the spectrum analyzer. The spectrum is the power of the complete Fourier information.

![Signal analyzing schematic](image)

Fig.1 Signal analyzing schematic

The advantage of RTSA is that it can show transient response of periodic random waves. The disadvantage is that it is very expensive. Moreover, the bandwidth ranges, the number of filters limit their function.

The high price of the RTSA motivates the authors to build a simply constructed real-time spectrum analyzer taking advantage of the equipments in the laboratory. The assembled spectrum analyzer changes the original ideas of integrally design.
1.2. Introduction to the project

The aim of this project is to use software Matrix Laboratory (MATLAB) instead of hardware to achieve a simple real-time spectrum analyzer in order to cut down the cost for school lab. In addition, technology engineers can achieve the aim of collecting signals, processing signals and analysis signals by only using a computer, an available oscilloscope and some cables. It is easy and achievable.

As shown in Fig.2, this thesis project is to build a simple real-time spectrum analyzer using MATLAB, function generator and oscilloscope. The function generator transmits the sine wave signal to the oscilloscope and the oscilloscope captures the signal and uses the analog-to-digital convertor (ADC) inside it to transform the analog data into digital format. Then the PC receives the time-domain signal through the GPIB and saves it. Finally, MATLAB uses the FFT function to transform the time-domain signal into frequency domain. Then the spectrum of signal with one frequency is displayed on the screen. Furthermore, the frequency is changed manually and a 3D spectrum figure is plotted. In this thesis, only the functionality of the 3D analyzing of the RTSA will be achieved.

![Diagram of the instruments]

Fig.2 The construction of the instruments
2. Sampling and Fourier transform

2.1. Sampling theorem

Sampling theorem states the constraint for converting and reconstructing signals without distortion during continuous signal sampling process. Referring to baseband signals, signal sampling frequency, $f_s$, is greater than or equal to two times the maximum frequency of the signal, $f$. That is $f_s \geq 2f$. The theorem contains two processes: one is a sampling process, which is to convert a continuous signal, $x(t)$, to a discrete signal, $x[n]$; and the other is a reconstruction process, which is to recover the continuous signal, $x(t)$, from the discrete signal, $x[n]$. In details, as shown in Fig.3, a continuous signal, $x(t)$, is measured every sampling interval $T$, units of time. Then the sampling results can be obtained in a discrete sequence, $x[n]$, shown as:

$$x[n] = x(nT) \tag{2.1}$$

where $x[n] = $ a discrete sequence

$n = $ an integer

$T = $ the sampling interval

![Fig.3 A discrete sequence $x[n]$ form a continuous signal $x(t)$ by sampling](image-url)
2.2. Coherent sampling

Coherent sampling is the sampling of a periodic signal without spectral leakage. It increases the spectral resolution of an FFT. It refers to a certain relationship between frequency of periodic signal, \( f \), sampling frequency, \( f_s \), integer number of cycles, \( M \), in the sampled set and number of samples, \( N \). They obey the following formula [4]:

\[
\frac{f}{f_s} = \frac{M}{N}
\]

(2.2)

where
\( f \) = frequency of periodic signal
\( f_s \) = sampling frequency
\( M \) = integer number of cycles
\( N \) = number of samples

2.3. Discrete and fast Fourier transform

1) Discrete Fourier transform (DFT)

In a variety of signal sequences, finite sequence is of an important role. Finite sequence can be analyzed by using DFT. The DFT not only can reflect spectral characteristics of the sequence well, but also is easy to use the FFT algorithm for analysis on the computer.

The DFT is a transform for Fourier analysis from time-domain functions into frequency-domain functions. It requires that an input function is periodic and discrete, and has non-zero values with a finite duration. Such inputs are often created by sampling a continuous function using the sampling theorem.

The DFT for a signal with periodic \( N \) complex numbers is [5]

\[
X[k] = \sum_{n=0}^{N-1} x[n]e^{-\frac{2\pi nk}{N}} \quad k = 0, \ldots, N - 1
\]

(2.3)

where \( X[k] \) = discrete Fourier transform
\( x[n] \) = a discrete sequence

The inverse discrete Fourier transform (IDFT) is [5]

\[
x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{\frac{2\pi j k n}{N}} \quad n = 0, ..., N - 1
\]

\( X[k] \) = discrete Fourier transform
\( x[n] \) = a discrete sequence
\( N \) = number of samples

In general, the DFT pair can be replaced by [5]:

\[
X[k] = \sum_{n=0}^{N-1} x[n] W_N^{kn} \quad k = 0, ..., N - 1
\]

\(
x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn} \quad n = 0, ..., N - 1
\)

where \( X[k] \) = the discrete Fourier transform
\( x[n] \) = a discrete sequence
\( N \) = number of samples
\( W_N = e^{-\frac{2\pi j}{N}} \)

2) Fast Fourier transform

Fast Fourier transform is a fast algorithm for computing the DFT. According to Cooley-Turkey algorithm, the FFT breaks down a DFT transformation formula repeatedly, making it a combination of several small data points, in order to reducing computation time.
3. Signal analysis

3.1. Principles of waveform analysis

Assume the input signal is a periodic and continuous signal, such as sine wave, cosine wave, triangle wave, and rectangular wave.

3.1.1. Signal frequency

Regarding a periodic signal, its period can be determined by analyzing the time-domain waveform, which is to calculate the time difference of two adjacent peaks or the time difference of two adjacent zero crossing. Assume the period of a signal is $T$, then the frequency, $f$, is the reciprocal of the period, $T$, i.e. $f = \frac{1}{T}$.

3.1.2. Signal amplitude

Signal amplitude, $A$, is equal to its peak value, $V_p$. In the sample data, the difference between the maximum, $V_{max}$, and the minimum, $V_{min}$, of the signal is the peak-to-peak value, $V_{pp}$.

$$A = V_p = \frac{1}{2} V_{pp} = \frac{1}{2} (V_{max} - V_{min}) \quad (2.7)$$

where

- $A$ = signal amplitude
- $V_p$ = peak value of signal amplitude
- $V_{pp}$ = peak-to-peak value of signal amplitude
- $V_{max}$ = maximum of signal amplitude
- $V_{min}$ = minimum of signal amplitude
3.2. Principles of spectrum analysis

Time domain analysis can only reflect changes of the signal amplitude over time. Expecting the simple single with unifrequency component, time-domain analysis is difficult to reflect the composition of the signal frequency components and the size of each frequency component. However, the spectrum analysis can be a good solution to this problem. It contains amplitude spectrum and power spectrum.

3.2.1. Amplitude spectrum

Spectrum amplitude, $S(f)$, is defined by the IEEE [6] as two times the magnitude of the Fourier transform, $|X(f)|$, of a time domain signal function, $x(t)$. The unit of $S(f)$ is volt-second (Vs) or volt per hertz (V/Hz) [7].

$$X(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft}dt \quad (2.8)$$

$$S(f) = 2|X(f)| \quad (2.9)$$

where $X(f) = \text{the Fourier transform}$

$x(t) = \text{a time domain signal function}$

$f = \text{signal frequency}$

$S(f) = \text{spectrum amplitude}$

$|X(f)| = \text{the magnitude of the Fourier transform}$

In common usage, spectrum amplitude is expressed in decibel.

$$S_{dBV} = 20\log_{10}S(f) \quad (2.10)$$

where $S_{dBV} = \text{spectrum amplitude in decibel Volt}$

$S(f) = \text{spectrum amplitude}$
3.2.2. Power spectrum

The power spectrum gives a plot of the portion of a signal's power, i.e. energy per unit time, falling within given frequency range. The most common way of generating a power spectrum is by using a DFT.

Regarding to a continuous time signal, \( x(t) \), with infinite length, the power spectrum is given by the formula:

\[
P(f) = \frac{|X(f)|^2}{R}
\]  

(2.11)

Where \( P(f) \) = power of spectrum

\( X(f) \) = the Fourier transform

In common usage, power of spectrum is expressed in decibel.

\[
P_{dBW} = 10 \log_{10} \left| \frac{|X(f)|^2}{R} \right|
\]  

(2.12)

\[
P_{dBm} = 10 \log_{10} \left| \frac{|X(f)|^2}{R} \times 1000 \right|
\]

\[
= 10 \log_{10} \left| \frac{|X(f)|^2}{R} \right| + 10 \log_{10}[1000]
\]

\[
P_{dBm} = 30 + 10 \log_{10} \left| \frac{|X(f)|^2}{R} \right|
\]  

(2.13)

where \( P_{dBW} \) = power of spectrum in decibel Watt

\( P_{dBm} \) = power of spectrum in decibel milliWatt

\( X(f) \) = the Fourier transform
4. MATLAB review

MATLAB [7] is commercial mathematic software developed by a U.S. company named MathWorks. It is a high technical computing language and interactive environment used for algorithm development, data visualization, data analysis and numeric computation. The FFT, fftshift, plots and TMtool in MATLAB are described in detail in the following sections.

4.1. FFT in MATLAB

MATLAB provides a rich set of mathematical functions for the discrete fast Fourier transform. For instance, there are fft and fftshift. One of the syntax for FFT is \( Y = \text{fft}(X) \). This function is used to compute the DFT of vector \( X \) with a FFT algorithm. One of the syntax for fftshift is \( Y = \text{fftshift}(X) \). This function moves the zero-frequency component to the center of the array to rearrange the outputs of FFT. It swaps the left half and the right half of vector \( X \). This is useful for visualizing a Fourier transform with the zero-frequency component in the middle of the spectrum [8].

4.2. Plots in MATLAB

MATLAB provides a rich set of plots. For instance, there are 2D-plot, 3D-plot and so on. Parts of the syntaxes for 2D-plot in MATLAB include: 1) Line plot: \( \text{plot}(x,y) \). This plot is composed of smooth lines. 2) Stem plot: \( \text{stem}(x,y) \). This plot draws a marker for each \( x, y \) value that is connected to a common baseline by a vertical line. An example used these two plots are shown in Fig.3.

Mesh plot is one of 3D-plot in MATLAB. The syntax is \( \text{mesh}(X,Y,Z) \). This draws a wireframe mesh with color determined by \( Z \) so that color is proportional to surface height. If \( X \) and \( Y \) are vectors, \( Z \) must be a matrix, where size \( (Z) = [m,n] \) corresponding to length \( (x) = n \) and length \( (Y) = m \). That is, vector \( X \) and vector \( Y \) correspond to the columns and rows of matrix \( Z \), respectively [8].
4.3. TMtool in MATLAB

The Test & Measurement Tool (TMtool) displays the resources, such as signal generator and oscilloscope, accessible to the toolboxes that support the tool, and enables to configure and communicate with those resources. It can be used to manage instrument control session. Its functionality contains searching for available hardware and drivers, creating instrument objects, connecting to an instrument, configuring instrument settings, writing data to an instrument, reading data from an instrument and saving a log of session as a file.
5. Process and results

In this thesis project, the oscilloscope and the signal generator were used based on GPIB. Besides, MATLAB was used as a tool during the whole process. There are many toolboxes handling different functions in different areas in MATLAB. In this project, the TMtool was mainly used. The first step of the process was to select a suitable signal generated by the signal generator and transferred to the oscilloscope, and then to communicate with the instruments. Then collect data from the oscilloscope and transfer it to the computer. Next, plot one signal in time domain and make an FFT to plot the amplitude and power spectrum in frequency domain. Then measure the signal over time, and make and update a matrix of power spectrums with plotting them in 3D. Finally, do the same plot but changing the frequency of the signal over time. More details were contained in the following sections.

5.1. Measurement setup

Firstly, the experimental instruments were set by connecting the oscilloscope with the signal generator and the computer through GPIB cable, as shown in Fig.2 in section 1.2.

![Fig.2 The construction of the instruments](image)
Secondly, use the signal generator to generate a suitable input signal. In this project, a sine signal with the amplitude of 250 mV and the frequency of 60 KHz was used. Then present the signal waveform on the screen of the oscilloscope.

Then start the MATLAB to communicate with the instruments. In this project, the TMtool was used. The main procedure was shown in APPENDIX.

5.2. Single tone analysis

First, generate and measure a sine signal. The signal was analyzed in time domain and frequency domain, respectively.

5.2.1. Time domain

After communicating with instrument object and reading data information, the maximum, minimum and time range of the signal can be obtained. In addition, the characteristics of the source were set as followed: the offset value is 0; the peak-to-peak value is 500 mV; the time range is 0.01 s . The commands used for reading data information included $Fs = \frac{2000}{fTimeRang}$, $N = 2000$, $Ts = \frac{1}{Fs}$, $t = (0: N - 1) * Ts$ . Through these commands, the following data can be obtained: the sampling number $N = 2000$; the sampling frequency $F_s = 200 \text{ KHz}$; the sampling rate $T_s = 5 \mu s$; the time range $t = 0.01 s$ . Then there was an important step, which was to convert the data into volts. The final step was to plot the data. The completed MATLAB-code was as shown in APPENDIX. Save the file and run the program. The signal in time domain was plotted in Fig.4.
5.2.2. Frequency domain

Regarding to the communication with instruments and based on the commands in section 3.2.1, the commands for reading data information were added in the edition part, including $f = (\frac{-N}{2}; \frac{N}{2} - 1) \cdot Fs/N$. This gave the frequency range, which was from $-100$ KHz to $100$ KHz.

Then according to Eq. (2.10) and Eq. (2.13), an FFT transform was made by adding the command $y = dB(abs(fftshift(fft(x, length(x)))/length(x)))$ for amplitude spectrum, and $Y = 30 + 10 \cdot \log10((abs(fftshift(fft(x, length(x)))/length(x))))^{2/50}$ for power spectrum. In the command of the power spectrum, the value 50 stands for the common used resistance. Use the command `plot(f(end/2 + 1: end), y(end/2 + 1: end))` and the command `plot(f(end/2 + 1: end), Y(end/2 + 1: end))` to plot the two figures, which cut out the negative parts. The plots of the amplitude spectrum and the power spectrum were shown in Fig.5 and Fig.6, correspondingly.

*Fig.4 The sine signal in time domain collected from the oscilloscope*
Fig. 5 The amplitude spectrum of the sine signal in frequency domain

Fig. 6 The power spectrum of the sine signal in frequency domain
5.3. Spectrum monitoring (3D-plot)

Since the aim is to continuously plot the spectrum in every 5 seconds, 3D-plot needs to be applied.

First, an empty matrix with 10 rows and 1000 columns was set. Then the first row was written into the measured data, which was stored in $y$. Then after 5 seconds, the data in the first row was shifted into the second row and the first row was entered with the new-measured data, so after 10 times shifting, the empty matrix was filled with the measured data. The whole process was illustrated in Fig.7, as shown below. Since the function generator kept generating the signal, the matrix kept updating new data.

Each row contained the measured signal at one instant time and was plotted in frequency domain as a spectrum, so when the loop continued running, there would be 10 different measured signals and was plotted in frequency domain every 5 seconds. As a result, there appeared a 3D spectrum diagram. The illustrated principle was shown in Fig.8.

In MATLAB, command $mesh(f,T,M)$ can be used to plot 3D diagram. In this command, $f$ represents the frequency vector, $T$ represents the time vector and $M$ represents the power matrix. Therefore, this command was chosen to plot the final 3D spectrum diagram.
5.3.1. Fixed frequency

Firstly, the measured frequency was fixed. The result was shown in Fig.9.
In order to observing the spectrum more clearly and finding the useful frequency, A side view from xz-plane and an overhead view were plotted as shown in Fig.10 and Fig.11.

**Fig.10** The side view from xz-plane of the 3D-plot with the same frequency spectrum

**Fig.11** The overhead view of the 3D-plot with the same frequency spectrum
5.3.2. Variable frequency

After measuring one fixed frequency 60 kHz, a multiple-frequency measurement was done. The frequency was manually changed from 10 kHz to 100 kHz in step of 10 KHz.

The 3D-plotting result was shown in Fig.12.

Fig.12 The 3D-plot for changing frequency
In addition, the plot of a side view from xz-plane was presented in the following Fig.13.

**Fig.13** The side view from xz-plane of the 3D-plot for changing frequency

Finally, the plot of an overhead view was presented in the following Fig.14.

**Fig.14** The overhead view of the 3D-plot for the changing frequency
6. Discussion

6.1. Discussion of methods

In this project, the methods were good due to the following reasons. Firstly, the function generator, oscilloscope and computer were chosen. They are very common in laboratory and easy to handle so that the communication between the operator and the equipments and the data transmission between the equipments was successful. Secondly, choosing the software MATLAB as a tool to communicate with the oscilloscope was a good method. In this thesis work, an abundant data needed to be processed and complicated spectrum analyzing needed to be calculated. MATLAB accomplished them easily and precisely.

6.2. Analysis of the results

The results can be analyzed in three different sections: time domain, frequency domain and multiple domains. In each section, the results displayed will be compared with the expected results.

6.2.1. Time domain

After setting the function generator with the frequency of 60 KHZ, the oscilloscope should display the signal with 60 KHZ. The frequency of the signal in time-domain on the screen can be calculated by reading the duration of a period $T$ on the oscilloscope and using the formula $f = \frac{1}{T}$. The duration of one period of the signal is 0.0167 ms, so that $f = \frac{1}{T} = \frac{1}{0.0167 \text{ ms}} = 60 \text{ KHz}$. The signal frequency on the oscilloscope is equal to the frequency displayed on the function generator. Therefore, the result in time domain satisfies the authors.
6.2.2. **Frequency domain**

As Fig.6 shown, there is one peak in the positive half of the spectrum, which is located in 60 KHz. It corresponds to the measured signal frequency 60 KHz. According to the sampling theorem, the measured signal is less than $f_s/2$, so there is no aliasing. The spectrum of the measured signal is located at the frequency of the signal, so the result in frequency domain is acceptable.

6.2.3. **Multiple domain**

From the plot of the overhead view in Fig.11, it could be seen that there was one straight dark line at the frequency of 60 KHz. It was because that the sine signal had a frequency of 60 KHz and the sampling frequency was 200 KHz. According to the sampling theorem, the frequency of the signal was less than half the sampling frequency, so the amplitude spectrum was exactly located at the signal frequency. The plot in 3D in Fig.9 was a matrix of the power spectrums measured every 5 seconds. Because the signal generator kept up generating signals, the matrix was updated for 10 times. However, signals generated by the function generator were relatively stable so that the power spectrums were almost at the same frequency. This was the reason why there was one straight dark line in the overhead view plot in Fig.11. Because the negative frequency was not used in practice, the negative part was cut out. From this result, it could be seen that the frequency at which the line was, was the useful frequency. In this case, it was 60 KHz. This result was acceptable in this project.

From the 3D-plot in Fig.12, it can be seen that the spectrums were at different frequencies since the signal frequency was changed every 5 seconds. People can pick up the spectrum they want to use by checking the plot, which marks the frequency, the time and the amplitude or power.

The fundamental theory of the RTSA is the ability to trigger an RF signal and capture it into memory and analyze it in multiple domains, this realized the process of detecting and characterizing the RF signals that change over time. In this thesis work, the spectrum plotted in 3D is clear, it reveals the power, and the frequency of the spectrum over time, so this results is what the authors expected. The functionality of the RTSA was achieved.
6.3. Advantages and disadvantages

This project is to build a real-time spectrum analyzer and use it to plot the spectrum of the changing frequencies in 3D, so the advantages and disadvantages will be discussed based on the functionality the light version of RTSA.

1) The light version can adjust the time duration between each measurement. Therefore, the look of the upcoming 3D plotting of the spectrum is flexible. Even though the time duration can be set to be very small, the light version is still incapable to achieve seamless measurement of the signal, which the real RTSA is capable.

2) The result of the 3D spectrum plotting is colorful and is added with the color bar to show the power distribution, which gives a better explanation of the spectrum. However, it is not as smooth as the plot of the real RTSA.
7. Conclusion

The aim of this thesis project is to design a simple real time spectrum analyzer to achieve its functionality of plotting in 3D. The main theory was to use MATLAB for communicating with the signal generator and the oscilloscope through GPIB cable. Based on the literature review before this project, the real time spectrum analyzers produced by companies were very expensive. Therefore, it needed to use the available instruments in school lab to construct a simple one in order to cut down the cost. During the whole project, a sine signal was handled and analyzed in both time domain and frequency domain. Besides, measurements of the signal over time were updated to a matrix of FFT and were plotted in 3D. In addition, a 3D-plot of the power spectrums whose frequencies were changed over time was analyzed.

The final 3D-plot result shows that the power spectrums of sine signals, whose frequencies changed from 10 KHz to 100 KHz in step of 10 KHz every 5 seconds, were updated for 10 times. From the side-view and the overhead view figures, it can be seen clearly at which frequency the spectrums were located.

In the future work, this project can be also extended in radio frequency analysis in order to finding the common useful frequencies of RF signals. Since RF signals are relatively unstable and there are many interferential signals, common useful frequencies need to be found. This can be done by using an antenna to receive RF signals as input signals instead of using the signal generator to generate signals. Then RF signals are transformed to spectrums using FFT and the matrix is plotted in 3D. The plot is updated over time. From the plot of the overhead view, it can be seen that there were dark lines, which are the common useful frequencies of those RF signals.


APPENDIX

The main procedure of communicating with the instrument using TMtool:

Start tmtool in MATLAB ==> Expand hardware ==> Select GPIB ==> Push the scan button ==> Expand GPIB ==> Select NI (National Instruments) ==> Scan ==> Expand NI ==> (At this moment, the instruments connected with the computer were shown in the front, which were the oscilloscope and the signal generator. The instrument numbers were 7 and 12, respectively.) ==> Select the oscilloscope ==> Connect ==> Write the GPIB command *idn? and push the query button ==> (Now an answer with the id for the oscilloscope was obtained.) ==> Write commands and push the query button ==> Disconnect ==> Switch over to the session log tab ==> (Now the matlabcode for the whole session can be seen.) ==> Save the file by pushing the save session button. ==> Switch to the MATLAB window and run the file ==> Edit the m-file ==> Save the file and run the program.

MATLAB-code:

```matlab
% Find a GPIB object.
obj1 = instrfind('Type', 'gpib', 'BoardIndex', 0, 'PrimaryAddress', 7, 'Tag', ' ');

% Create the GPIB object if it does not exist, otherwise use the object that was found.
if isempty(obj1)
    obj1 = gpib('CONTEC', 0, 7);
else
    fclose(obj1);
    obj1 = obj1(1)
end

% Connect to instrument object, obj1.
set(obj1, 'inputbuffersize', 2048)
fopen(obj1);
oinstrID = query(obj1, '*idn?');```
% Set source
fprintf(obj1, ':wav:form byte');
fprintf(obj1, ':meas:sour chan1');
fprintf(obj1, ':CHANnel1:OFFSet 0');
fprintf(obj1, ':CHANnel1:RANGe 1000E-03');
fprintf(obj1, ':TIMebase:RANGe 10E-03');
ftimeRange = str2num(query(obj1, ':TIMebase:RANGe?'));

% Communicating with instrument object, obj1.
data1 = query(obj1, ':meas:vmax?');
data2 = query(obj1, ':meas:vmin?');
fprintf(obj1, ':wav:form byte');
fprintf(obj1, ':meas:sour chan1');

% Read data information
oInstrPreamble = query(obj1, ':WAV:PREamble?');
arrInstrPreamble = str2num(oInstrPreamble);
oStructPRE.Format = arrInstrPreamble(1);
oStructPRE.Type = arrInstrPreamble(2);
oStructPRE.Points = arrInstrPreamble(3);
oStructPRE.Count = arrInstrPreamble(4);
oStructPRE.XIncrement = arrInstrPreamble(5);
oStructPRE.XOrigin = arrInstrPreamble(6);
oStructPRE.XReference = arrInstrPreamble(7);
oStructPRE.YIncrement = arrInstrPreamble(8);
oStructPRE.YOrigin = arrInstrPreamble(9);
oStructPRE.YReference = arrInstrPreamble(10);

% Read data
Fs = 2000/fTimeRange;
Ts = 1/Fs;
N = 2000;
t = (0:N-1)*Ts;
f = (-N/2:N/2-1)*Fs/N;
T = 0.5:48;
M = zeros(10,1000);

% The following loop is repeated 10 times
for i=1:10
    fprintf(obj1, ':wav: data?');
    oDataTemp = fread(obj1,oStructPRE.Points+11');
    oDataRAW = oDataTemp( 11:(end-1) );
    i
    % Convert into volts: (codes-cod.ref)
    oDataVolts = ((oDataRAW - oStructPRE.YReference)*oStructPRE.YIncrement) -
    oStructPRE.YOrigin;
    x = oDataVolts;
    figure(1);plot(t,x);
    % Make an FFT and plot the amplitude spectrum
    y = dB((abs(fftshift(fft(x,length(x))/length(x)))));
    figure(2);plot(f(end/2+1:end),y(end/2+1:end));
    % Make an FFT and plot the power spectrum
    Y = 30 + 10*log10((abs(fftshift(fft(x,length(x))/length(x)))).^2/50);
    figure(3);plot(f(end/2+1:end),Y(end/2+1:end));
    % Update the plot of matrix
    M(i,:) = Y(end/2+1:end);
    figure(4);mesh(f(end/2+1:end),T,M);
    % Collect datat every 5 seconds
    pause(5);
end

% Disconnect all objects.
fclose(obj1);
% Clean up all objects.
delete(obj1);

Begin here. The following two parts need to add respectively in the forloop in order to see every figure.

% Plot an overhead view
view(2);
shading flat;

% Plot side view from xz-plane
view(0,0);
grid;

End here