Military Technical College
Kobry Elkobbah,
Cairo, Egypt

5th International Conference
on Electrical Engineering
ICEENG 2006

REAL-TIME IMPLEMENTATION OF A NEW RADAR SIGNAL PROCESSOR BASED ON THE TIME-FREQUENCY APPROACH

K.H. Moustafa [*] and H.E.Abou-Bakr Hassan [***]

Abstract

Although the number of ways of describing a given signal is countless, the most important and fundamental variables in nature are time and frequency. While the time domain function indicates how the signal's amplitude changes over time, the frequency domain function tells how often such changes take place. The bridge between time and frequency is the Fourier transform. One of the important applications of time-frequency transform is the detection and extraction of a radar signal immersed in noise jamming. In this paper a new technique dealing with improvement of radar performance under jamming conditions is presented. This new technique of time-frequency processing with non-linear threshold level is applied to a chirp radar pulse imbedded in a high level noise. A software program is built using MATLAB program.

This paper also introduced a real-time implementation of the proposed model. The real-time implementation is done based on input/output data acquisition card using MATLAB, SIMULINK toolbox, Real-Time Workshop (RTW) toolbox and Real-Time Windows target (RTWT) toolbox. In addition, real-time windows target requires the Watcom C/C++ compiler.

I. INTRODUCTION

Radar transient signals are usually transient in time domain and wide-band in frequency domain. Since classical methods (such as Fourier transform) have difficulty to handle transient signal analysis, time-frequency or time-scale analysis [1] is more suitable for preserving high-frequency contents of the information carried by transient signals. Comparing with Fourier transform, the time-frequency/ time-scale transform of a transient signal may detect and locate rapid changes of the signal better than Fourier transform. One of the important applications of time-frequency/ time-scale transform is the detection and extraction of unknown radar signals in noise. The localization of the time and frequency by time-frequency/time-scale transform makes it possible for de-noising, signal detection and extraction in the time-frequency domain. In this paper a new technique dealing with improvement of radar performance under jamming conditions is presented. This new technique of time-frequency processing with non-linear threshold level is applied to a chirp radar pulse imbedded in a high level noise. The purpose of the analysis is to detect, extract the

[*] Egyptian Armed Forces
[***] Egyptian Armed Forces

signal imbedded in noise and finally recover the radar pulse again in time domain. A software program is built using MATLAB program. The recovered chirp radar pulse has an excellent agreement with the original one.

The fundamental idea of time-frequency analysis is to understand and describe situations where the frequency content of a signal is changing in time. Time and frequency analysis are not good enough because they do not fully describe what is happening. The main motive of time-frequency analysis is to devise a procedure to construct a distribution in the time-frequency domain. We want a joint distribution, which will give us the fraction of the total energy of the signal at time t and frequency f. We call that distribution $P(t, f)$. If we add all bits of energy from different time-frequency cells, we will get the total energy

$$\int_{-\infty}^{\infty} P(t,f)\,dt \quad df = E \tag{1}$$

For any particular case, the intensity of the signal can be rescaled so that E=1, without loss of generality. For a given time if we add up the energy at different frequencies we will get the total energy at the specified time. But the energy at a given time is given by $|s(t)|^2$. Thus we would like our distribution to satisfy

$$\int_{-\infty}^{\infty} P(t, f)\ df = |s(t)|^2 \tag{2}$$

.

For a given frequency we add all time pieces. We hope to get the density in frequency

$$\int_{-\infty}^{\infty} P(t, f)\ dt = |s(f)|^2 \tag{3}$$

The equations (2) and (3) are marginal conditions [1].

The Wigner distribution [2,3,4] (also referred as Wigner-Ville distribution) was the first distribution introduced and has been extensively studied. It has some remarkable properties, it is the prototype, of these methods. The Wigner distribution is

$$W(t, f) = \int_{-\infty}^{\infty} e^{-j2\pi f\tau} s^*(t - \frac{1}{2}\tau)s(t + \frac{1}{2}\tau)\ \ d\tau \tag{4}$$

The Wigner distribution is real as can be easily verified [1]. There is a considerable advantage to having a simple method to generate all possible time-frequency distributions. This allows one to pick and choose one with desirable properties. Such a method is available and may be expressed in a few different ways. The most direct way is to generate the distributions from [5].

$$P(t, f) = \iiint e^{j2\pi\nu(u-t)} g(\nu,\tau) s^*(u - \frac{1}{2}\tau)s(u + \frac{1}{2}\tau)e^{-j2\pi f\tau} d\nu\ \ du\ \ d\tau \tag{5}$$

Where $g(\nu, \tau)$ is an arbitrary function called the kernel. Once the kernel is chosen the distribution is fixed. If we want to study the class of distributions, which satisfy the marginals, then we consider only kernels, which satisfy these conditions.

II. TIME-FREQUENCY PROCESSING

The noisy chirp data used consists of 10880 samples. The SNR of this data is about -4.6693 dB. The time domain representations of the chirp radar pulse in both clear environment and

embedded in a high level noise is shown in Fig.1. It is clear from Fig.1 that the chirp radar

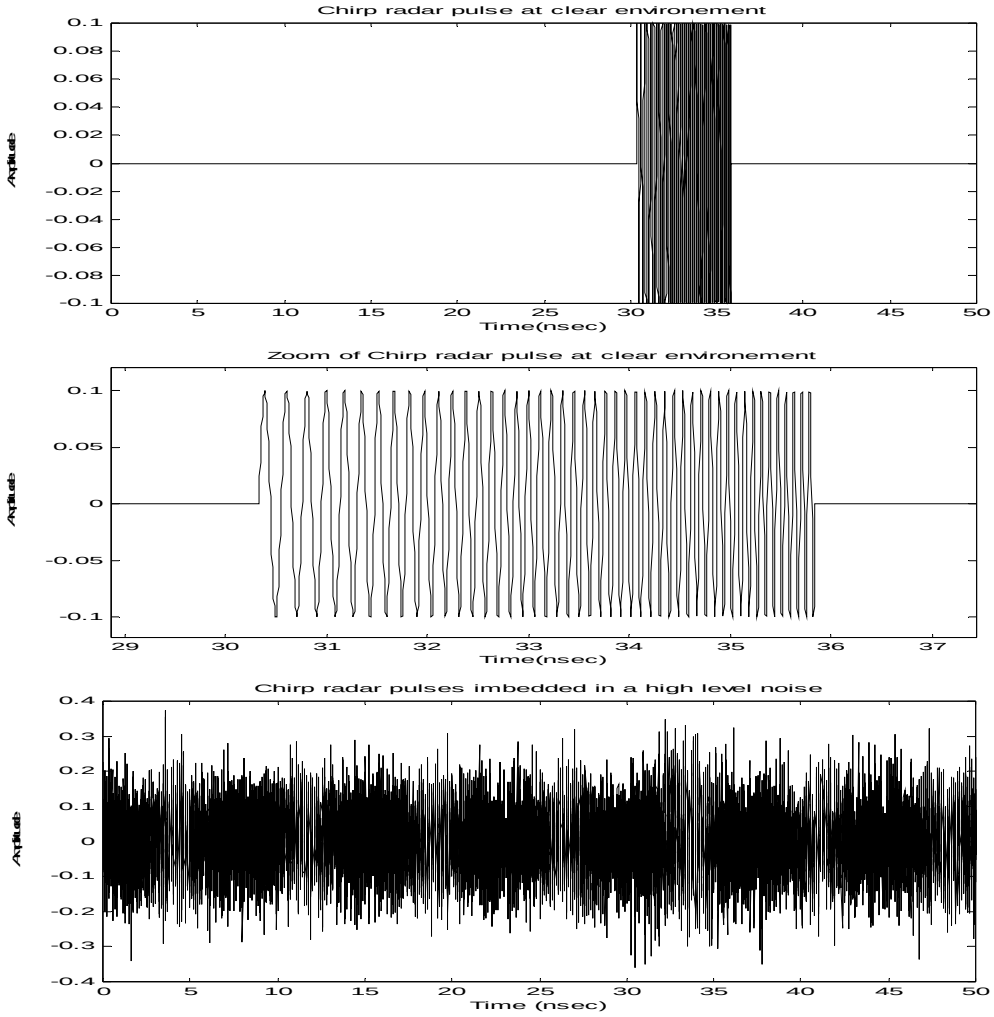


Fig.1. Representation of the chirp radar pulse in both clear environment and embedded in a high level noise respectively.

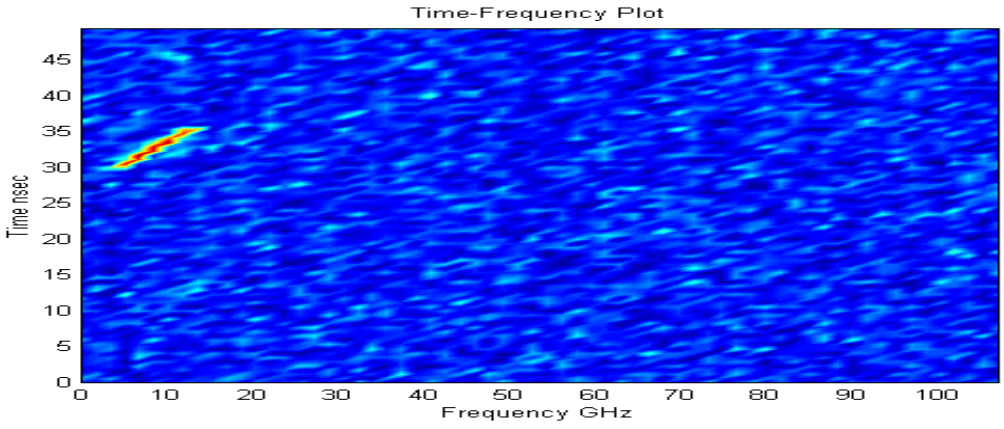


Fig.2. Two-dimension representation of the chirp radar pulse embedded in noise in time-frequency domain.

The previous representation produces a three-dimension representation of the chirp radar pulse imbedded in noise in time-frequency domain as shown in Fig.3.

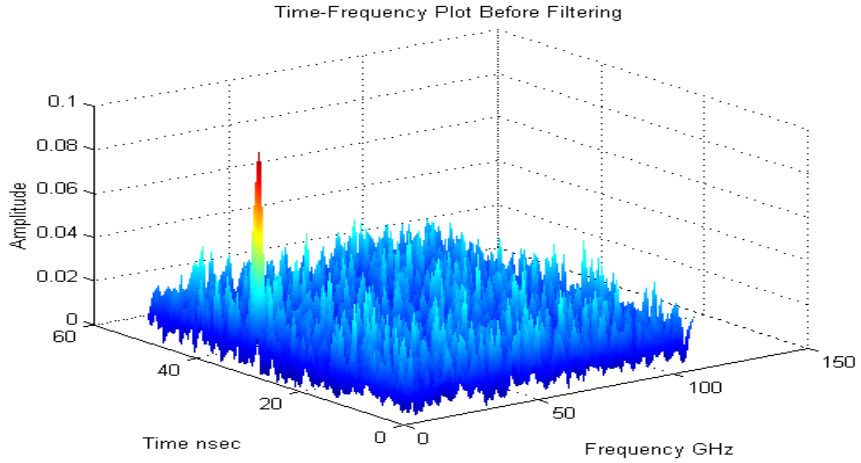


Fig.3 Three-dimension representation of the chirp radar pulse imbedded in noise in time-frequency domain.

III. Simulation of Time-Frequency processing:

The proposed time-frequency algorithm applies a sliding window on a time series and performs the Fourier transform for 128 samples in 85 windows. It plots the time-frequency diagram and the color map of the results. It applies a non-linear filtering and performs an Inverse Fourier transform for 128 samples in 85 windows to extract the chirp radar pulse. Finally it calculates signal to noise ratio. The signal power is calculated from the output of the time-frequency filter. The power is the sum of the squares. The power of the signal plus noise is then calculated by SNR = signal/(signal plus noise- signal).
This algorithm is simulated using SIMULINK. This new model shown in Figure can be added to the library of components and devices as a subsystem within SIMULINK to be useful for those searchers interested in this field. Fig.4 shows the representation of time-frequency processing using SIMULINK. This Figure also has a conventional BPF tuned to the chirp radar signal with the same parameters as before.

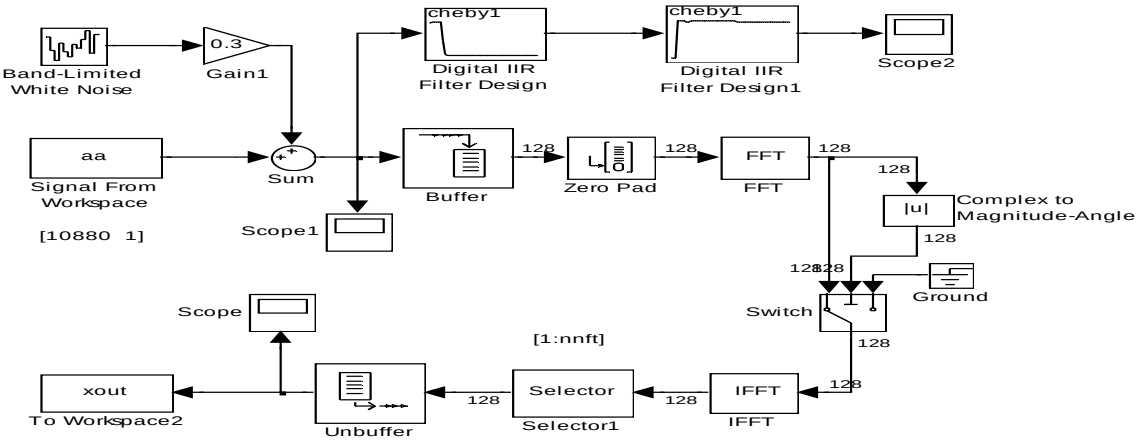


Fig.4 Model of Time-frequency

The original chirp radar pulse in combination with a band-limited white noise is applied to the buffer block. The Buffer block acquires a sequence of input samples into a frame (window) of 128 samples. The Zero Pad block increases the frame size of the input by appending an equal number of zeros to each channel. This is followed by FFT block, which computes the fast Fourier transform of each input window independently at each sample time. Then a non-linear threshold is applied using the switch. The Switch block propagates one of two inputs to its output depending on the value of a third input called the control input. If the signal on the control (complex to magnitude-angle) is greater than or equal to the threshold parameter, the block propagates the first input (output of FFT block); otherwise, it propagates the third input (ground). The switch feeds the IFFT block, which computes the inverse fast Fourier transform of each window independently at each sample time. This is followed by a selector block to select the actual frame size. The selector output is applied to the Unbuffer block, which converts each frame into a sequence of scalar outputs suitable for the Scope display.

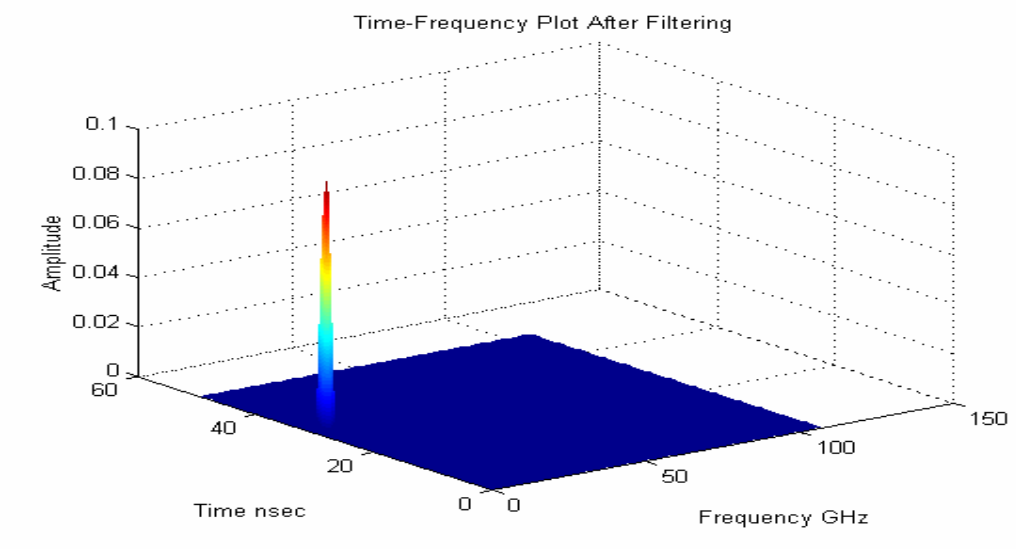
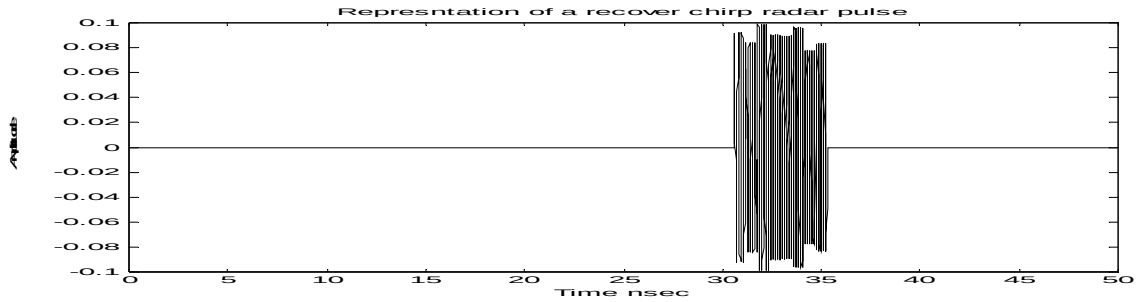
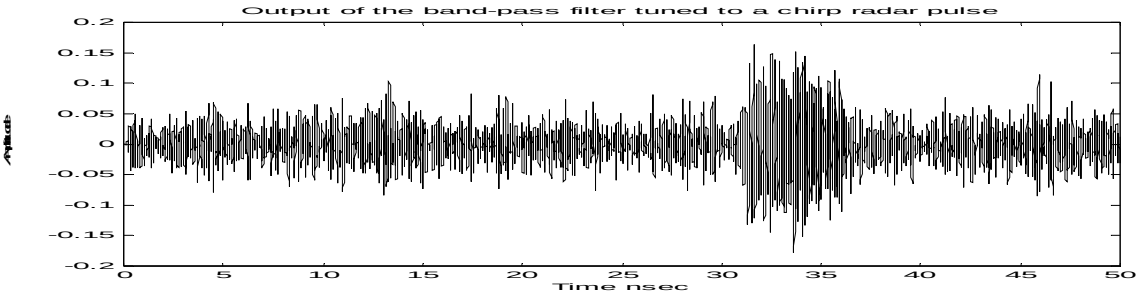


Fig.5 Three dimensions representation of a recovered chirp radar pulse in time-frequency domain

Fig.5 shows a good capability of the introduced algorithm in extracting the chirp radar pulse from the noise. The representation of the recovered chirp radar pulse is shown in Fig.6(a). The result from the tuned BPF is shown in Fig.6(b). From this Figure it is clear that the time-frequency model is successful in extracting the chirp radar pulse from a high noise while the conventional filter failed.



(a) Representation of the recovered chirp radar pulse using time-frequency model



(b) Output of the BPF filter tuned to the chirp radar pulse.

Fig.6 (a) Representation of the recovered chirp radar pulse using time-frequency algorithm and (b) Output of the BPF tuned to the chirp radar pulse.

It is very useful to compare the spectrum of the recovered chirp radar pulse using time-frequency algorithm to the spectrum of the conventional BPF as shown in Fig.7

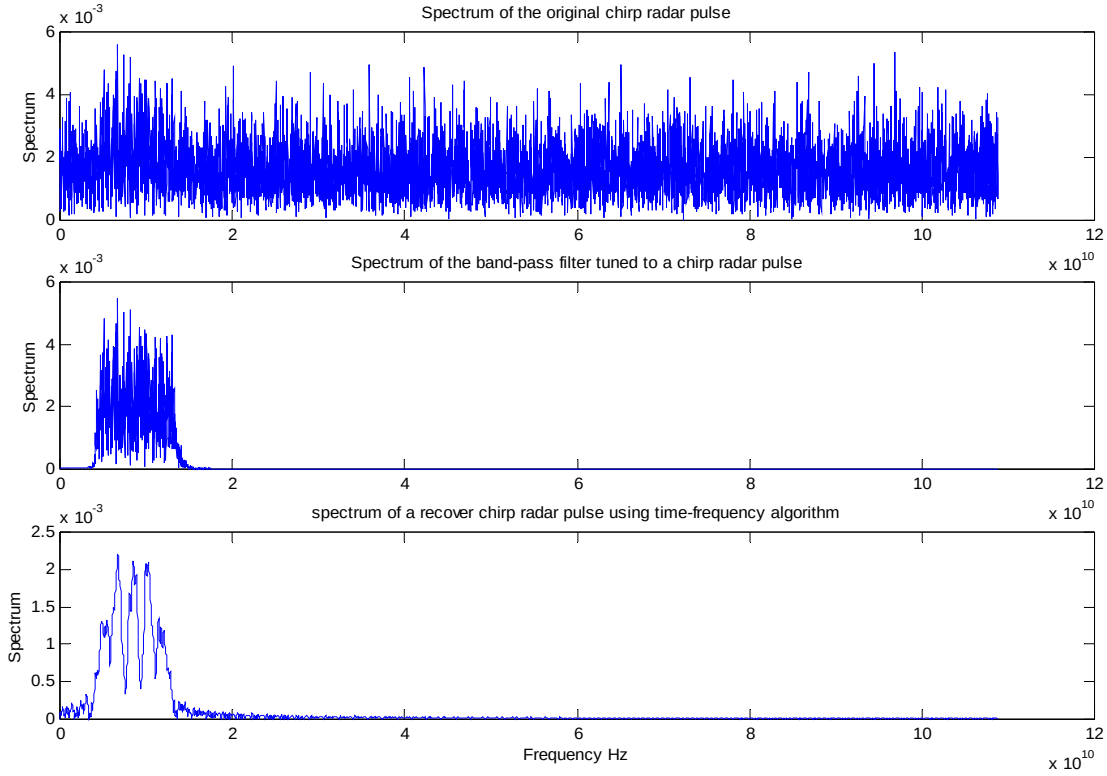


Fig.7 Spectrum of the chirp pulse, spectrum of the conventional BPF and spectrum of the recovered chirp pulse using time-frequency algorithm

IV. REAL-TIME IMPLEMENTATION OF TIME-FREQUENCY TECHNIQUE

The real-time implementation is based on input/output data acquisition card using MATLAB, SIMULINK, the RTW toolbox and the RTWT toolbox. In addition, RTWT requires the Watcom C/C++ compiler. Real time implementation requires that the model is free of algebric loops (delay free loops). In computer time implementations, models with algebric loops are solved iteratily using an optimisation subroutine. This is not possible in real time.

The process of creating and running a real-time application are; creating a SIMULINK model, entering simulation parameters for real-time workshop, entering scope properties for signal tracing, creating a real-time application, connecting the SIMULINK model to the real-time application, and running a real-time application.

The RTW provides the utilities for converting SIMULINK models into C code, and then compiling the code into a real time executable. The executable is then loaded into memory and the RTWT runs the executable in real time. The RTWT uses a small real-time kernel to ensure that the real-time application runs in real time. The real-time kernel runs at PC clock as its primary source of time. It is only during real-time execution of the model that the kernel intervenes to ensure that your model is given priority to use CPU to execute each model update at the prescribed sample intervals. Once the model update at a particular sample interval completes the kernel releases the CPU to run any other windows application that may need servicing. The kernel interfaces and communicates with I/O hardware using I/O drivers. It checks for proper installation of the I/O board using plug and play. If the board has been properly installed, the drivers allow the real-time application to run.

Integration between SIMULINK external mode and the RTWT enables capturing and displaying signals from the real-time application. During real-time execution, SIMULINK's external mode provides bi-directional exchange of data. New model parameter value values can be passed down to the real-time application while signal data can be retrieved from the real-time application and displayed in SIMULINK scope blocks. SIMULINK external mode provides a universal interface for use with RTW targets. It allows us to make parameter changes in the SIMULINK block diagram and have the new parameter changes exported automatically to the real-time application. External mode requires a communication interface to pass parameters external to SIMULINK, and on the receiving end, the same communications protocol must be used to accept new parameter values and insert them in the proper memory locations for use by the real-time application. In the case of RTWT, the host computer also serves as the target computer. Therefore, only a virtual device driver is needed to exchange parameters between MATLAB and SIMULINK memory space and memory that is accessible by the real-time application.

The RTWT provides drivers for different boards. These drivers connect the physical world to the real-time application. Connecting sensors and actuators to I/O boards does this connection to the physical world. The drivers then read data from and write data to I/O boards.

The adapter block is a special block that is not connected to any other blocks in the SIMULINK block diagram. The purpose of the adapter block is to provide a reference to a particular I/O board. After we add an adapter block to our SIMULINK model, we can configure the I/O driver to match settings on the physical I/O board, and selecting settings that are software programmable. The RT In and RT Out blocks provide an interface to our physical I/O board. They ensure that the C code generated with real-time workshop correctly maps block diagram signals to the appropriate I/O channels. The RTWT uses the RT In block

for all input signals: analog inputs, and digital inputs. The RT Out block handles all output signals: analog outputs, and digital outputs. The real-time model of time-frequency technique is shown in Fig.8

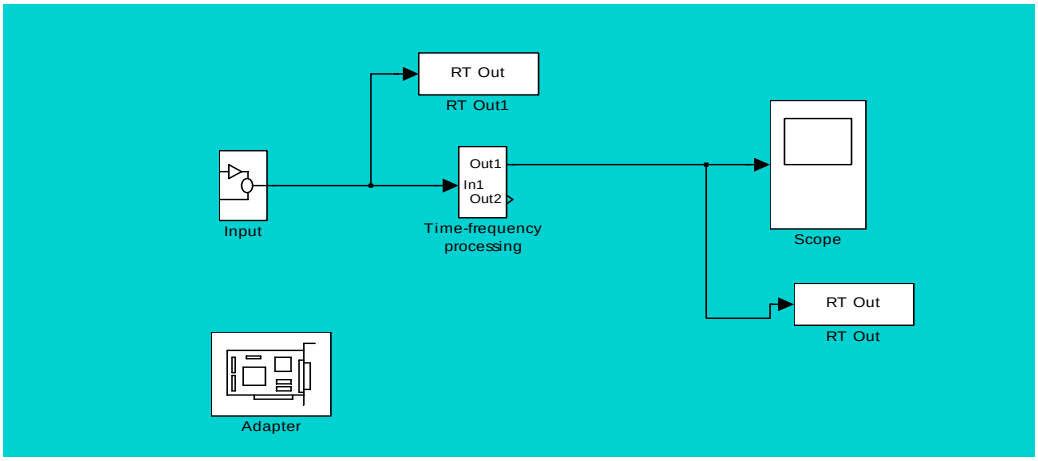


Fig. 8 Real time model of the time-frequency processing

The real-time representation of both the input and output of the time-frequency processing mentioned above in both clear and noise conditions are shown in Figures 9 and 10 respectively.

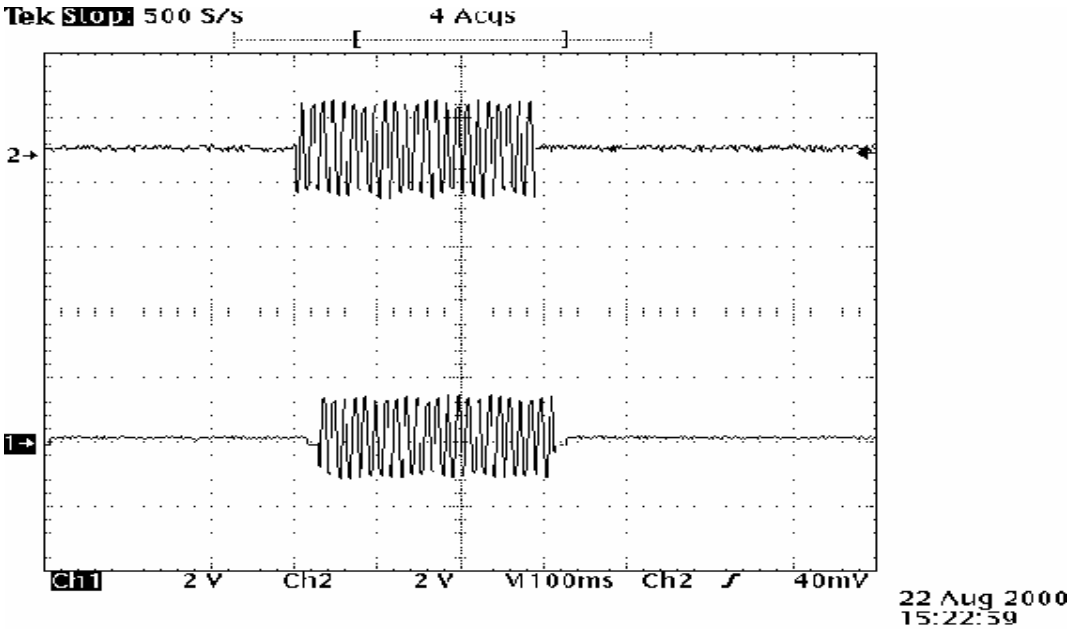


Fig. 9 Real-time representation of both the input and output of the time-frequency processing under clear conditions

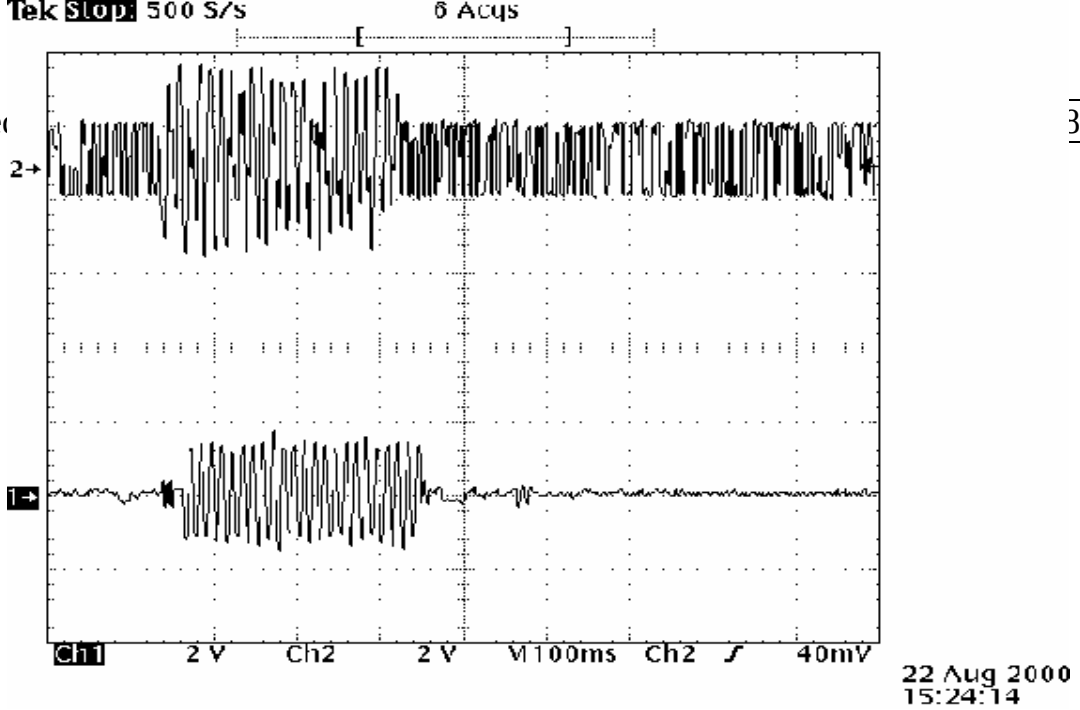


Fig. 10 Real-time representation of both the input and output of the time-frequency processing under noise conditions

V.CONCLUSION

In this paper a new technique to improve radar performance under jamming conditions is introduced. This new technique of time-frequency processing with a non-linear threshold level is applied to the chirp radar pulse imbedded in a high level noise. The purpose of the analysis is to detect, extract the signal and finally recover the radar pulse. This work is achieved by the model built using SIMULINK subroutine at MATLAB program. The recovered radar pulse has an excellent agreement with the original one.

This paper also introduces a real-time implementation of the proposed time-frequency model. The real-time implementation is achieved based on Lab-PC-12000 National instrument input/output data acquisition card using MATLAB, SIMULINK, the real-time workshop toolbox, and the real-time windows target toolbox. The measurement result has a good agreement with that obtained from the simulation model.

REFERENCES

[1] Boualem boashash, "Time-frequency signal analysis (Methods and applications)," Jon Wiley& Sons, (1992).

[2] R. A. Altes, " Wide-band, proportional bandwidth Wigner-Ville analysis," IEEE Trans Acoustic, Speech and Signal Processing, vol. 38, No. 6, pp. 1005-1012, June (1990).

[3] M. G. Amin, " Time varying spectrum estimation for a general class of nonstationary processes," Proc. IEEE, vol. 74, No. 12, pp. 1800-1802, Dec.(1986).

4] M. G. Amin., " Time and lag window selection in Wigner-Ville distribution," Proc. IEEE Int. Conference Acoustic, Speech and Signal Processing, 87, pp. 1529-1532, (1987).

[5] M. G. Amin., " Spectral smoothing and recursion based on the nonstationarity of the autocorrelation function," IEEE Trans. Acoustics, Speech and Signal Processing, vol. SP-39, No. 1, Jan. (1991).