A STUDY ON IDENTIFICATION OF RADAR EMITTERS

A MASTER'S THESIS

in

Electrical and Electronics Engineering

Atılım University

by

ASLANBEK M KILINÇARSLAN

APRIL 2011

A STUDY ON IDENTIFICATION OF RADAR EMITTERS

A THESIS SUBMITTED TO

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

OF

ATILIM UNIVERSITY

BY

ASLANBEK M KILINÇARSLAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

APRIL 2011

Approval of the Graduate School of Natural and Applied Sciences, Atılım University.

_____

Prof. Dr. İbrahim AKMAN

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Asst. Prof.Dr. Hakan TORA

Head of Department

This is to certify that we have read the thesis "A study on Identification of Radar Emitters" submitted by "Aslanbek Mustafa Kılınçarslan" and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Assoc. Prof. Dr. Ali KARA

Supervisor

Examining Committee Members

Assoc.Prof.Dr. Ali Kara _____

Asst. Prof.Dr. Hakan Tora _____

Asst Prof.Dr. Enver Çavuş _____

Date:

I declare and guarantee that all data, knowledge and information in this document has been obtained, processed and presented in accordance with academic rules and ethical conduct. Based on these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Aslanbek Mustafa Kılınçarslan

Signature:

ABSTRACT

A STUDY ON IDENTIFICATION OF RADAR EMITTERS

Kılınçarslan, Aslanbek M

M.S., Electrical and Electronics Engineering Department

Supervisor: Assoc.Prof.Dr. Ali Kara

April 2011, 69 pages

In this thesis, a methodology for identification of radar emitters with stable Pulse Repetition Interval (PRI) is studied. Also, a simulator for threat generation is developed for testing the system. Threat generation simulator generates Pulse Descriptor Words (PDW) that consists of Radio Frequency, Angle of Arrival, Pulse Width, Pulse Amplitude and Time of Arrival values. Also, jitter can be injected to these parameters except Pulse Amplitude, according to the user's input. The methodology consists of 3 parts. These are Clustering, PRI Estimation and Identification. The identified threats are given as output, after the incoming Pulse Descriptive Words are processed by these algorithms.

Keywords: Deinterleaving, PRI Transform, Clustering

ÖZ

RADAR YAYICILARIN KİMLİKLENDİRİLMESİ ÜZERİNE BİR ÇALIŞMA

Kılınçarslan, Aslanbek M

Yüksek Lisans, Elektrik-Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr.Ali Kara

Nisan 2011, 69 sayfa

Bu tezde kararlı Darbe Tekrarlama Aralıklarına (DTA) sahip radarların kimliklendirilmesi üzerine bir yöntem çalışılmıştır. Ayrıca, sistemi denemek için tehdit üretme simulatörü geliştirilmiştir. Tehdit üretme simulatörü, Radyo Frekansı, Geliş Açısı, Darbe Genişliği ve Darbe Genliğinden oluşan Darbe Tanımlama Kelimesini (DTK) üretir. Ayrıca Darbe Genişliği dışındaki parametrelere, kullanıcı tarafından girildiği taktirde gürültü de eklenebilir. Bu yöntem 3 kısımdan oluşmaktadır. Bunlar Kümelendirme, DTA çıkarımı ve Kimliklendirmedir. Gelen Darbe Tanımlama Kelimeleri bu algoritmalar tarafından işlendikten sonra, tanımlanan tehditler çıktı olarak verilir.

Anahtar Kelimeler: Ayrıştırma, DTA Dönüşümü, Kümelendirme

ACKNOWLEDGMENTS

I express sincere appreciation to my supervisor Assoc.Prof.Dr. Ali KARA for his guidance and insight throughout the research. Also, I would like to thank Levent Öktem, Suat Ekinci and Taner Kolçak from Synopsys's Turkey Office for providing me the working environment and for their support. To my wife, Kamuran, I offer sincere thanks for her continuous support and patience during this period.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

AOA	-	Angle of Arrival
CORDIC	-	Coordinate Rotation Digital Computer
EA	-	Electronic Attack
ECM	-	Electronic Counter Measures
ECCM	-	Electronic Counter Counter Measures
EP	-	Electronic Protection
ESM	-	Electronic Support Measures
EW	-	Electronic Warfare
FPGA	-	Field Programmable Gate Array
PA	-	Pulse Amplitude
PDW	-	Pulse Descriptor Word
PW	-	Pulse Width
RAM	-	Random Access Memory
RADAR	-	Radio Detection and Ranging
RF	-	Radio Frequency
RTL	-	Register Transfer Level
SMC	-	Synphony Model Compiler
SP&I	-	Signal Processing and Identification
TOA	-	Time of Arrival

CHAPTER 1

INTRODUCTION

In military operations it is important to know not only whether a radar emitter is watching you, but also its identity the radar emitters' parameters to determine whether it is a friend or foe. An Intercept Receiver is used for identification of radar emitters for this purpose. The received digital signals should be processed and then the detected emitters should be identified to warn the system operator. Also, this process should be fast enough for counter measuring against the threats. In this context, a design is implemented with MATLAB's m-language, and a limited hardware design is developed.

The aim of this thesis is to implement some signal processing function of an Intercept Receiver and realizing some of the signal processing part on a Field Programmable Gate Array (FPGA). It is assumed that digital signals are provided by an analog to digital conversion following the RF front end in a typical Intercept Receiver. In this study the

digital inputs to the identification algorithms are the Time of Arrival(TOA), Radio Frequency(RF), Angle of Arrival (AOA), Pulse Width (PW) and Pulse Amplitude (PA) values. These values are assumed to be taken properly and given as digital input to the system designed in this study. The output of the whole system is a table of threats detected or unknown. The identification system consist of three main parts. These are Clustering, Pulse Repetition Interval (PRI) Estimation and Emitter Identification section. This signal processing intensive identification system is developed on MATLAB and on its Simulink platform. First, the system is designed and simulated with the m-language of the MATLAB platform. Then, Clustering and PRI Estimation parts are implemented with some limitations on FPGA. The Synphony Model Compiler (SMC) toolbox on Simulink platform is used in this implementation.

The first part of the identification system is the Clustering. The Pulse Descriptor Words (PDW) are first given to the Clustering algorithm. This algorithm clusters the incoming PDWs according to the RF, AOA and PW values. After than each cluster is given to the PRI Estimation algorithm. PRI values are estimated for each cluster where more than one PRI value can be found in a cluster. Finally, the detected threats with PRI, RF, AOA and PW values are identified by comparing them with the emitters in the database. This system works only for stable emitters and the goal is to work in a dense environment where there may be many emitters with similar parameters. Also, an algorithm for threat generation is developed for testing the design. The emitters generated with this algorithm are given to the identification system. Threat Generation, Clustering, PRI Estimation, Identification and Hardware Implementation parts of the whole system are discussed in the following chapters.

CHAPTER 2

LITERATURE SURVEY

Nowadays, military forces are more dependent on Electronic Warfare (EW). Electronic Warfare is defined as a military action involving the use of electromagnetic energy to determine, exploit, reduce, or prevent radar use of electromagnetic spectrum [1]. Electronic Warfare can be divided into three main elements. These are Electronic Support Measures (ESM), Electronic Attack (EA) and Electronic Protection (EP). ESM involves actions of intercepting, identifying and locating of electromagnetic energy for supporting the military operations. EA involves actions to prevent or reduce a radar's effective use of electromagnetic spectrum. EP involves counter actions against EA. Intercept receiver has an important role in all ESM, EA, EP branches of EW. In this context the intercept receiver does the actions of identifying the emitters from the intercepted electromagnetic energy which is converted into digital signal and further analyzed for identifying possible emitters in a database. It is important to identify a

received emission source whether it is a friend or foe in a dense environment like litteral environments. This was the motivation of this study.

There are several thesis about threat identification in intercept receivers. One of them studied the clustering techniques [3]. Where it is implemented in C language. An emitter simulation software developed for this purpose. Emitters with constant, hopping, agile frequency and constant, fading pulse width values and stable, jittered, staggered PRIs are available in the simulation. Another one has the improvements done to the above study by using the TOA and PA values [4]. The implementation is in MATLAB. Also TOA deinterleaving algorithms CDIF and SDIF are discussed. In the next study, a wavelet transform is used for deinterleaving by using the TOA parameter [5]. In the last study a deinterleaving algorithm and its implementation on an FPGA is discussed [6].

In this study, while developing the clustering algorithm, the algorithms in [8] and [9] are implemented in clustering of interleaved pulse trains by using RF, AOA and PW. Moreover, the algorithm in [7] is implemented in TOA deinterleaving process. The algorithm proposed in [7] is stated as a complex-valued autocorrelation-like integral which leads to a kind of PRI spectrum wherein the locations of the spectral peaks indicate the PRI values.

CHAPTER 3

DEVELOPMENT OF ALGORITHMS FOR THREAT IDENTIFICATION

An intercept receiver comprises various sections. These are antennas, RF front ends, direction finding, analog to digital conversion and signal processing and identification. Signal processing and identification section processes the incoming PDWs. Its location in an intercept receiver is after the analog to digital conversion of the signals coming from the RF front end. In this study, it is assumed that the incoming analog emitter signals are intercepted by the RF front end following the antenna and converted from analog to digital form. Then, they are encoded as PDWs for processing in signal processing and identification section.

The Signal Processing and Identification consists of three main parts. These are the Clustering, PRI Estimation and the Identification parts. The incoming PDWs are clustered according to their RF, AOA and PW values. After clustering have been

completed, there may still be more than one emitter due to the structure of the Clustering algorithm. That is why, to identify the emitters in each cluster, the clusters are given to the PRI Estimation algorithm. After the parameters are estimated, the threats are identified by comparing the incoming threats with the emitters in the database. Also for simulation an algorithm that generates the threat parameters is developed. Generation of threat parameters, Clustering, PRI estimation and Identification of the design are discussed in the following sections.

3.1 Generation of Threat Parameters

The parameters that identify a threat are TOA, RF, AOA, PW and PA values in Signal Processing and Identification (SP&I). Threats are generated for testing and simulating the SP&I design by the algorithm for threat generation. The algorithm generates the TOA, RF, AOA, PW and PA values randomly according to the given inputs. The PA is not used with the SP&I design, however, it is generated by threat generation algorithm. The inputs; PRI values (p), the sigma values (δ) and the jitter for TOA (j_{TOA}) are given for the generation of TOA values. Jitter for RF (j_{RF}), AOA (j_{AOA}) and PW (j_{PW}), upper-lower inputs for RF (RF_u-RF_l), AOA (AOA_u-AOA_l) and PW (PW_u-PW_l) are given for the generation of RF, AOA and PW values. Sampling Frequency (Fs), minimum maximum values for each emitter (S_{min}, S_{max}), frequency of sinc (f_{sinc}) and scan period (λ) inputs are given for generating the PA values. The number of threats (N_t) and number of pulses (N_p) inputs are given for all TOA, RF, AOA, PW and PA values.

The number of threats (N_t) input given to threat generation algorithm determines the number of emitters to be generated. Each threat will have a PRI (p) value, and these PRI values are given as a row matrix to the algorithm. If a 5x1 [a b c d e] input is given to threat generation algorithm, each threat has the related PRI values respectively. As an example, if the number of threats is 3, the PRI of each threat may be as a, b and c respectively. If number of threats is 5, the PRI values may be given as [a b c d e]. If number of threats is 8, then the PRI values may be as a, b, c, d, e, a, b and c respectively. According to these PRI values, the TOA (t) values are created with the following equation,

$$t_{n+1} = t_n + p \qquad n = 0,1,2 \ldots N_p . \tag{1}$$

Here initial TOA values (time-offset) are determined randomly, and denoted by sigma (δ). According to the given PRI values, for each PRI value a random δ is set as t_0 value for each PRI. Also the number of pulses (N_p) input determines how many pulses to be generated for each threat.

A jitter value (j_{TOA}) is also calculated in parallel with the calculation of the TOAs. Before calculating the jitter, an interval for the jitter is determined. This helps to determine the boundaries of the jitter value. The interval (I) calculation for jitter is shown in (2).

$$I = \left(\frac{p\, j_{TOA}}{100}\right)/2 \tag{2}$$

Here the jitter percentage is the value given by the user. Then, a random number is generated between the $p-I$ and $p+I$. For example, if a 20% jitter to be added to the TOA values with a PRI value of 0.5, then the jitter interval becomes $0.5\mathrm{x}(20/100)/2$, or $\pm 1/20$. Finally, a random number between $-1/20$ and $+1/20$ is generated, and then added to the TOA sequence as p respectively.

The upper and lower values (RF_u-RF_l), (AOA_u-AOA_l) and (PW_u-PW_l) are also given as a input to threat generation algorithm. These upper and lower values determine the range of the RF, AOA and PW values. According to these values the random numbers in those ranges are generated. RF range may be between (10MHz-18GHz), AOA may in (0-359) degrees and PW in (0.1μs and 50μs).

The jitter for RF (j_{RF}) and PW (j_{PW}) values are given as percentage and for AOA (j_{AOA}), it is given in degree. An interval is determined according to this given inputs, and then a random number generated in the range. The boundary values for RF and PW are calculated according to the percentage. If 10% jitter value is given to the algorithm, then the results in the following equations are added to the RF or PW values as jitter.

$$I_{RF} = \pm\left(\frac{RF\ j_{RF}}{100}\right)/2 \tag{3}$$

$$I_{PW} = \pm\left(\frac{PW\ j_{PW}}{100}\right)/2 \tag{4}$$

The jitter value for AOA is provided in degree, therefore, the random jitter value that to be added to the AOA value calculated in degrees as shown in the following.

$$I_{AOA} = AOA \pm j_{AOA}/2 \tag{5}$$

Sampling Frequency (Fs), minimum maximum values for each emitter (S_{min}, S_{max}), frequency of sinc function (f_{sinc}), amplitude of sinc function (A_{sinc}) and scan period (λ) are given to the algorithm to calculate the PA values. The sinc function pretends the beam of the antenna with main and side lobes. First of all, time values are calculated by adding the $1/Fs$ to the minimum value until it reaches to the maximum value for each emitter. For example, assume that the minimum and maximum values are given as S_{min}=-1 and S_{max}=1 respectively and Fs is given as 1024. Then, the time values are calculated as shown in the following until it reaches to the maximum value (S_{max}).

$$t_{sinc}(n+1) = t_{sinc}(n) + \frac{1}{(Fs)} \qquad n = 0,1,2...(S_{max} - 1/Fs) \quad t(0) = S_{min} \tag{6}$$

Then t_{sinc} has a sequence; -1.0000,-0.9990, -0.9980... 0.9980, 0.9990. After the calculating of t_{sinc}, the PA values are calculated as sinc function shown in the following.

$$PA = A_{sinc}\, sinc(2\pi t_{sinc} f_{Sinc}) \tag{7}$$

These values are the amplitudes of the pulses in a train of an emitter. In Figure 1, the PA shapes are shown with Fs values of 1024, 64, 32 and 128, respectively. The minimum

and maximum values are -1 and 1. The amplitude of sinc (A_{sinc}) values are chosen as 0.3, 0.5, 0.9 and 1 respectively.

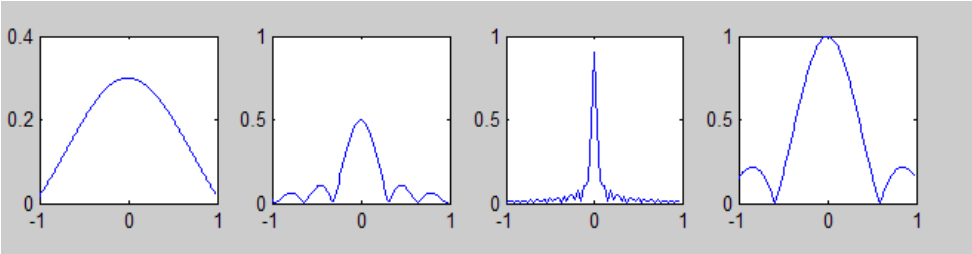

Figure 1. t_{sinc} vs. PA values

When these PA values are matched with TOA values, the pulse sequence in Figure 2 and 3 are obtained. In Figure 2, there are 64 PA values. This is calculated as (Fs)x2=32x2 from the 3rd pattern of Figure 1. Each of these PA values are matched to the generated TOA values in the next step. Also, Figure 3 shows the 4th pattern of Figure 1. There are 128x2=256 pulses according to the chosen Fs.

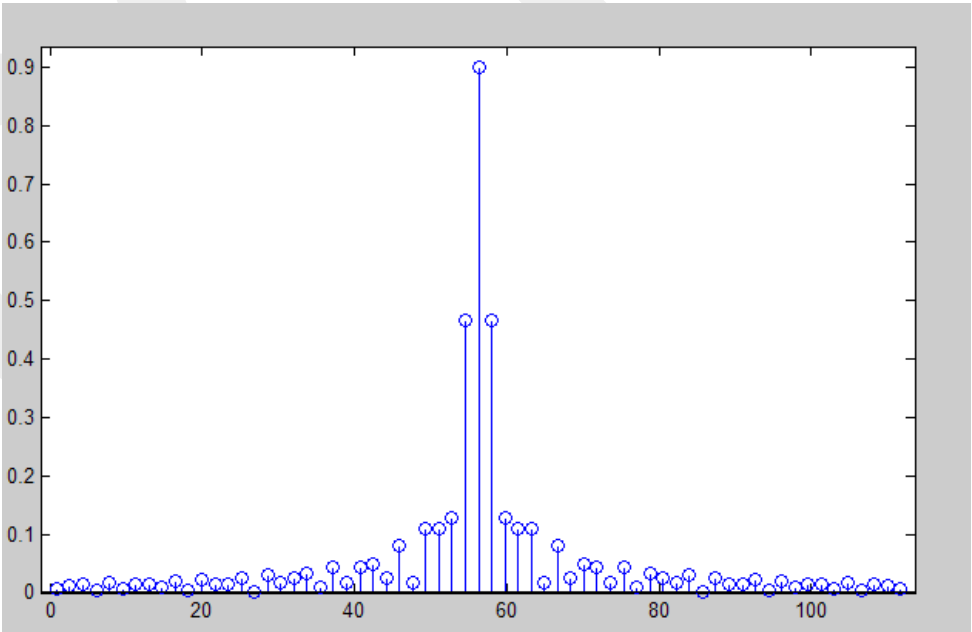

Figure 2. TOA vs PA plotted for the 3rd pattern of Figure 1

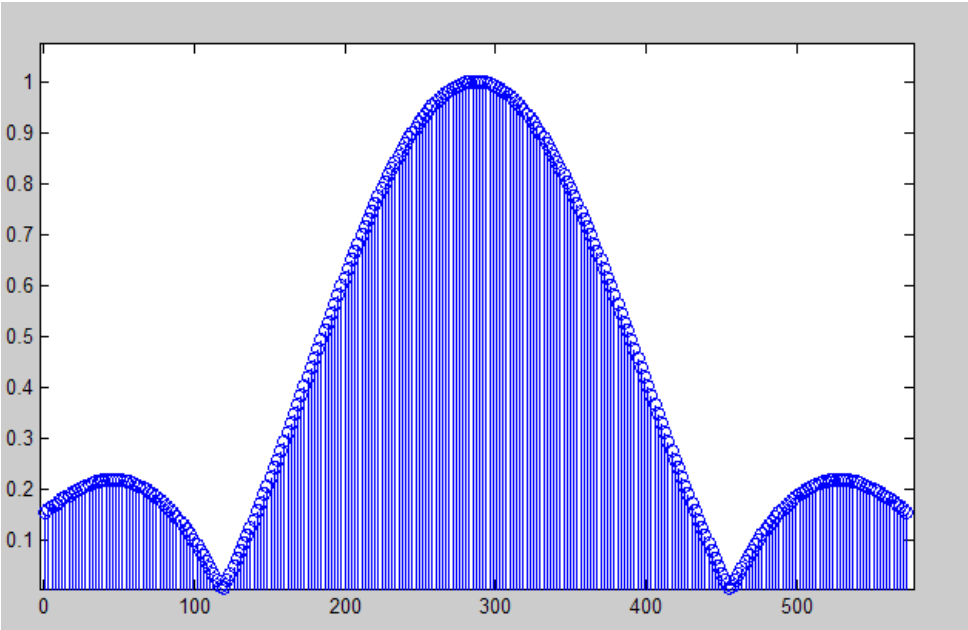

Figure 3. TOA vs PA plotted for the 4th pattern of Figure 1

Figure 4 is an example of the use of scan period λ which is another parameter in threat identification. The shape in Figure 2 repeats itself 4 times. Here, each repetition has 64 PA values. There is a total of 64x4=256 PA values. Later, these PA values are matched with TOA values according to the scan period of λ. In Figure 4, the scan period λ is taken as 30. Then, the TOA values are delayed with (Fs x2x λ) 64x30=1920 samples. Here, the antenna scan period is pretended by the λ given to the algorithm.

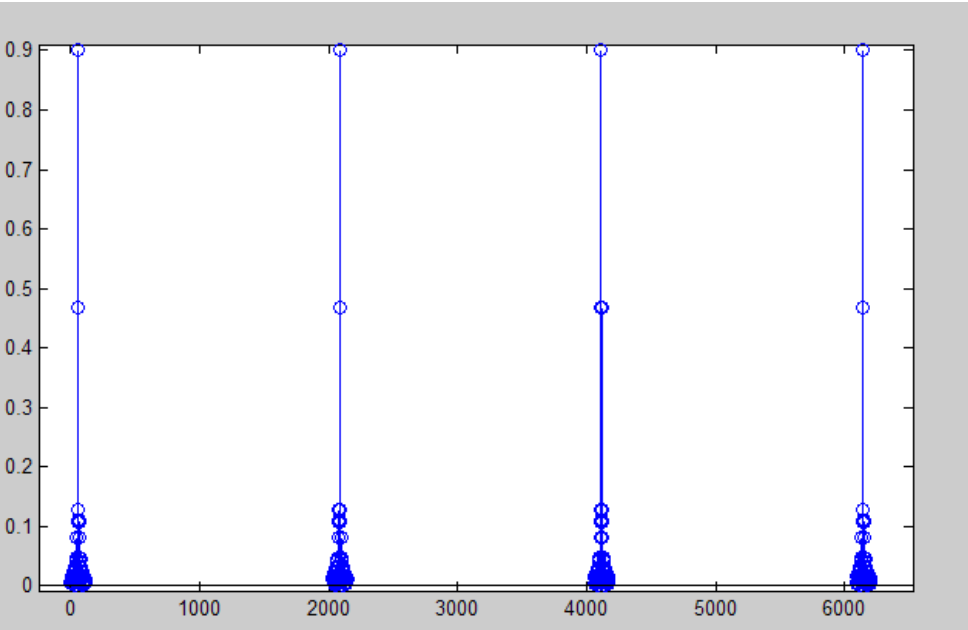

Figure 4. Simulated pulses for a scanning emitter of the 3rd pattern in Figure 1.

The inputs of threat generation algorithm and their descriptions are listed in Table 1.

Input to the algorithm	Description
PRI(p)	Pulse Repetition Interval
Number of pulses(N_p)	How many pulses to be generated for one emitter
Jitter for TOA(j_{TOA})	The percentage of the jitter that to be added to the TOA
Number of Threats(N_t)	How many different pulse trains to be generated
The sigma value(δ)	Determines the t(0) values for TOA (time offset)
RF upper- RF lower(RF_u-RF_l)	Determines the upper and lower bound for RF value
AOA upper- AOA lower (AOA_u-AOA_l)	Determines the upper and lower bound for AOA value
PW upper- PW lower(PW_u-PW_l)	Determines the upper and lower bound for PW value
Jitter for RF(j_{RF})	Percentage value for the jitter added to RF
Jitter for AOA(j_{AOA})	Degree value for the jitter added to AOA
Jitter for PW(j_{PW})	Percentage value for the jitter added to PW
Sampling Frequency(Fs)	Determines the number of samples available in sinc shape
Min max values for PA (S_{min}, S_{max})	Determines the interval that the sinc shapes to be plotted
Frequency of sinc (f_{sinc})	Used for generating the sinc function. (See Eqn. 6)
Scan period(λ)	Refers to antenna scan period
Sinc Amplitudes(A_{sinc})	Determines the amplitude of the sinc function

Table 1. The inputs and the descriptions

3.1.1 Description of the Threat Generation Algorithm

As a result of threat generation algorithm, the Pulse Descriptor Words can be created with the TOA, RF, AOA and PW values. The threat generation algorithm consists of mainly 4 different parts. These parts can be ordered as follows:

-Generation of Pulse Trains(TOA) and Pulse Amplitudes(PA) according to the given inputs

-Injecting of jitter to the Pulse Trains

-Generation of RF, AOA and PW values

-Matching Pulse Trains with RF, AOA, PW, PA values, and creating the PDWs

-Injecting of jitter to the RF, AOA and PW values, and sorting in TOA axes.

The blocks of the algorithm may be illustrated as in Figure 5 for threat generation, and its flow chart is also shown in Figure 6.

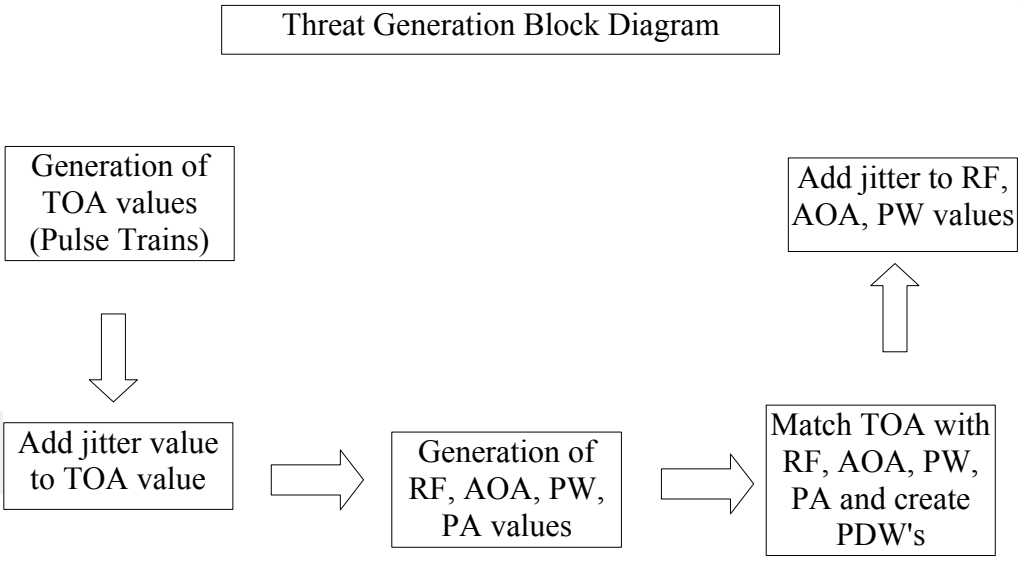

Figure 5. The block diagram for Threat Generation Algorithm

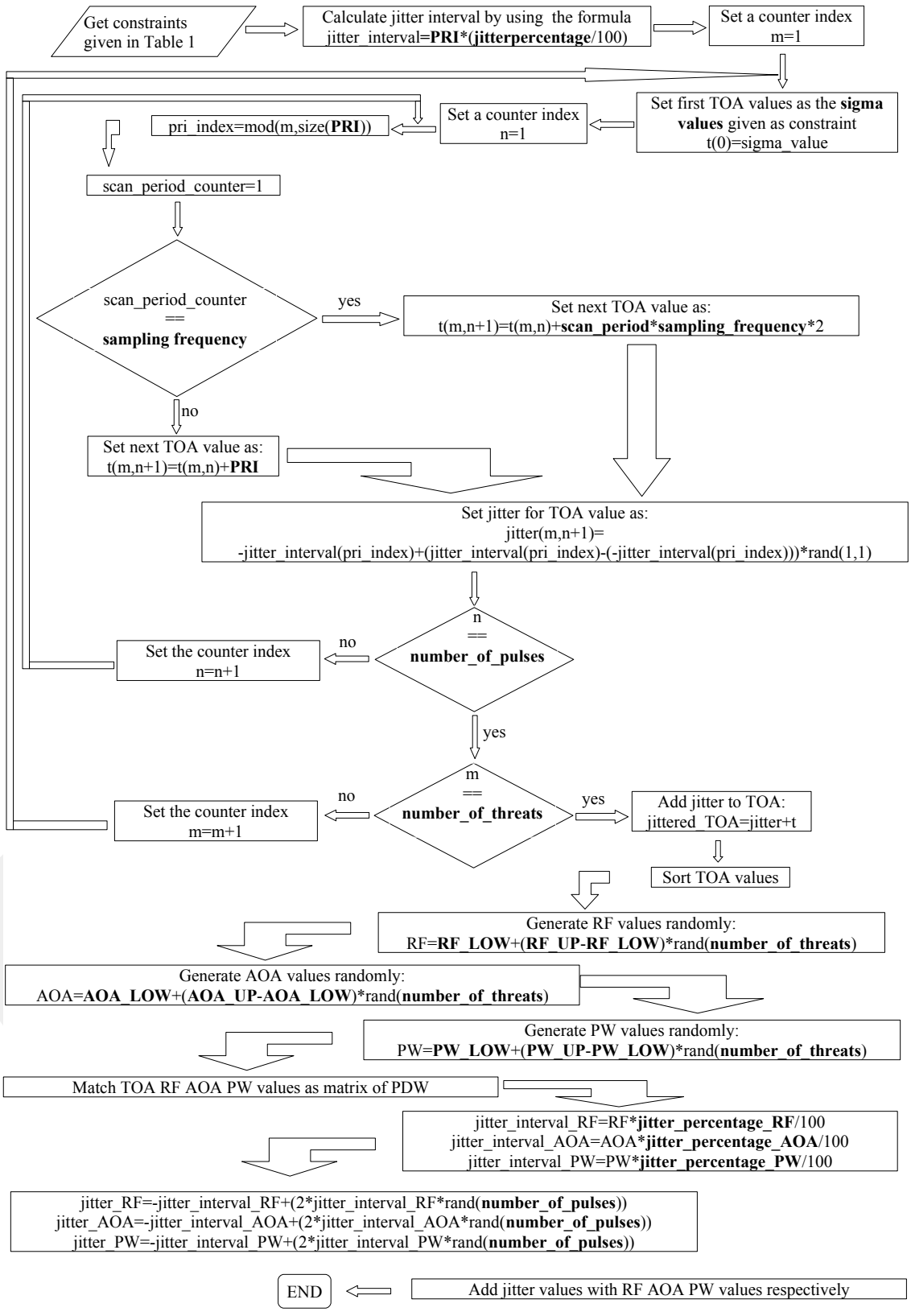

Figure 6.Flow Chart of the Threat Generation Algorithm

3.2 Clustering and Identification of Threats

The methodology of signal processing and identification method is shown in Figure 7. First of all, the incoming PDWs are clustered according to the RF, AOA and PW values. Secondly, each cluster is sent to PRI estimation algorithm. Finally, after all PRI values for each emitter in each cluster is determined, they are sent to the identification algorithm. The identified emitters and the detected threats are given as output. Each of the processes are discussed in the following subsections.

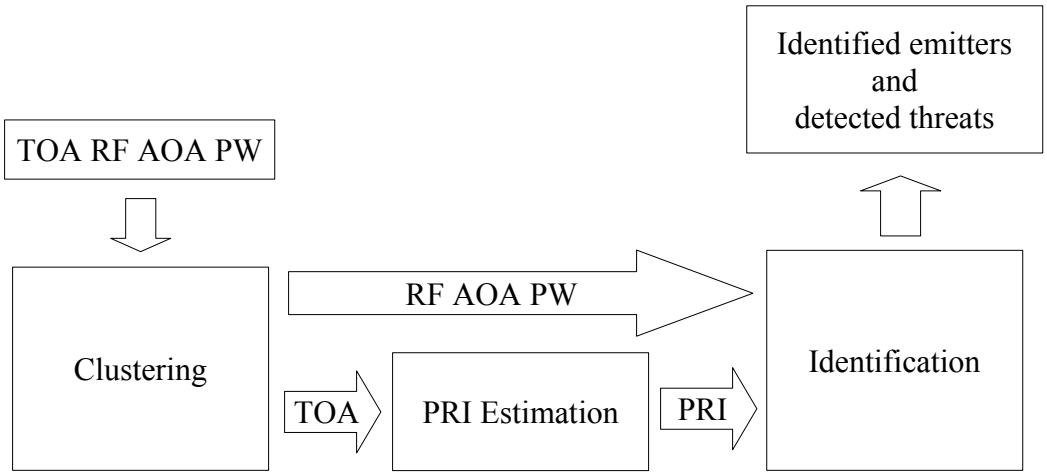

Figure 7. The methodology followed for identifiying threats.

3.2.1 Clustering Process

In clustering algorithm, the incoming pulse descriptor words are clustered according to the RF, AOA and PW values in three dimensional space. The algorithm used in the design is similar to the improved chain algorithm in [3], [4]. The improved chain

algorithm has the minimum complexity. However, the clustering performance is relatively less than the others [3], [4]. Since its performance/complexity ratio is reasonable, when compared to others, an algorithm similar to improved chain algorithm is developed and used in this thesis.

The input to the algorithm is the pulse train generated with the algorithm in previous section, and the output is the cluster data table and processed pulse table. The cluster data table consists of the cluster number, center of cluster and the cluster boundary values. The processed pulse table consists of the PDW and the cluster number. The cluster number can be used as a transition between the cluster data table and the processed pulse table.

The algorithm starts with checking whether there is any previously created cluster. If there is not, then the algorithm labels the first incoming PDW as the center of the first cluster with the RF, AOA and PW values. This is shown in the following.

$$C_{center} = \begin{bmatrix} RF \\ AOA \\ PW \end{bmatrix} \tag{8}$$

C_{center} is the center of the cluster. If there is a previously created cluster and the incoming PDW is falling into that cluster, then the center of the cluster is calculated by taking the mean of the C_{center}, and the incoming PDW. This may be shown as

$$C_{center} = mean\left(C_{center}, \begin{bmatrix} RF \\ AOA \\ PW \end{bmatrix} \right) \quad . \tag{9}$$

Each cluster has a boundary to be used for comparing whether an incoming PDW belongs to that cluster or not. A predetermined Δ value is needed to added and subtract from the center values. The cluster boundary is calculated as

$$C_{boundary} = \begin{bmatrix} RF_{center} \pm \Delta RF \\ AOA_{center} \pm \Delta AOA \\ PW_{center} \pm \Delta PW \end{bmatrix} \quad . \tag{10}$$

If there is a cluster already created, then, the RF, AOA and PW values are compared with the center values of the cluster is

$$\begin{matrix} 'Is\ current\ PDW \\ inside \\ Cluster\ Boundary' \end{matrix} = \begin{Bmatrix} RF < RF_{center} + \Delta RF \\ \wedge \\ RF > RF_{center} - \Delta RF \\ \wedge \\ AOA < AOA_{center} + \Delta AOA \\ \wedge \\ AOA > AOA_{center} - \Delta AOA \\ \wedge \\ PW < PW_{center} + \Delta PW \\ \wedge \\ PW > PW_{center} - \Delta PW \end{Bmatrix} \quad . \tag{11}$$

This comparison is done as follows; if the incoming PDW is inside these clusters' boundary values, it is assumed that this pulse belongs to that cluster, otherwise a new

cluster is created. This comparison is shown as a logical operation in (11). Then all new incoming pulses are compared with all previous clusters.

3.2.1.1 Distance Calculation in Clustering

If the incoming pulse belongs to more than one cluster, the distances are calculated from this incoming pulse to the center of all clusters that the pulse may belong to. Finally, the pulse added to the cluster that has minimum distance to the center, as the distance is formulated in the following.

$$D = \sqrt{((RF_{center}) - (RF))^2 + ((AOA_{center}) - (AOA))^2 + ((PW_{center}) - (PW))^2} \quad (12)$$

3.2.1.2 Description of the Clustering Algorithm

Clustering occurs by giving cluster numbers for each PDW. After all pulses are given a cluster number, in other words, after all of PDWs are clustered, the clusters are ready to be given to the PRI estimation algorithm. In some cases, there may be more than one emitter in one cluster. This case may arise when RF, AOA and PW values of different threats are so close to each other in a dense environment. Also the selection of Δ values for boundary calculation causes this. The effect of the Δ is to be discussed in detail under simulations and results section. After clustering process is completed, the clusters are given to the PRI Estimation algorithm to extract the emitters inside the cluster.

The procedure of the clustering algorithm is as follows:

1-Take each pulse and process one by one

2-Check whether a previous cluster is created, if there is not label this pulse as the first cluster.

3-If there are clusters already created, compare the incoming pulse with the clusters.

4- As a result of comparison, if the pulse does not belong to any previous cluster, create a new cluster.

5-If the pulse belong to one or more than one cluster, add the pulse to that cluster or the cluster that has the center with minimum distance to the pulse coordinates. Update the mean and boundary values of the cluster.

6-Do these steps from 1-5 until all the pulses are processed.

The flowchart of the algorithm is shown in Figure 8.

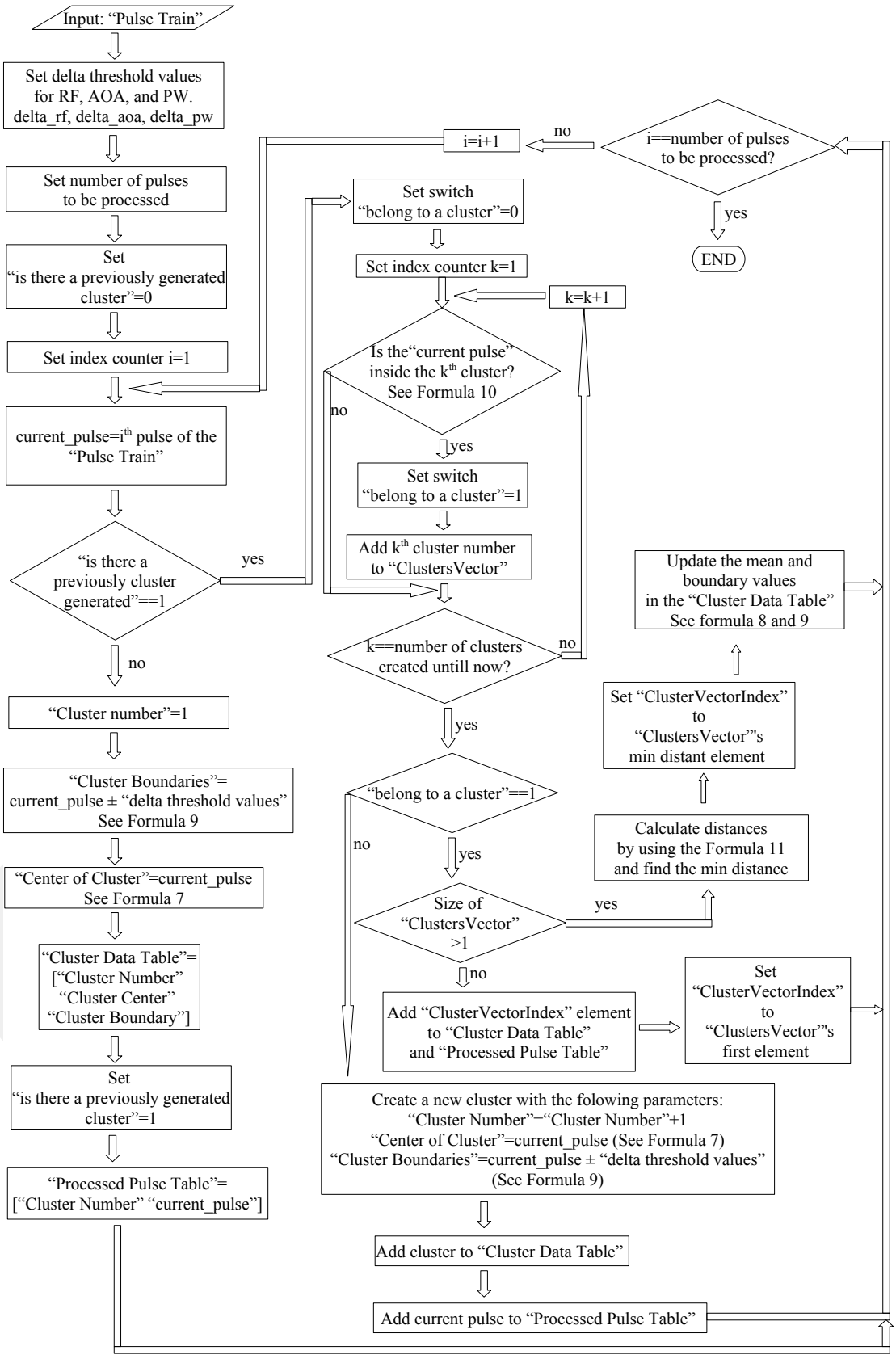

Figure 8. Flow Chart of the Clustering Algorithm

3.3 Estimation of Pulse Repetition Interval

The PRI value of the incoming pulse trains are estimated by using the TOA deinterleaving algorithm. The estimation algorithm is realized by using [7]. In [7], the proposed algorithm stated as a complex-valued autocorrelation-like integral which leads to a kind of PRI spectrum where spectral peaks correspond to possible PRI values. There are two distinct algorithms available; the first one is the PRI transform and the second one is the improved version of the first one. There are also CDIF and SDIF TOA deinterleaving algorithms. These algorithms suppress the subharmonics partially[7]. However, the algorithm in [7] suppresses the subharmonics successfully in the autocorrelation. The input to the algorithm is the TOA arrival values, and the outputs are the PRI spectrum and the autocorrelation function.

3.3.1 PRI Transform

In PRI Transform, the incoming pulse train captured in a TOA sequence that is denoted as t and it is investigated in a time interval between τ_{min} and τ_{max}. The interval is divided into K equal intervals that are called PRI bins. The width of the bins which is denoted by b are calculated by

$$b = \frac{\tau_{max} - \tau_{min}}{K} \quad . \tag{13}$$

The center of each bin (τ_k's), which to be used in the calculation of PRI transform, are calculated from

$$\tau_k = \left(k - \frac{1}{2}\right) b + \tau_{min} \quad k = 1,2,\ldots K \quad . \tag{14}$$

The center of bin (τ_k) and the width of the bins (b) are used for determining in which bin the difference of two TOA value t_n-t_m falls. The formulation of the discrete PRI transform is given as

$$D_k = \sum_{(m,n);\ \tau_k - \frac{1}{2} < t_n - t_m < \tau_k + \frac{1}{2}} e^{\left(\frac{2\pi i t_n}{t_n - t_m}\right)} \qquad k = 1,2,\ldots K \quad . \tag{15}$$

Here the t is the incoming TOA values and the n, m values are the indices that scan the TOA ranges. In the algorithm, the index n is incremented from 2 to the number of TOA values (N_p) that the PRI transform is calculated, and the index m is incremented from 1 to the index n. For example, if n is 2, 3,4 and 5, then for each value of n, the m is 1 and 1, 2 and 1, 2, 3 and 1,2,3,4 respectively. The k^{th} element of the PRI transform D_k is updated if the (t_n-t_m) value falls into the k^{th} bin. The exponential part of the D_k suppresses harmonics of the PRI values. The plot of the PRI Transform (D_k vs τ_k) is shown in Figure 9 as an example. The peaks correspond to the PRI values that are detected in the pulse train. In Figure 10, the result of the transform without exponential term, which corresponds to the autocorrelation function (C_k) is shown. It can be seen that the harmonics are not suppressed in the autocorrelation function. The autocorrelation function C_k is calculated as C_k=C_k+t_n for the PRI transform. It is similar to the

calculation of Dk, however, Dk has the exponential term. Also the calculation of the autocorrelation function (C_k) for the improved PRI transform is shown in (25).

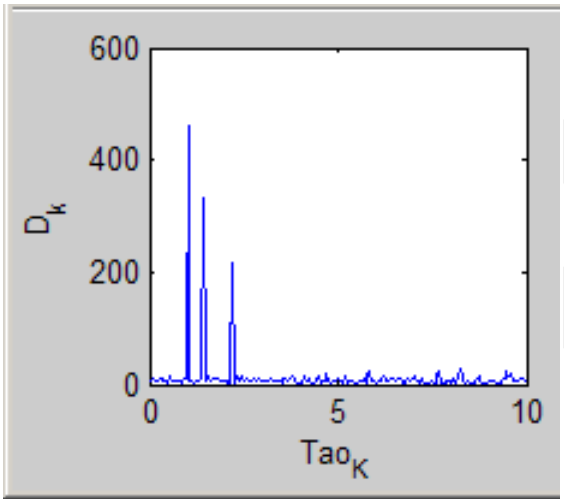

Figure 9: D_k vs τ_k

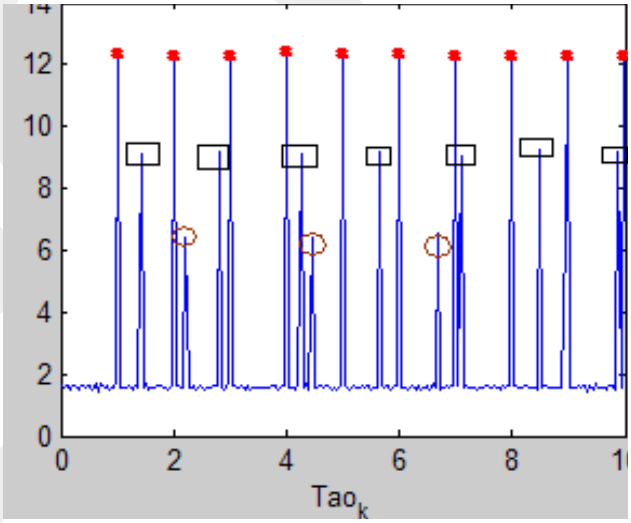

Figure 10: C_k vs τ_k

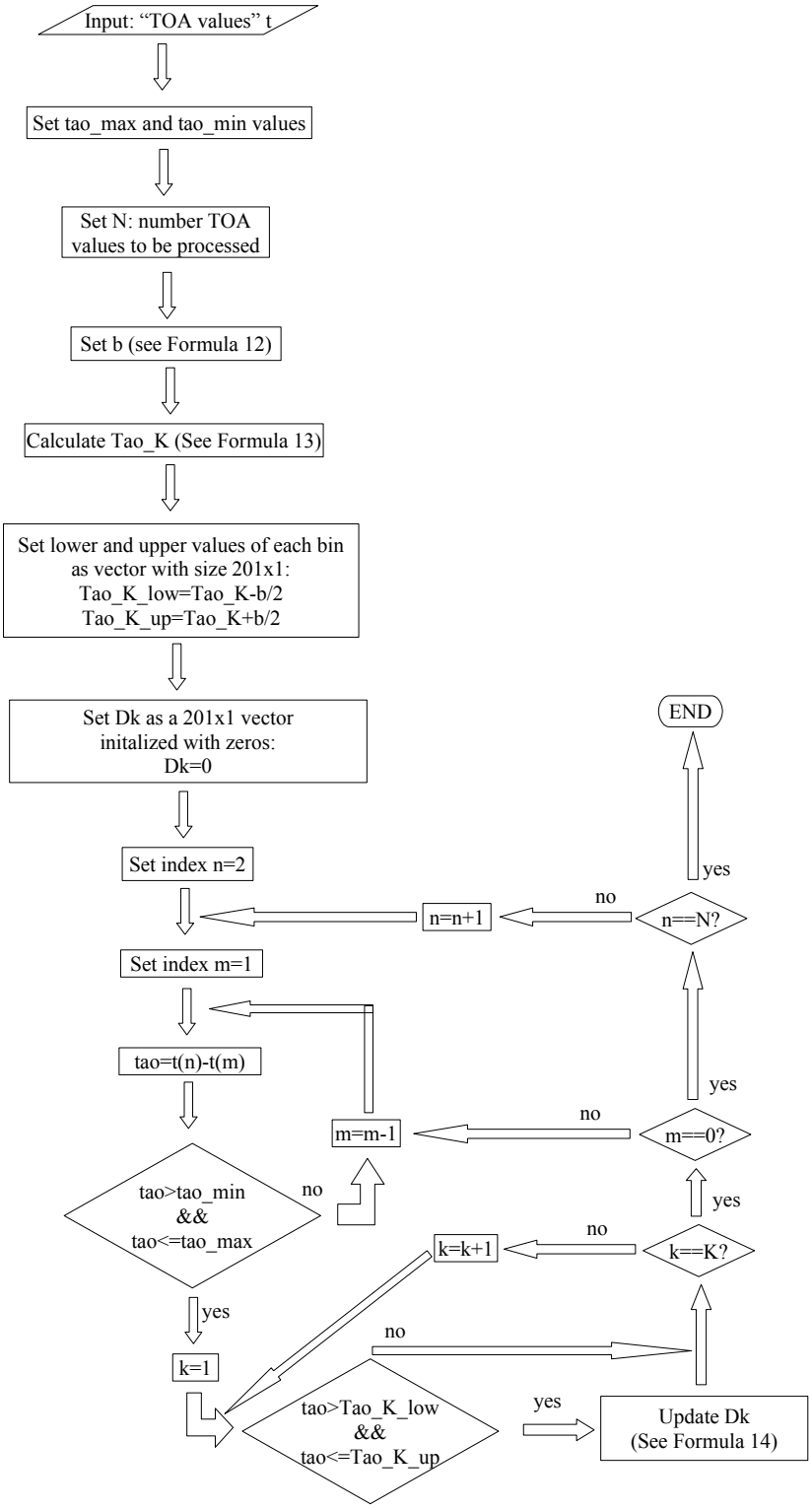

Figure 11. Flow Chart of the PRI Transform Algorithm

The two major improvements done in PRI Transform are the shifting of time origins and overlapped PRI bins [7]. The peaks are decreased due to the phase error in the original PRI transform. The phase error results from the growing of the TOA's from the origin. In PRI Transform the time origin is set t(0). In improved PRI transform, the time origins are shifted in the period of subharmonics and in the period of PRI to eliminate this bottleneck. Also, in PRI transform, the peaks are reduced due to the concentration of pulse pairs in several bins. In improved PRI transform, to overcome this situation, the overlapped PRI bins are introduced as an improvement.

Equations 16 and 17 shows the range of the PRI bins in improved PRI transform. The PRI transform is updated if the index k is between k_1 and k_2 values. These formulas are shown in the flowchart later. Here, the τ_{max}, τ_{min}, $\tau=t_n-t_m$ and K are same as in the PRI transform. The ϵ is a constant.

$$k_1=\left[\left(\left(\frac{\tau}{1+\epsilon}\right)-\tau_{min}\right)/\frac{(\tau_{max}-\tau_{min})}{K}\right]+1 \tag{16}$$

$$k_2=\left[\left(\left(\frac{\tau}{1-\epsilon}\right)-\tau_{min}\right)/\frac{(\tau_{max}-\tau_{min})}{K}\right]+1 \tag{17}$$

Formula 18, 19 and 20 are used for calculating the initial phase and decomposition of the phases. These values later to be used for deciding whether to shift the time origin or not.

$$\eta_{zero}=(t_n-O_k)/\tau_k \tag{18}$$

$$v=\eta_{zero}+0.4999 \tag{19}$$

$$\varsigma = \eta_{zero} / v - 1 \tag{20}$$

Equation 21 is used for calculating of the phase while equation 22 is used for calculating of the PRI Transform in [7].

$$\eta = (t_n - o_k) / \tau_k \tag{21}$$

$$D_k = D_k + \mathrm{e}^{(2\pi i \eta)} \tag{22}$$

After PRI transform is calculated, the PRI values are extracted by using the combination of 3 threshold functions. The combination of these functions are discussed in [7], and is given by

$$A_k = max\{\alpha \frac{\tau_{max} - \tau_{min}}{\tau_K}, \beta C_k, \gamma \sqrt{(\tau_{max} - \tau_{min})(\frac{N}{(\tau_{max} - \tau_{min})})^2 b_k}\} \quad . \tag{23}$$

Here, α, β and γ are constants for tuning each of the functions. The b_k is the width of the bin as described by.

$$b_k = 2\epsilon\tau_k \quad . \tag{24}$$

The autocorrelation function is calculated by

$$C_k = C_k + \tau \quad . \tag{25}$$

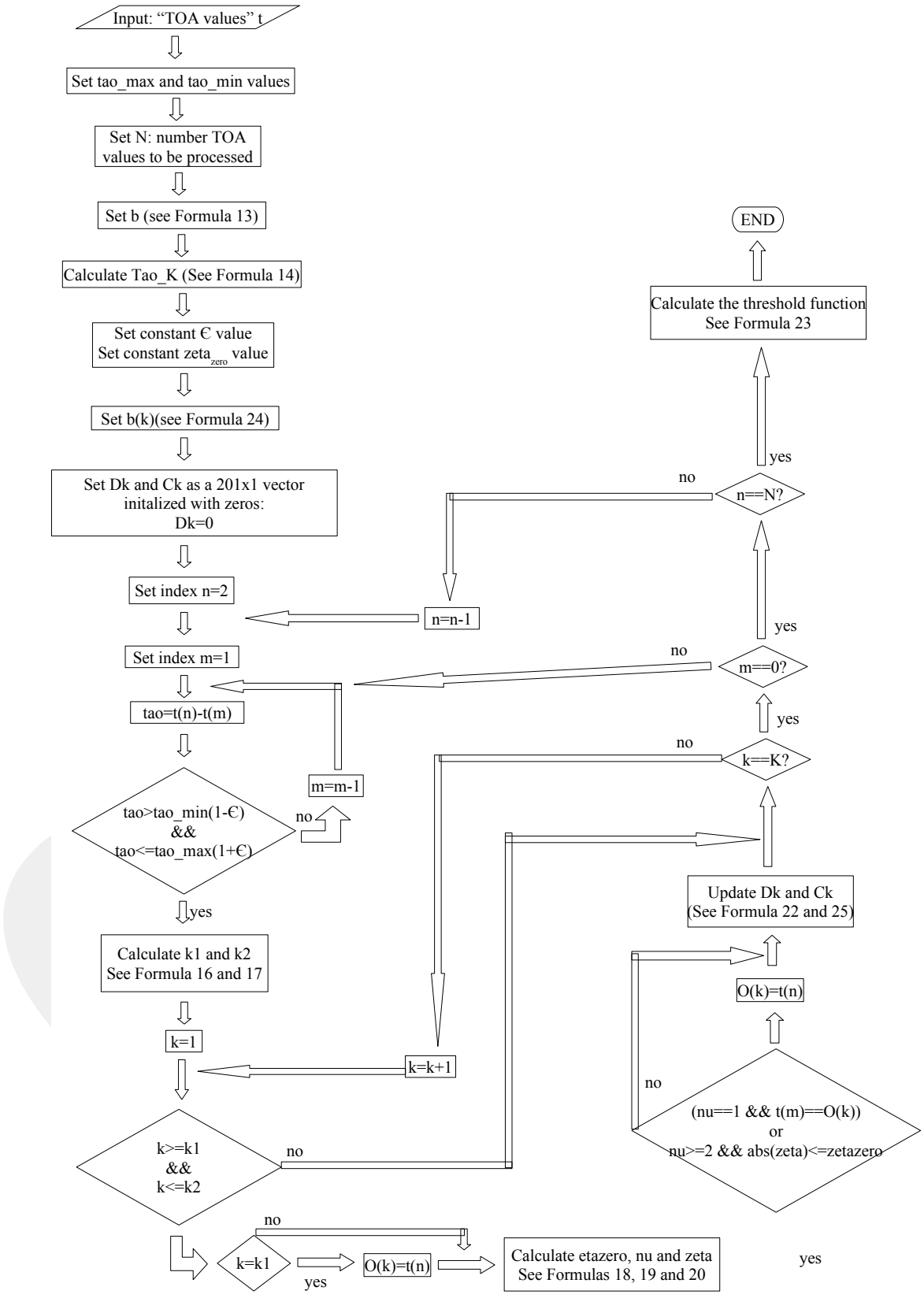

Figure 12. Flow Chart of the Improved PRI Transform

3.4 Identification

The input to the Identification algorithm is the threat list that are detected in the clustering and the PRI estimation algorithms. The detected threats with RF, AOA, PW and PRI values are compared with possible emitters available in a database. The minimum, maximum and mean values are calculated by considering the clustered pulses that has the same PRI values for RF, AOA and PW. For example, if a cluster has 20 pulses, maximum, minimum and mean values are calculated by using these 20 pulses. The values used in identification are listed in Table 2.

RFMin	RFMax	RFMean	AoAMin	AoAMax	AoAMean	PWMin	PWMax	PWMean	PRI

Table 2. Threat parameters used in identification process

Also an upper and lower value is calculated by adding and subtracting a constant value for the PRI of that threat. After an upper and lower value is calculated for the PRI, then the values are compared. If the means of the RF, AOA and PW values of the threats fall into minimum and maximum values of the emitter values in the database, then the threat is labeled as a known emitter previously recorded in the database. Otherwise, the threats are labeled as new or unknown. The outputs of the identification algorithm are either known threats or unknown (new) threats.

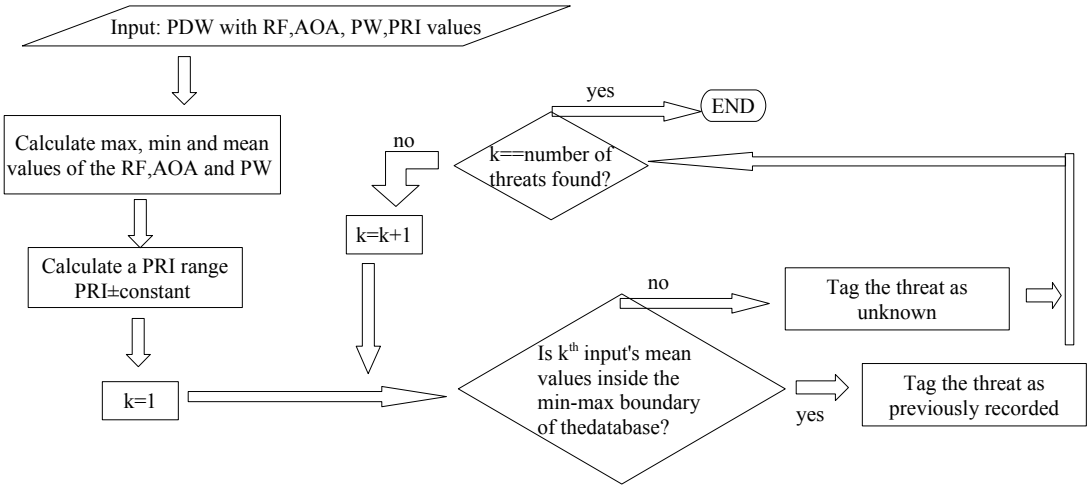

Figure 13. Flow Chart of the Identification Algorithm

3.5. Simulations and Results

In this section, the simulation results of the algorithm in MATLAB are presented. There are two scenarios considered in simulation. The first one is realized with small number of threats (N_t). The goal is to demonstrate how system works and to validate each subsystem. The second one is realized with larger number of threats which is 80. Here, the goal is to demonstrate the result of the whole system when there is larger number of threats, that is, dense emitter environment where emitters with smaller parameters exists.

3.5.1 Scenario 1

The input parameters for the first Scenario 1 is shown in Table 3. In this scenario, there are 3 threats with 3 PRI values. The PRI values are 1 √2 √5. The sigma values (time offsets) are generated randomly once, and then they are used for all other scenarios. The

number of pulses for each threat is taken as 1000. No jitter is considered in this case. RF upper and RF lower (RF_u-RF_l) values are selected in between 1GHz and 18GHz, AOA upper and AOA lower (AOA_u-AOA_l) values are selected between 0° and 359°, PW upper and PW lower (PW_u-PW_l) values are selected in between 0.1μs and 50μs, respectively. All values are listed in Table 3.

Threat Generation Algorithm Parameters-Scenario 1.1	**Values**
PRI(p)	[1 √2 √5]
Number of pulses(N_p)	1000 pulses for each emitter
Jitter for TOA(j_{TOA})	0
Number of Threats(N_t)	3
The sigma value(δ)	0,8147 0,9057 0,1269 (Randomly generated once)
RF upper- RF lower(RF_u-RF_l)	1e9Hz -18e9 Hz
AOA upper- AOA lower (AOA_u-AOA_l)	0°-359°
PW upper- PW lower(PW_u-PW_l)	0.1e-6s - 50e-6s
Jitter for RF(j_{RF})	0
Jitter for AOA(j_{AOA})	0
Jitter for PW(j_{PW})	0
Sampling Frequency(Fs)	[1024 64 256]
Min max values for PA (S_{min}, S_{max})	[-1 -1 -1] and [1 1 1]
Frequency of sinc (f_{sinc})	[0.15 0.5 3]
Scan period(λ)	[0 0 0]
Sinc Amplitudes(A_{sinc})	[0.5 1 1.8]

Table 3. The input parameters to threat generation algorithm for Scenario1.1.

In Scenario1.2, the only difference from the Scenario1.1 is the scan period values added and it is shown in Table 4. This case may show how the scan period affects the pulse train.

Threat Generation Algorithm Parameters-Scenario 1.2	**Values**
Scan period(λ)	[10 5 2]

Table 4. Scenario 1.2 - The input parameters that are different from Scenario1.1

In Scenario1.3, the jitter is introduced to the parameters shown in Scenario1.1. The parameters different from the Scenario 1.1 are shown Table 5 which has parameters for the Scenario 1.3. These are the 10% jitter for TOA, RF, PW and 5° jitter for AOA. All other input values are the same as in Table 3.

Threat Generation Algorithm Parameters-Scenario 1.3	**Values**
Jitter for TOA(j_{TOA})	10%
Jitter for RF(j_{RF})	10%
Jitter for AOA(j_{AOA})	5°
Jitter for PW(j_{PW})	10%

Table 5. Scenario 1.3 - The input parameters that are different from Scenario1.1.

3.5.1.1 Generation of Threats

In Scenario1.1, 1.2 and 1.3 shown in Tables 3,4 and 5, the PA shapes shown in Figure 7 are generated as discussed in Section 3.1.5. There are total of 1024x2 64x2 and 256x2 samples for each PA pattern as shown in Figure 14, respectively.

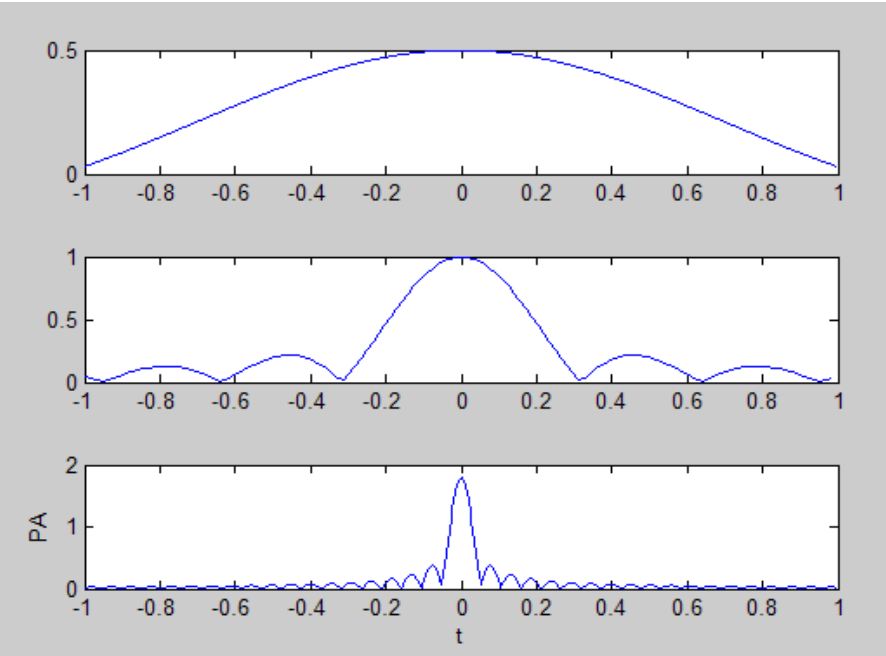

Figure 14. Various sinc shaped antenna pattern of threats

In Figure 15, it is demonstrated how these all sinc shaped pulse trains for Table 3, 4 look like in a mixed sequence that models pulse sequences of three different emitters. The first sinc shape in Figure 14 has 1024x2 that is 2048 samples and the Number of Pulses (N_p) value for each threat is 1000. That is why, the first 1000 sample of the first sinc shape takes place in the mixed pulse train shown in Figure 15 with the arrow numbered 1. Since, each threat has 1000 TOA values, the total number of samples in Figure 15 is 3000. The second sinc shape in Figure 14 is shown with the arrow number 2 in Figure 15. When the Number of Pulses is 1000 and the Sampling Frequency is 64 for the second shape in Figure 14, this corresponds to 64x2=128 samples for 1 sinc shape that repeats itself 8 times as shown in Figure 15. Also the third sinc shape repeats itself 2 times, since it has 512x2=1024 samples. One another thing is that, the last 24 samples of the third sinc shape are not shown in Figure 15, because the Number of Pulses (N_p) is selected as 1000.

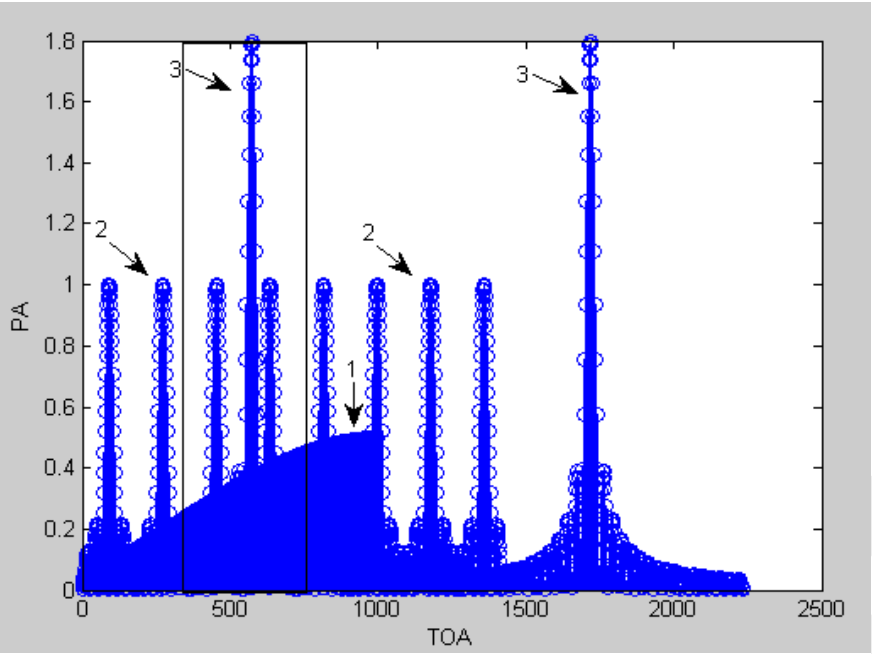

Figure 15. The mixed pulse trains on the same TOA axis vs PA

In Figure 16, the portion in the rectangle in Figure 15 is zoomed to show the pulses easily and, in detail.

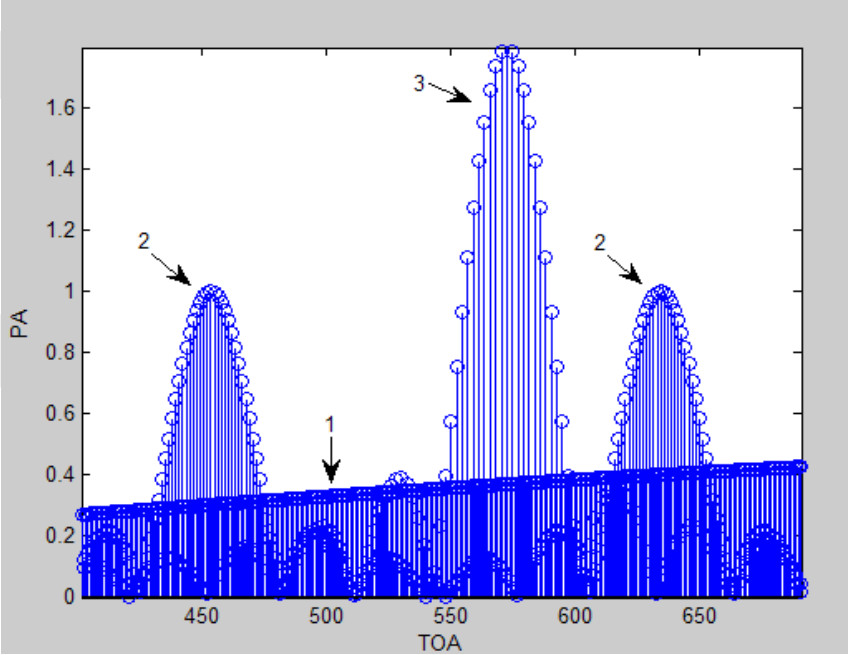

Figure 16. The zoomed version of Figure 8

If the scan period values are given to threat generation algorithm as in Table 4, then the pulse train is as shown in Figure 17.

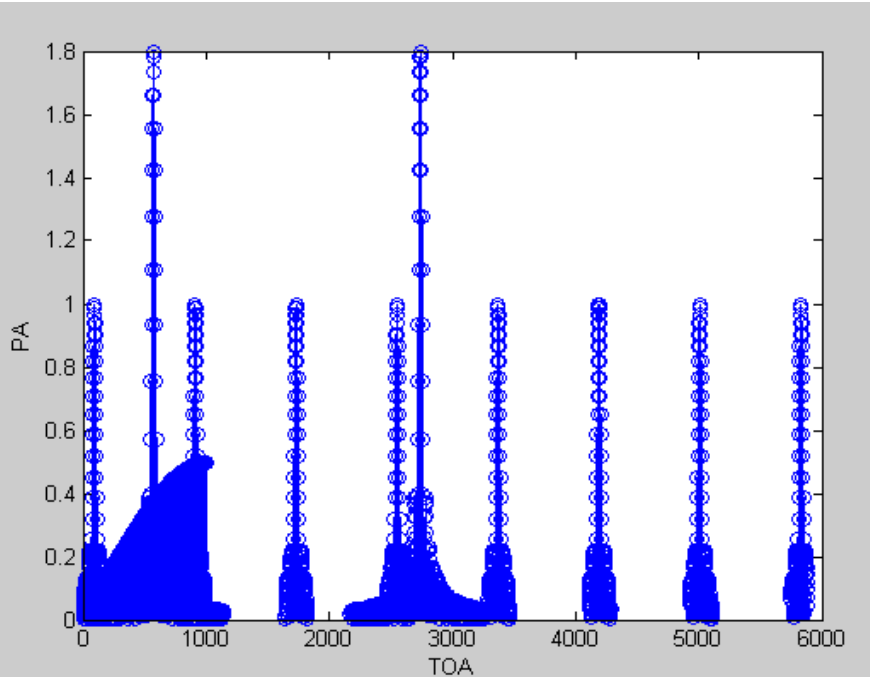

Figure 17. The pulse train with nonzero scan period (rotating antenna)

In Figure 17, there are gaps between the sinc shapes. For example, the length of the gaps between the sinc shapes that has the peak pulse amplitude of 1 is 5x64x2=640 samples. In every 640 samples, the sinc shape repeats itself.

If the threats are generated with the inputs shown in Table 3, the result shown in Figure 18 is obtained for the RF, AOA and PW values. Here the 3000 RF AOA and PW values are scattered in the figure. It seems, as only 3 points in space, because the jitter is zero in this case. However, when jitter is introduced as in Table 5, the resulting distribution becomes as in Figure 19.

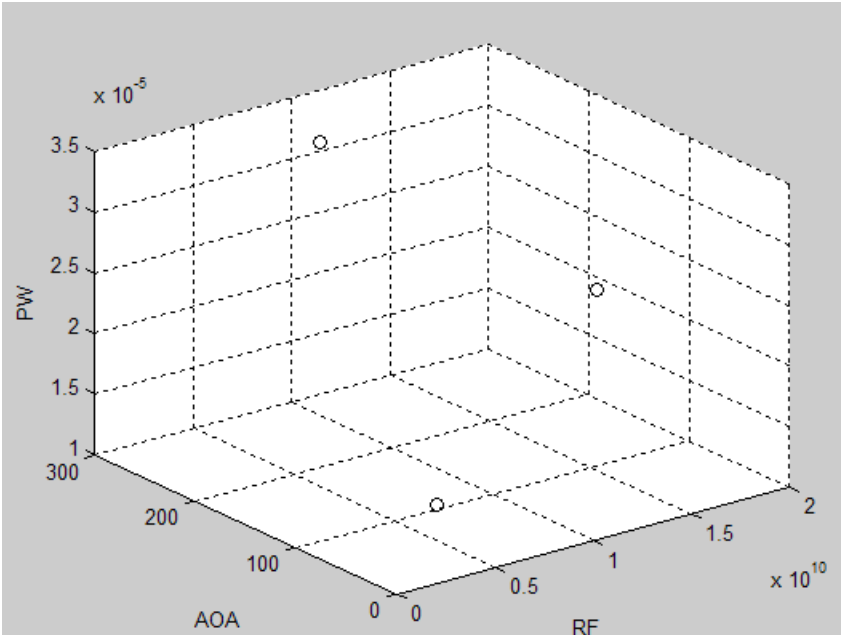

Figure 18. The 3000 RF, AOA and PW values with no jitter.

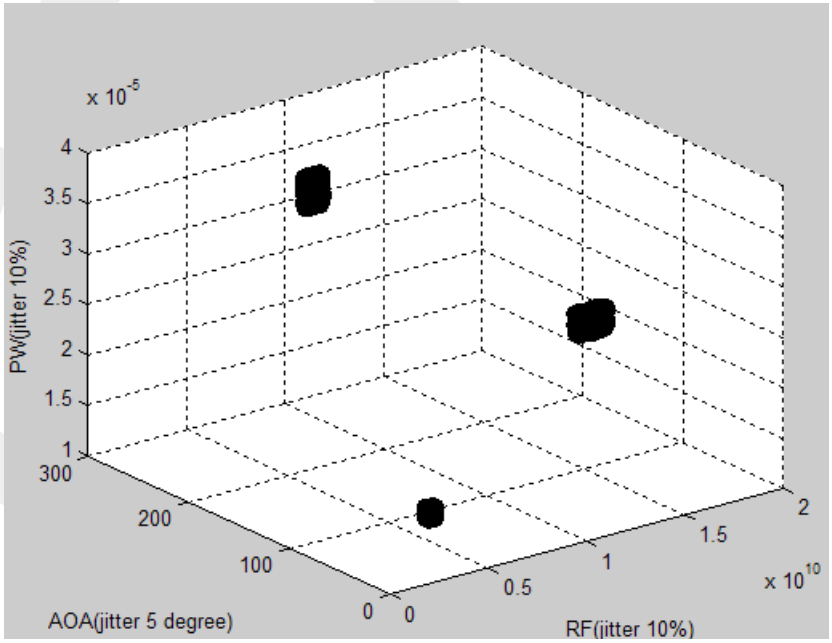

Figure 19. The 3000 RF, AOA and PW values with jitter (as shown in Table5)

In Figure 19, the 3000 points are not seen as one point due to the jitter. Each RF AOA and PW values can bee seen clearly in Figure 20 which is the zoomed version of Figure 12.

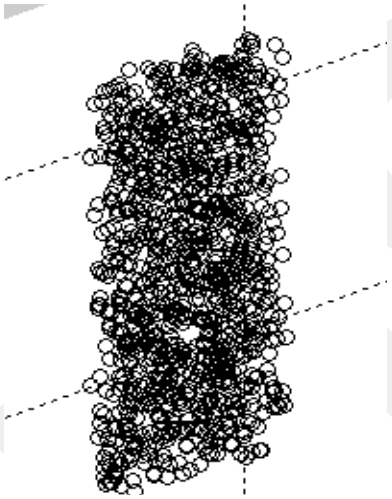

Figure 20. The zoomed version of Figure 12

3.5.1.2 Clustering

The generated threats with threat generation algorithm according to the Scenario 1.3 shown in Table 5, are given to the clustering algorithm. In clustering algorithm the important parameters that affects the results are the cluster boundary parameters shown in Table 6.

Parameters for Cluster Boundary	Case 1	Case 2	Case 3
ΔRF	3e9	5e8	11e9
ΔAOA	30	2	150
ΔPW	1e-5	1e-6	2.5e-5

Table 6. The parameters used for calculating cluster boundary.

There are 3 cases in Table 6. Those values are selected for showing the effect of the delta values to clustering process. In Table 7, the approximate RF AOA and PW values for threats 1, 2 and 3 are shown. Also, the minimum and maximum values of jitter added in Scenario1.3 are available in Table 7.

	Threat 1	Threat 2	Threat 3	Min jitter	Max jitter
RF	15.5e9	4.69e9	9.02e9	-0.775e9	0.773e9
AOA	103.78	51.57	253.08	-2.49	2.49
PW	2.42e-5	1.32e-5	3.36e-5	-0.168e-5	0.168e-5

Table 7. Min-Max jitter and Min-Max RF, AOA and PW values

In case 1, the parameters for calculating the cluster boundary values are chosen such that, they are greater than the jitter value in Table 7 and they are less than the maximum difference value between the threat 1 and 2, the threat 2 and 3 and the threat 1 and 3. For example, the ΔAOA value for Case 1 is 30 and this value is greater than the maximum jitter value 2.49. Also, the AOA differences between threats are; between threat 1 and 2 it is 103.78-51.57=52.10, between threat 1 and 3 it is 253.08-103.78=149.30, and between threat 2 and 3 it is 253.08-51.57=201.51. The ΔAOA which is 30 is less than

these differences between threats 1, 2 and 3. The ΔRF and ΔPW values are also similar with the selection of ΔAOA.

The result of the clustering algorithm according to case 1 is shown in Figure 21. The incoming pulse train, generated with input parameters according to the Table 5, is clustered into 3 clusters.

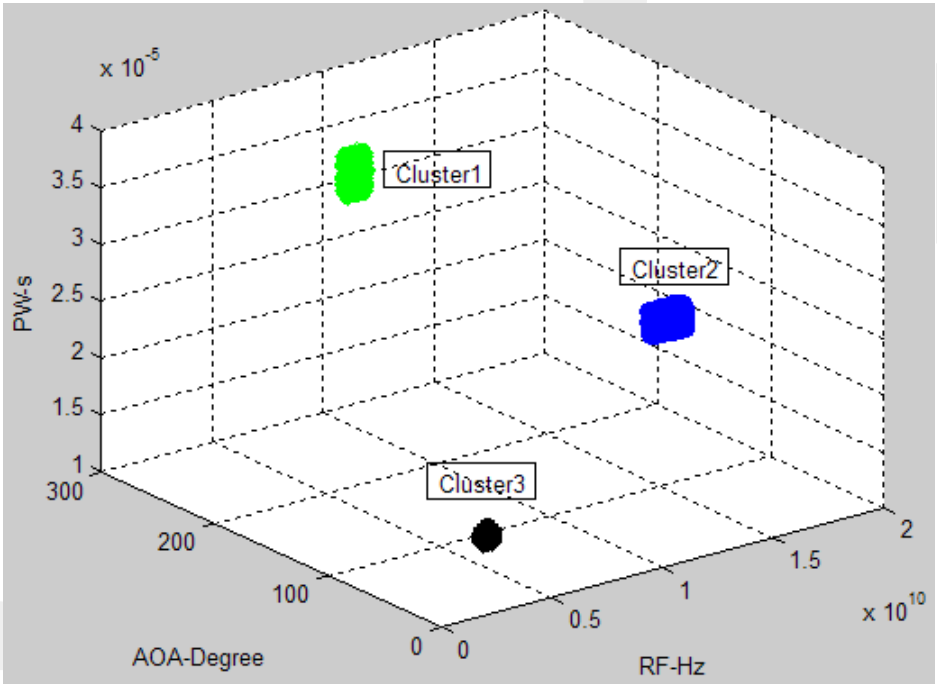

Figure 21. Case 1 clustering results (3 clusters)

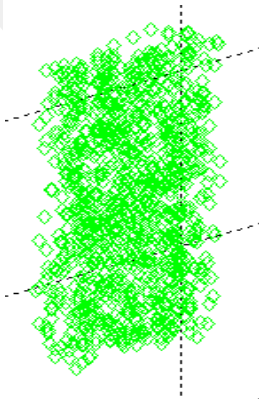

Figure 22. Cluster1 with diamond shapes

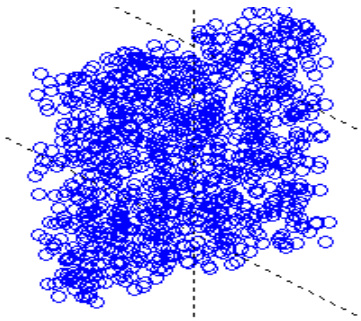

Figure 23. Cluster2 with circle shapes

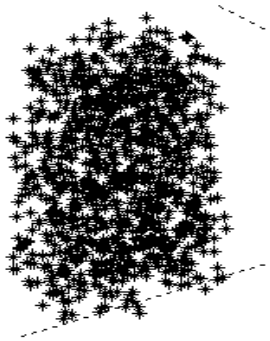

Figure 24. Cluster3 with star shapes

In Figures 22, 23 and 24, the zoomed versions of the Clusters 1, 2 and 3 are shown. In Case 2, the parameters for cluster boundary are between the minimum and the maximum jitter value. In this case, the results are as shown in Figure 25.

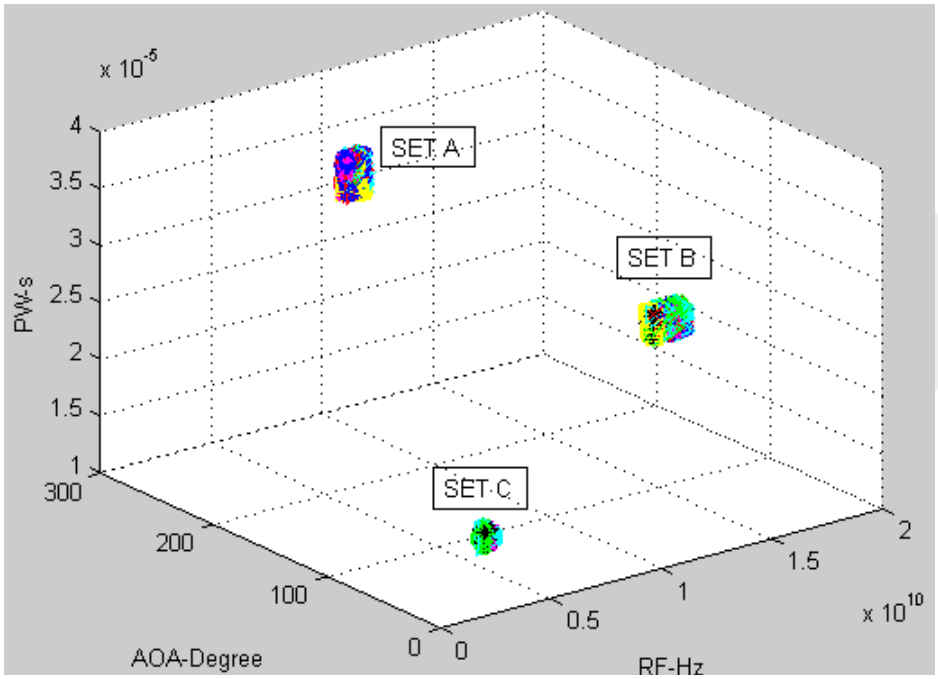

Figure 25. Case2 clustering results (42 clusters)

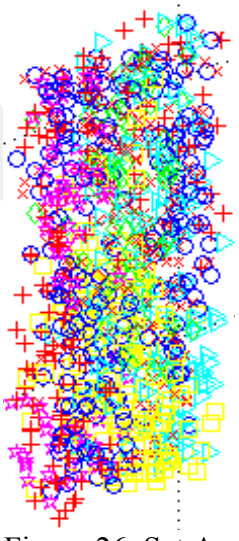

Figure 26. Set A

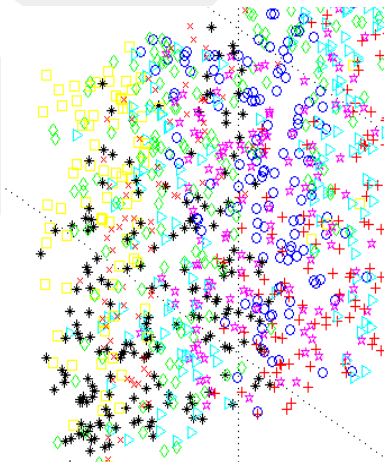

Figure27. Set B

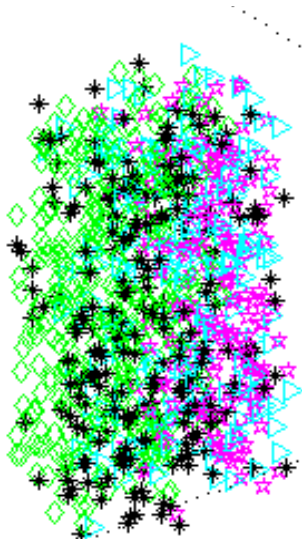

Figure 28. Set C

In Figure 26,27 and 28 the zoomed versions of the Sets A, B and the C are shown. The input pulse train is divided into 42 clusters, in here. In case 3, the parameters for cluster boundary are greater than the differences of the threats among each other. The result is shown in Figure 29.

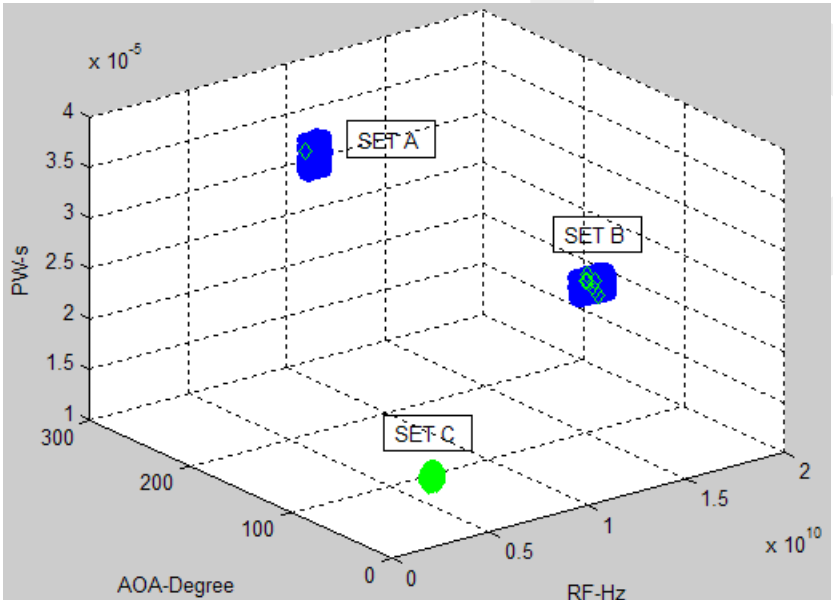

Figure 29. Case3 clustering result (2 clusters).

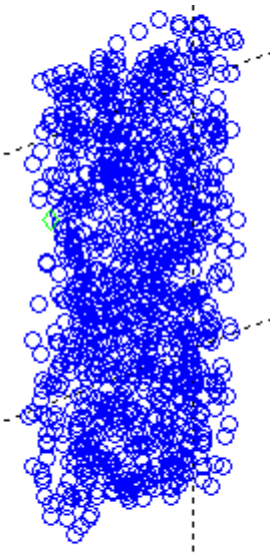

Figure 30. SET A

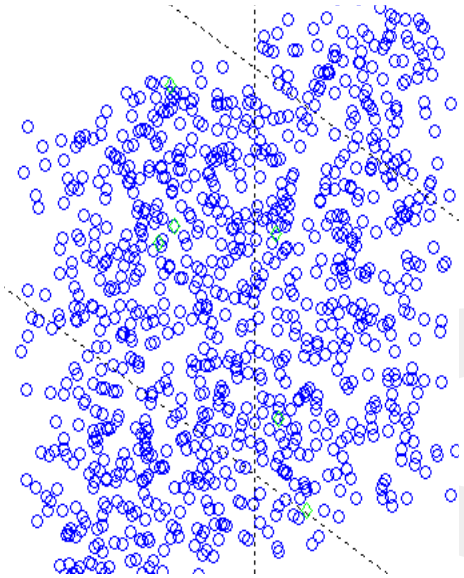

Figure 31. SET B

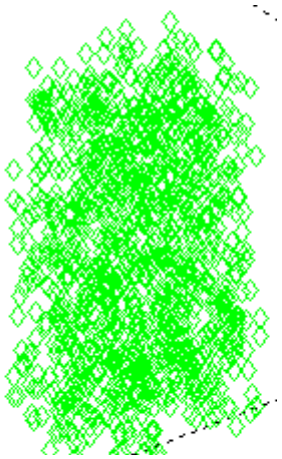

Figure 32. SET C

The input pulse train is divided into 2 clusters. The first cluster is shown in circle shape and the second cluster is shown in diamond shape. As a result, the boundary values in the Case 1 has the best results among 3 cases in clustering. If these values are chosen such that they are greater than jitter values and less than the differences among threats, then better result can be obtained in clustering process.

3.5.1.3 PRI Estimation

In this section, the results of the PRI estimation algorithm are introduced. The incoming pulse trains are given to both PRI transform and the improved PRI transform to show the difference between them. In the whole system, the clustered pulse trains are given to the PRI transform algorithm as an input. In Table 3 the Scenario1.1 is shown and in Table 6, the Case 1 is shown for clustering parameters. As a result of clustering algorithm, 3 clusters were found and each cluster have 1000 pulses. In Figures 33, 34 and 35, the

results of the PRI transforms are shown. Here the TOA jitter is zero. The peak values are gathered at PRI values √5, 1 and √2, respectively.

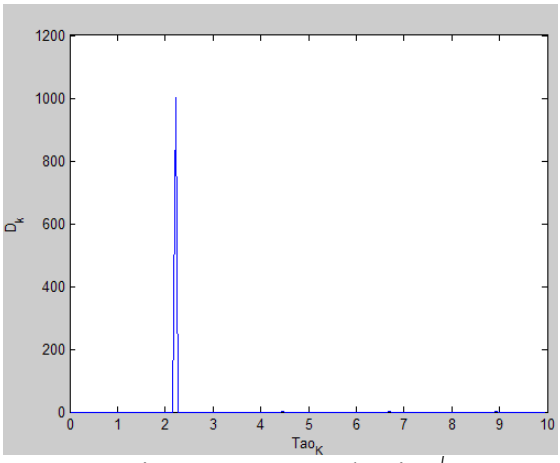

Figure 33. PRI value is √5

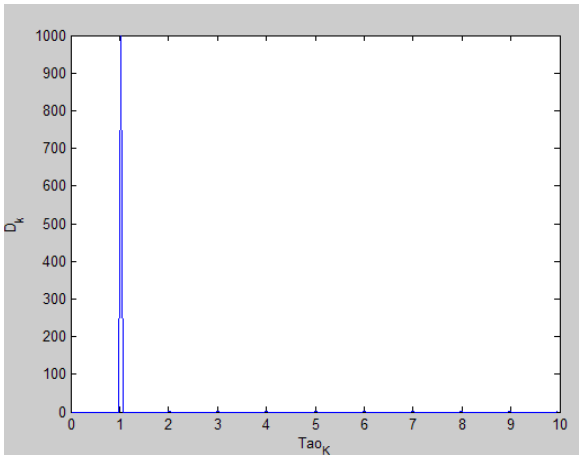

Figure 34. PRI value is 1

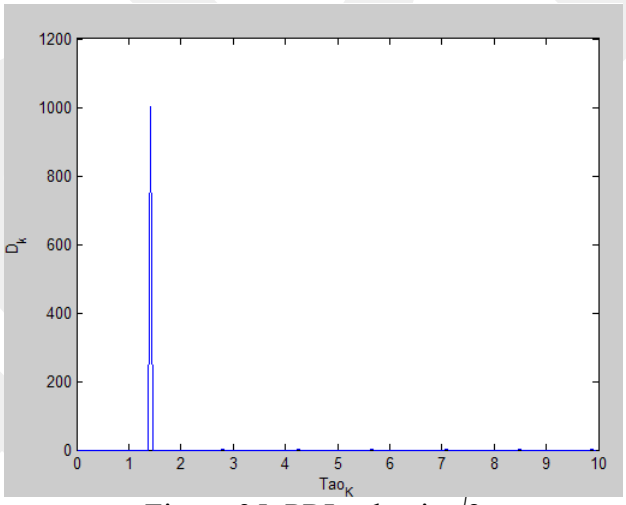

Figure 35. PRI value is √2

To show the difference between the original and the improved PRI transform algorithms, the input train that contains three of the PRI values are given to the PRI estimation algorithm. The first 1000 pulses from the pulse train generated by threat generation algorithm according to inputs in Scenario 1.1 and Scenario 1.2 are given to the PRI

Transform algorithm. The pulse train generated according to Scenario 1.1, when there is no jitter, the result of the original PRI transform is shown in Figure 36. The PRIs are 1, √2 and √5, and all the PRI values can be seen clearly.

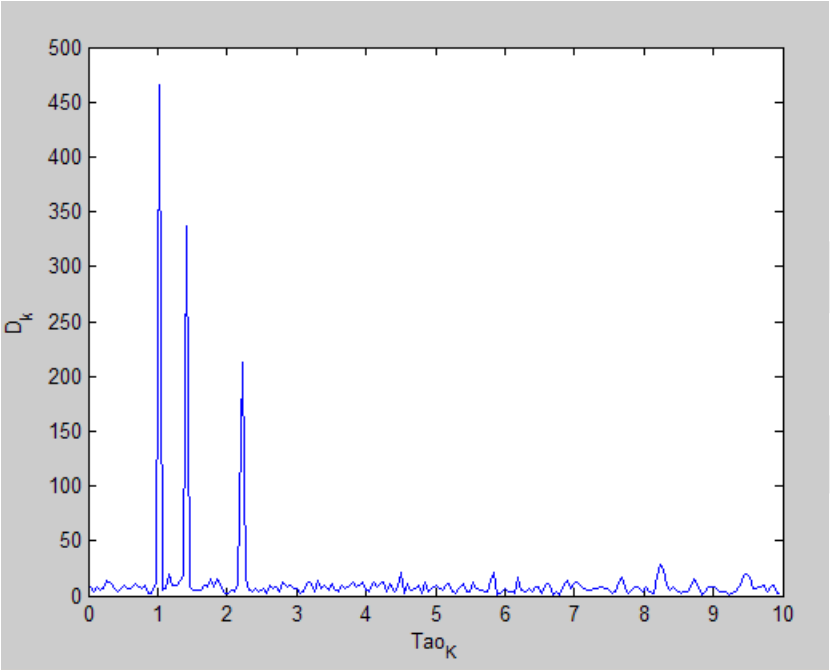

Figure 36. PRI transform of 1000 pulses from Scenario1.1. No jitter added to the TOA values.

In Figure 36 the PRI transform of the 1000 pulses from Scenario1.1 is shown. There is no jitter in the incoming pulse train and the peaks are gathered at PRI values. However, if jitter is added to the incoming pulse train, the original PRI Transform is vulnerable to jitter [7]. In Figure 37, according to Scenario 1.2, 10% jitter is added to the same pulse train and the PRI's can not be extracted by the PRI transform.

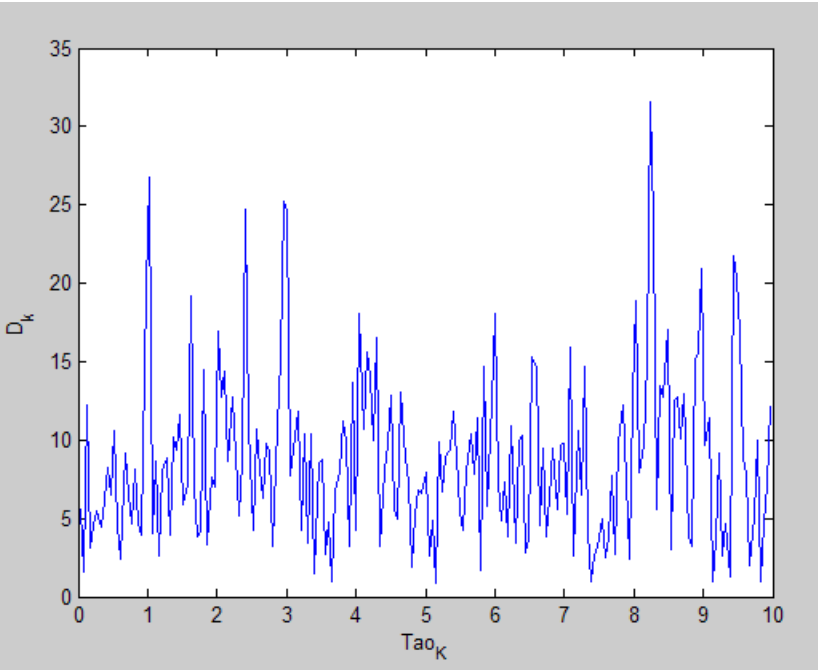

Figure 37. PRI transform of 1000 pulses from Scenario1.3-10% jitter

On the other hand, the improved PRI transform have better results. If the same pulse train with 10% jitter is given to the improved PRI Transform, the result in Figure 38 is obtained. In Figure 38, the continuous line shows the PRI spectrum and the dotted line shows the threshold function used to derive the PRI values. Here, it can be easily seen that the PRI values 1 and √2 can be detected, however, the √5 cannot be detected. In this case, if we increase the number of pulses given to the improved PRI Transform algorithm, then the PRI value √5 can also be detected. Please see the Figure 39, where the input pulse train has 3000 pulses and the PRI value √5 have also been detected.

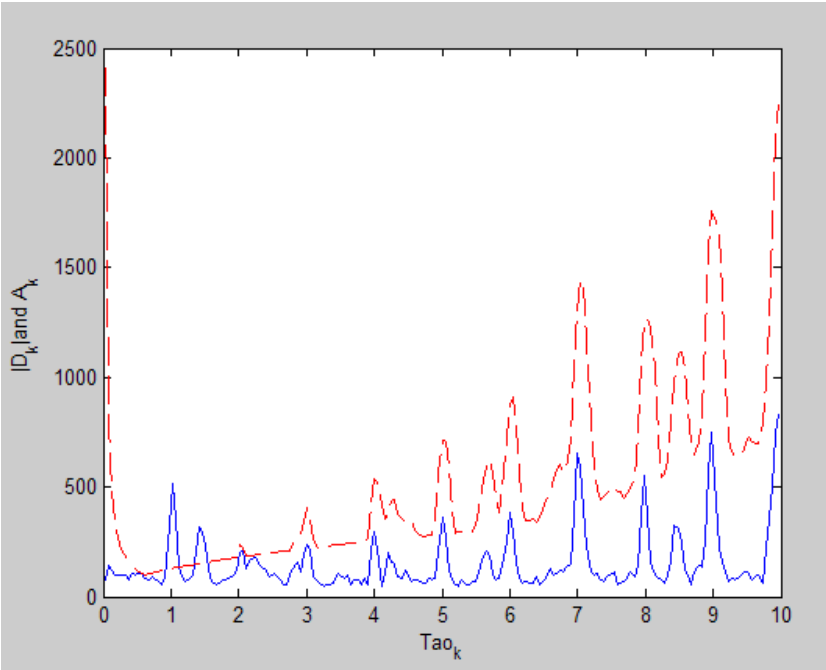

Figure 38. Improved PRI transform of 1000 pulses from Scenario1.3-10% jitter

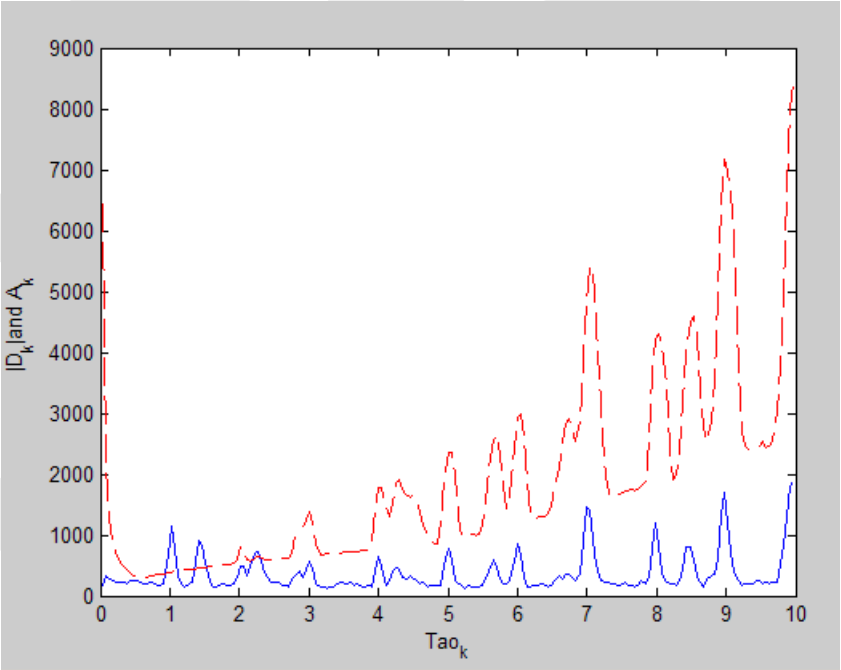

Figure 39. Improved PRI transform of 3000 pulses from Scenario1.3-10% jitter

In Figure 39, the PRI value √5 can also be detected from 3000 pulses. The magnified version of the Figure 39 is shown in Figure 40.

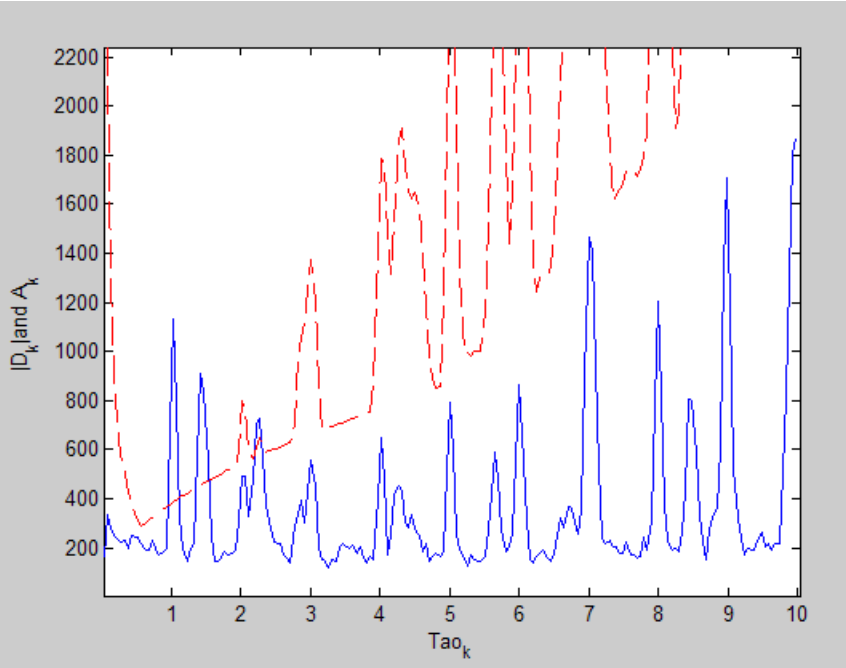

Figure 40. Zoomed version of the Improved PRI transform of 3000 pulses from Scenario1.3-10% jitter

Threshold function is calculated as described in section 3.3.1. The maximum of the 3 criteria, which is the A_k function, is shown in Figure 40, with dotted line. Also in Figure41, these 3 functions $\alpha(\tau_{max}-\tau_{min})/\tau_K$, βC_k and $\gamma\sqrt{(\tau_{max}-\tau_{min})(N/(\tau_{max}-\tau_{min}))^2 b_k}$ are shown. Here, the α, β and γ constants are chosen as 16, 0.3 and 3 respectively, such that, the peaks of the PRI values are above the threshold. Each of the function that the A_k consists of, can be seen in Figure 41.

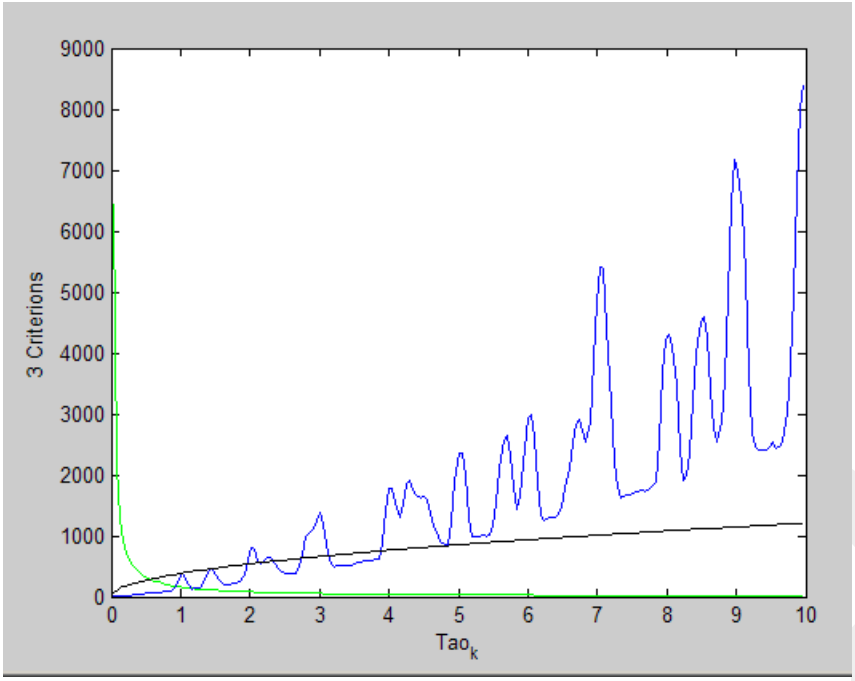

Figure 41. Three of the threshold functions

3.5.1.4 Identification

The Identification is done as stated in Section 3.4. Here, as an example the tables that are compared and the table that is the result of the comparison is shown. There are two tables for comparison as Tables 8 and 9. Table 8 lists the emitters that are available in a database created previously, and Table 9 has the threats that are detected by using the clustering and PRI transform algorithms.

RFMin	RFMax	RFMean	AoAMin	AoAMax	AoAMean	PWMin	PWMax	PWMean	PRI
8.57e9	9.47e9	9.02e9	250.59	255.58	253.07	3.19e-5	3.53e-5	3.37e-5	2.2139

Table 8. Emitters in a database

RFMin	RFMax	RFMean	AoAMin	AoAMax	AoAMean	PWMin	PWMax	PWMean	PRI
8.57e9	9.47e9	9.02e9	250.59	255.58	253.07	3.19e-5	3.53e-5	3.37e-5	2.2139
1.47e10	1.62e10	1.55e10	101.28	106.27	103.75	2.30e-5	2.54e-5	2.42e-5	1.0199
4.45e9	4.92e9	4.69e9	49.07	54.07	51.52	1.26e-5	1.39e-5	1.32e-5	1.4179

Table 9. Detected threats from the field

RFMin	RFMax	RFMean	AoAMin	AoAMax	AoAMean	PWMin	PWMax	PWMean	PRI
1.47e10	1.62e10	1.55e10	101.28	106.27	103.75	2.30e-5	2.54e-5	2.42e-5	1.0199
4.45e9	4.92e9	4.69e9	49.07	54.07	51.52	1.26e-5	1.39e-5	1.32e-5	1.4179

Table 10. Result of the comparison (Identified Threats)

In Table 10, the result of the comparison done as discussed in Section 3.4 is shown. Table 10 corresponds to the threats that are not available in the database.

3.5.2 Scenario 2

In Scenario 2, different from the Scenario 1, 80 threats are generated with 10 different PRI values. The parameters are shown in Tables 11 and 12.

Threat Generation Algorithm Parameters-Scenario 2.1	**Values**
PRI(p)	[1 √2 √3 √5 √7 √11 √13 √17 √19 √23]
Number of pulses(N_p)	1000 pulse for each emitter
Jitter for TOA(j_{TOA})	0
Number of Threats(N_t)	80
The sigma value(δ)	Randomly generated for each generation
RF upper- RF lower(RF_u-RF_l)	1e9Hz -18e9 Hz
AOA upper- AOA lower (AOA_u-AOA_l)	0°-359°
PW upper- PW lower(PW_u-PW_l)	0.1e-6s - 50e-6s
Jitter for RF(j_{RF})	0
Jitter for AOA(j_{AOA})	0
Jitter for PW(j_{PW})	0
Sampling Frequency(Fs)	[1024 64 32 128 16 2048 128 64 32 256]
Min max values for PA (S_{min}, S_{max})	[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1] and [1 1 1 1 1 1 1 1 1 1]
Frequency of sinc (f_{sinc})	[0.15 0.5 3 0.27 0.43 0.3 1 0.37 0.7 1]
Scan period(λ)	[0 0 0 0 0 0 0 0 0 0]
Sinc Amplitudes(A_{sinc})	[0.3 0.5 0.9 1 1.7 1.9 2.2 2.5 2.9 3]

Table 11. The input parameters for Scenario2.1

Threat Generation Algorithm Parameters-Scenario 2.2	**Values**
Jitter for TOA(j_{TOA})	10%
Jitter for RF(j_{RF})	10%
Jitter for AOA(j_{AOA})	5
Jitter for PW(j_{PW})	10%

Table 12. The input parameters for Scenario2.2 (that are different from the Scenario2.1)

3.5.2.1 Generation of Threats

In Figure 42 the generated pulse train according to the inputs in Table 11 is shown. 1000 pulses for each emitter corresponds to 80000 pulses in pulse train.

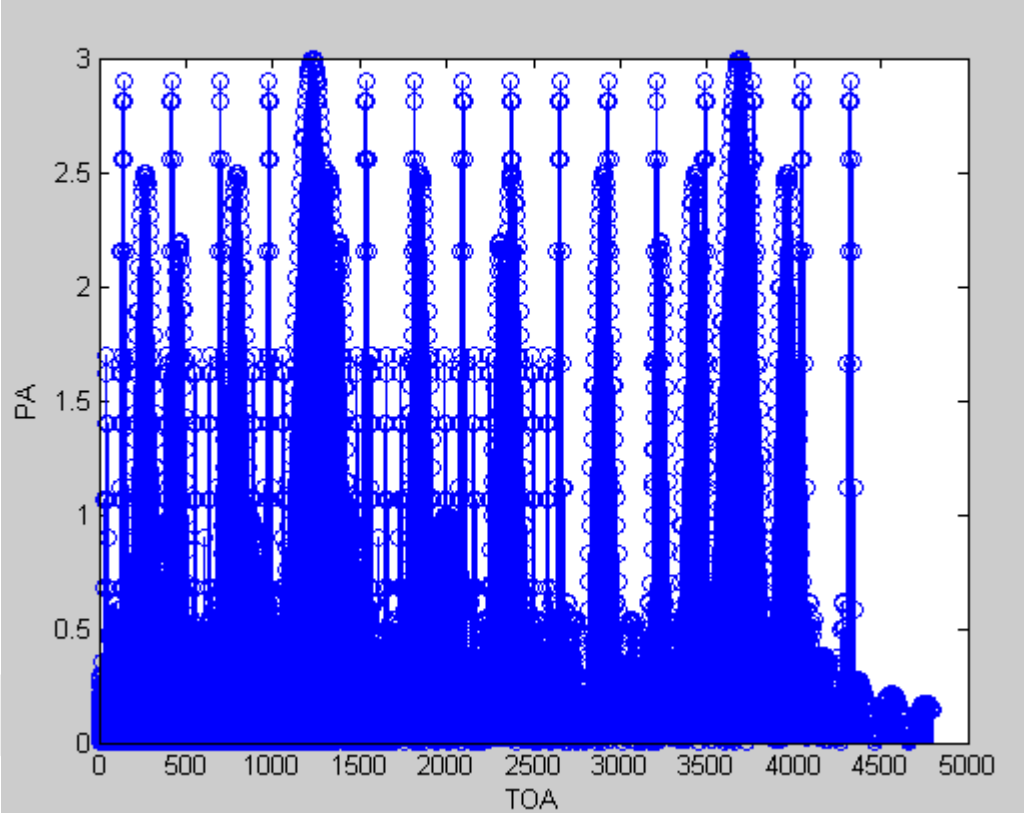

Figure 42. Pulse Train of Scenario 2. 80000 pulses-80 Emitters

The scattering of the pulse train according to RF, AOA and PW values which have no jitter are shown in Figure 43. Here 80000 pulses look like 80 pulses because there is no jitter added.

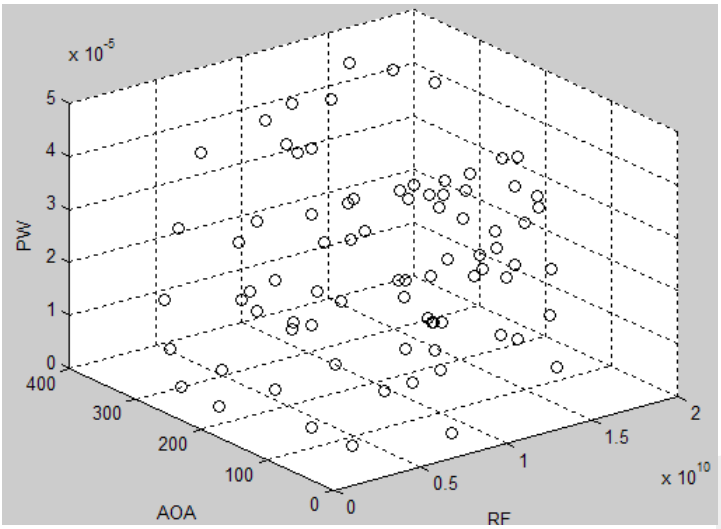

Figure 43. Pulses Train without jitter

When there is jitter according to the Table 12, the 80000 pulses look like as shown in Figure 44 where overlapping of clusters are visible. This may exemplify a dense emitter environment with emitters of similar parameters.

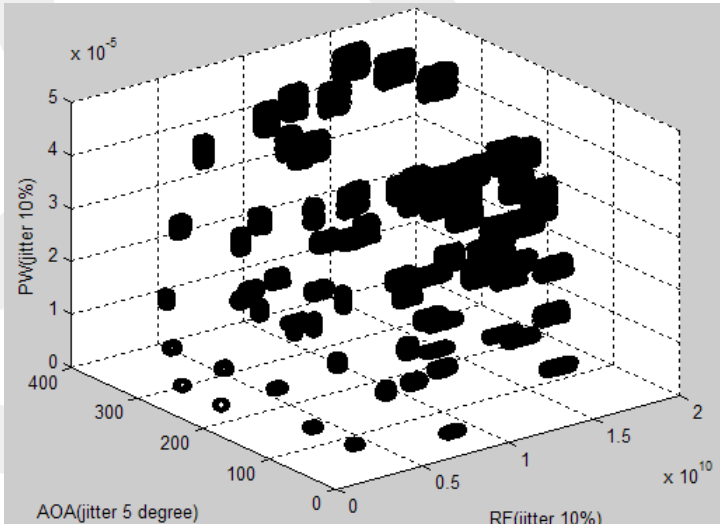

Figure 44. 10% jitter for RF and PW, 5 degree jitter for AOA

3.5.2.2 Clustering

In Figure 45, the pulse train with no jitter is clustered. There are 53 clusters found. According to the Cluster Boundary parameters, there may be more than one threat in one cluster. These parameters are generated randomly, that is why, the similar threats may be detected.

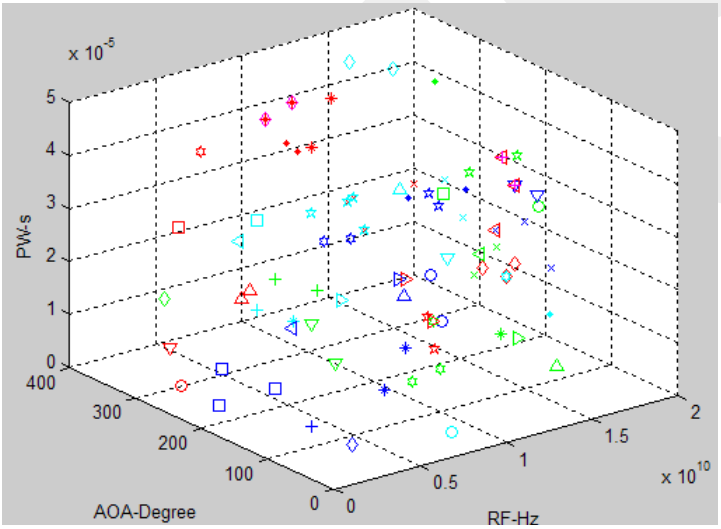

Figure 45. Each cluster is shown with a different shape (No jitter in RF,AOA and PW values-53 Clusters).

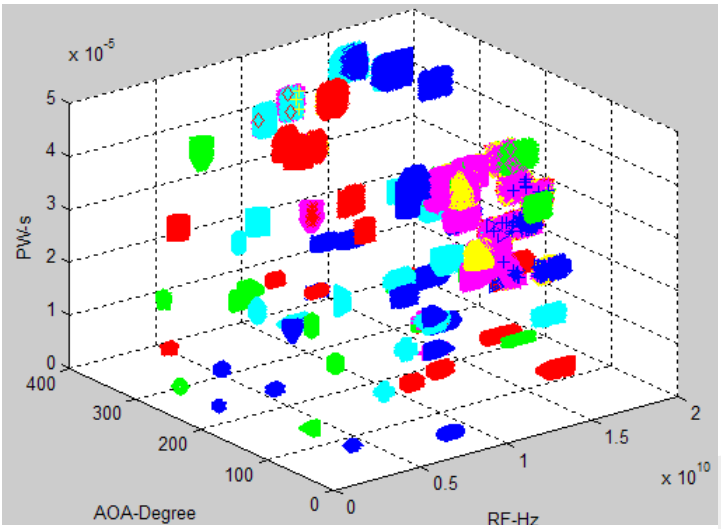

Figure 46. Clustered Pulse Train. (10% jitter for RF and PW, 5 degree jitter for AOA-74 clusters)

In Figure 46, the result of the clustering according to inputs from Table 12 is shown. There are 74 clusters found, and here, again there are clusters that are overlapped.

3.5.2.3 PRI Estimation

After clustering process, each cluster is sent to PRI Transform for estimating the PRIs. In Figures 47 and 48, the results of the first clusters for both Scenario2.1 and Scenario2.2 are shown. In Figures 49 and 50, the results of the second clusters are shown. Here, Figure 47 means that there are 2 threats with similar RF, AOA and PW values and Figures 41, 42 and 43 means there is one threat at all in each cluster.

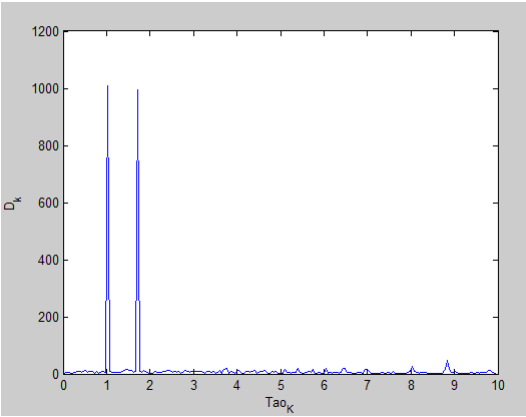

Figure 47. PRI estimation of 1st cluster. (The PRIs are 1,0199 and 1,7164.

PRI Transform, input has no jitter)

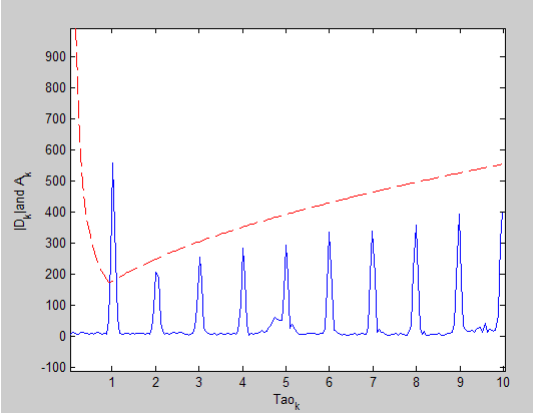

Figure 48. PRI estimation of 1st cluster. (The PRI is 1.0199. Improved PRI Transform, input has 10% jitter)

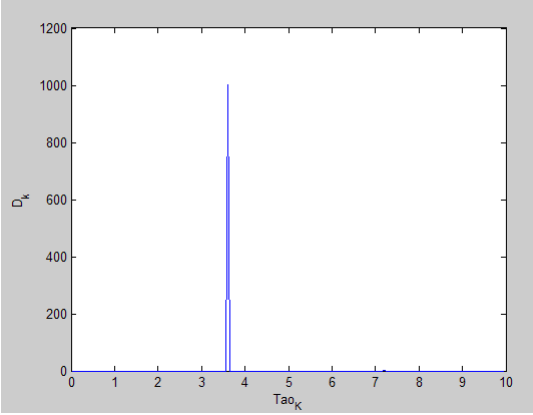

Figure 49. PRI estimation of 2nd cluster. (The PRI is 3.6070)

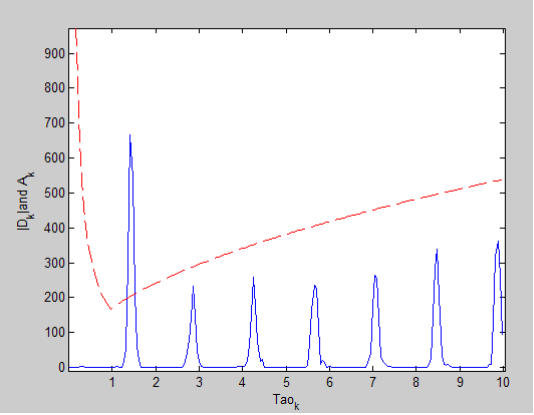

Figure 50. PRI estimation of 2nd cluster. (The PRI is 1.4179)

When there is no jitter, 78 threats are identified. This is due to the random generation of PDWs. If two of the threats are in same cluster and have similar PRI values, then these threats are detected as same threat. That is why, the number of identified threats is 78 not 80 in this case. When there is jitter, there are clusters that no PRI values are detected for that cluster, and also clusters that has false PRI values. This is due to the scaling of the threshold values. The threshold values are dependent on the input, however, the scales α, β and γ may also be different for each incoming pulse train. Finally, if we

remove the unidentified threats, among 80 threats, only the 42 threats are identified correctly for this run. Therefore, it is very important how α, β and γ are calculated, and it may require many runs.

3.6 Hardware Implementation

3.6.1 Design Flow

The Hardware implementation is designed by using the Synopsys's Synphony Model Compiler (SMC) tool. SMC is a toolbox provided by Synopsys for Simulink. This tool generates RTL code ready for synthesis. The tool guarantees that, the simulation result in Simulink environment to be same as in hardware. The tool captures test vectors for input and output, then, the generated RTL is verified with this test vectors. The Modelsim tool is used for this verification purpose. The generated code by SMC is compiled with Modelsim and then the same input test vector is given to the compiled design in Modelsim. Finally, the output test vector captured in Simulink environment and the output of the compiled design in Modelsim are compared. If there is no mismatch between these two outputs, it means that verification is successful. After verification, the RTL generated by SMC is synthesized by Synopsys's synthesis tool Synplify Pro. Then, the netlist generated by the SynplifyPro is given to the Xilinx's Place & Route tool. Finally, the generated bit file is loaded on FPGA.

3.6.2 Hardware design

The Clustering and the PRI Transform algorithms are designed for hardware. The Clustering and PRI Transform are not as completely functional as in the m-language design. In Clustering algorithm, if the incoming pulse belongs to more than one cluster, it is added to the one that has minimum distance to that pulse. However, in hardware design, the pulse is added to the first cluster that the pulse belongs to. Also, the number of pulses that the Clustering and PRI Transform can process are limited.

3.6.2.1 Clustering

The main subsystems in clustering design are the Counters, Input Buffer, Cluster Data Table, Check Previous Clusters, Selector, Create First Cluster, Create New Cluster, Add Previous Cluster and Processed Pulse Table. The relationship of these subsystems are shown in Figure 51. The output of the counters are used for reading and writing the data on Random Access Memories. The Input Buffer, Cluster Data Table and Processed Pulse Table subsystems consist of Random Access Memories. The inputs to the Clustering design are the RF, AOA, PW and the TOA values. The RF, AOA and PW values are first written to Input Buffer and then with TOA value and appropriate cluster number they are all written to Processed Pulse Table. Also, the data is processed by Selector subsystem to decide whether a first cluster or a new cluster is created or adding the current pulse to a previous cluster. These processes are completed by the Create First Cluster, Create New Cluster and Add Previous Cluster subsystems. The result of the Check Previous Clusters subsystem is used by the Selector subsystem and according to

the result of the Selector subsystem the results of Create First Cluster, Create New Cluster and Add Previous Cluster subsystems are written to the Cluster Data Table. Finally, the cluster number from Cluster Data Table is written to the Processed Pulse Table. The output of the Processed Pulse Table is the TOA, RF, AOA, PW and the cluster number of the pulse.

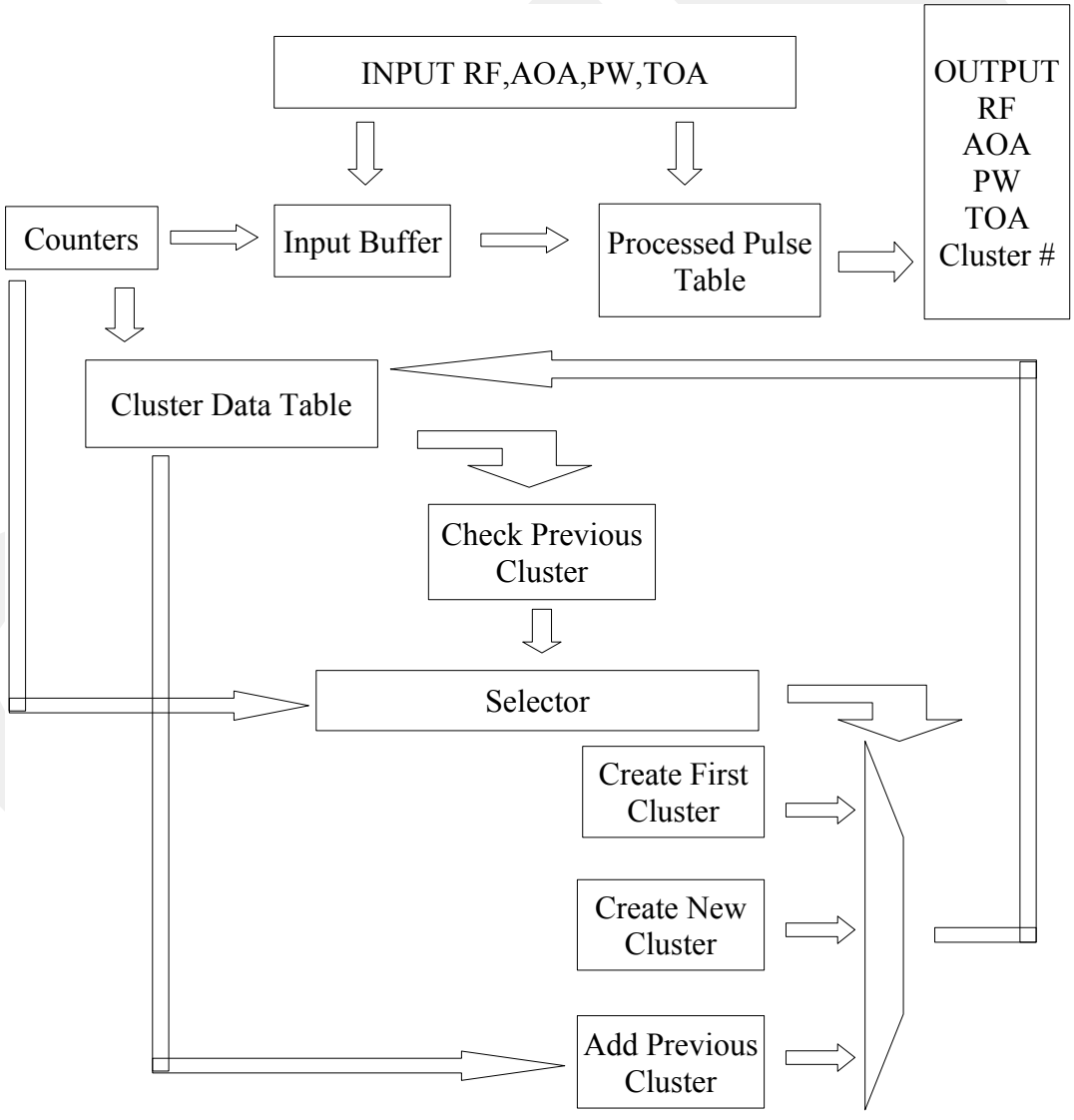

Figure 51. The relationship between subsystems of Clustering design

3.6.2.2 PRI Transform

The input to the PRI transform is the TOA values and the subsystems in PRI Transform are Counters, Input Buffer, Calculation of PRI Bin Boundary, Inside Boundary, Calculation of D_k and Update D_k. In Counters subsystem, there are 3 main counters. These are n, m and the k counters. Here, n counts from 1 to the number of pulses to be processed. The m counts from 1 to n and k counts from 1 to the number of bins K. n and m reads the TOA values written in Input Buffer. If the subtraction of the m^{th} value from the n^{th} value inside bin boundaries, then the PRI Transform D_k is updated. The relationship between these subsystems are shown in Figure 52.

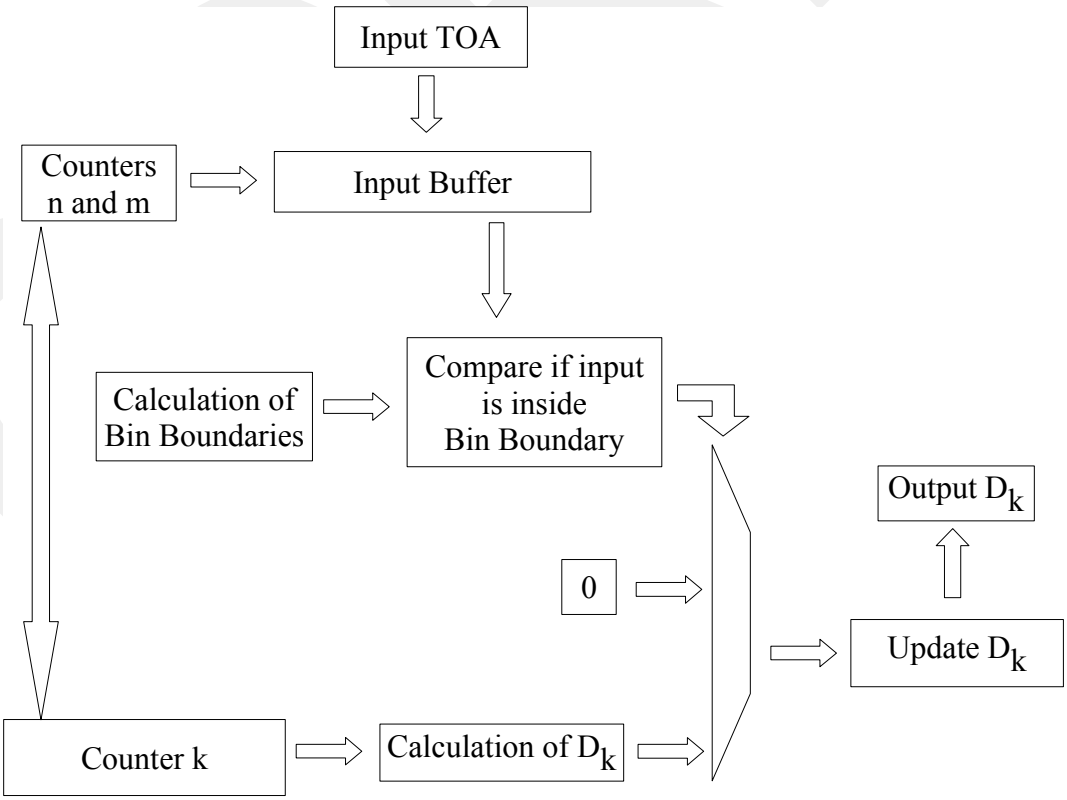

Figure 52. The relationship between subsystems of PRI Transform

3.6.3 Results

The clustering algorithm is designed such that it can only process 20 pulses at once. This is resulted from the sizes of the Random Access Memories chosen as design parameters. They are chosen so to make the design fit into the Digilent's Spartan 3E Starter kit board with 500K Gates. The inputs RF, AOA and PW given to the clustering algorithm have input bit widths of 83. 34 bits of this 83 bit is the fraction length. Each of these 83 bit inputs for RF, AOA and PW are buffered in memories. This corresponds to 83x3 bits. One needs 20x83x3 bits for buffering 20 pulses at once. Also, there are the Cluster Data Table and the Processed Pulse Table. These subsystems also uses memories. Again, one needs 20x83x3 bits for storing the RF, AOA and PW values in these subsystems. In addition to these pulses, the cluster number is stored as 11 bits. It corresponds to 11x20 bits for the Cluster Data Table and the Processed Pulse Table. One more data is the TOA value that is stored in Processed Pulse Table and Input Buffer as 83 bit. How many memory bits required for each variable is shown in Table 13.

Clustering Design	Input Buffer	Cluster Data Table	Processed Pulse Table
RF	83 bit	83 bit	83 bit
AOA	83 bit	83 bit	83 bit
PW	83 bit	83 bit	83 bit
TOA	83 bit	-	83 bit
Cluster #	-	11 bit	11 bit

Table 13. The required memory bits for each variable in each subsystem for clustering.

Totally, 83x11 + 11x2=935 bits required to process one pulse. Spartan 3E 500K starter kit has 360000 bit Block RAMs and 73000 bit Distributed RAM. 935 x 20 = 18700 bits required for processing 20 pulses. Also, when there are other peripherals, the design easily fits the board. However, if one desires to process 1000 pulses at once, it requires 935 x 1000 = 935000 bits which exceeds the resources. That is why, the results are shown for 20 pulses. This will not be functional however, it can be seen that the same results obtained for 20 pulses as in the m-language results.

PRI Transform Design	Input Buffer	D_k
TOA	83 bit	83x201x2 bits

Table 14. The required memory bits for each variable in each subsystem for PRI Transform.

In Table 14 the required memory bits for PRI Transform is shown. There are 201 bins for D_k and two RAM resource is used for each Real and Complex parts of the D_k. That is why the total bits for one pulse is 83+83x201x2=33449. There are also resources for calculating the phase that uses sine and cosine. That is why, the results are shown for 10 pulses for PRI Transform.

3.6.3.1Clustering

20 pulses are given as input and clustered. Then, the results of the m-language, the SMC and the Modelsim simulations are compared.

	1	2	3	4	5
1	0.1210	2.7094e+09	94.7428	2.4432e-05	1
2	0.2221	5.1637e+09	291.3339	1.8292e-05	2
3	0.2766	1.1588e+10	231.8780	2.2755e-06	3
4	0.5756	1.0645e+10	101.3311	3.4050e-05	4
5	0.7631	1.2059e+10	243.8207	1.7980e-05	5
6	0.7811	1.5404e+10	172.6873	1.0567e-05	6
7	0.8256	5.7157e+09	9.4906	4.6781e-05	7
8	0.8443	5.5273e+09	142.8150	1.2848e-05	8
9	0.8463	8.7120e+09	304.6869	9.8158e-06	9
10	0.9803	7.2885e+09	334.0967	2.0838e-05	10
11	1.1235	2.7094e+09	94.7428	2.4432e-05	1
12	1.6810	1.1588e+10	231.8780	2.2755e-06	3
13	1.8319	5.7157e+09	9.4906	4.6781e-05	7
14	1.9937	1.0645e+10	101.3311	3.4050e-05	4
15	2.1116	2.7094e+09	94.7428	2.4432e-05	1
16	2.5053	1.2059e+10	243.8207	1.7980e-05	5
17	2.5762	8.7120e+09	304.6869	9.8158e-06	9
18	2.8337	5.7157e+09	9.4906	4.6781e-05	7
19	2.8887	5.1637e+09	291.3339	1.8292e-05	2
20	3.0357	1.5404e+10	172.6873	1.0567e-05	6

Figure 53. The Clustering result of the m-language

In Figure 53, the result of the m-language run is shown. Here, the columns are TOA, RF, AOA, PW values and the Cluster numbers respectively. It is can be seen that, the cluster numbers assigned by the clustering algorithm from first to the last are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 3, 7, 4, 1, 5, 9, 7, 2 and 6 respectively.

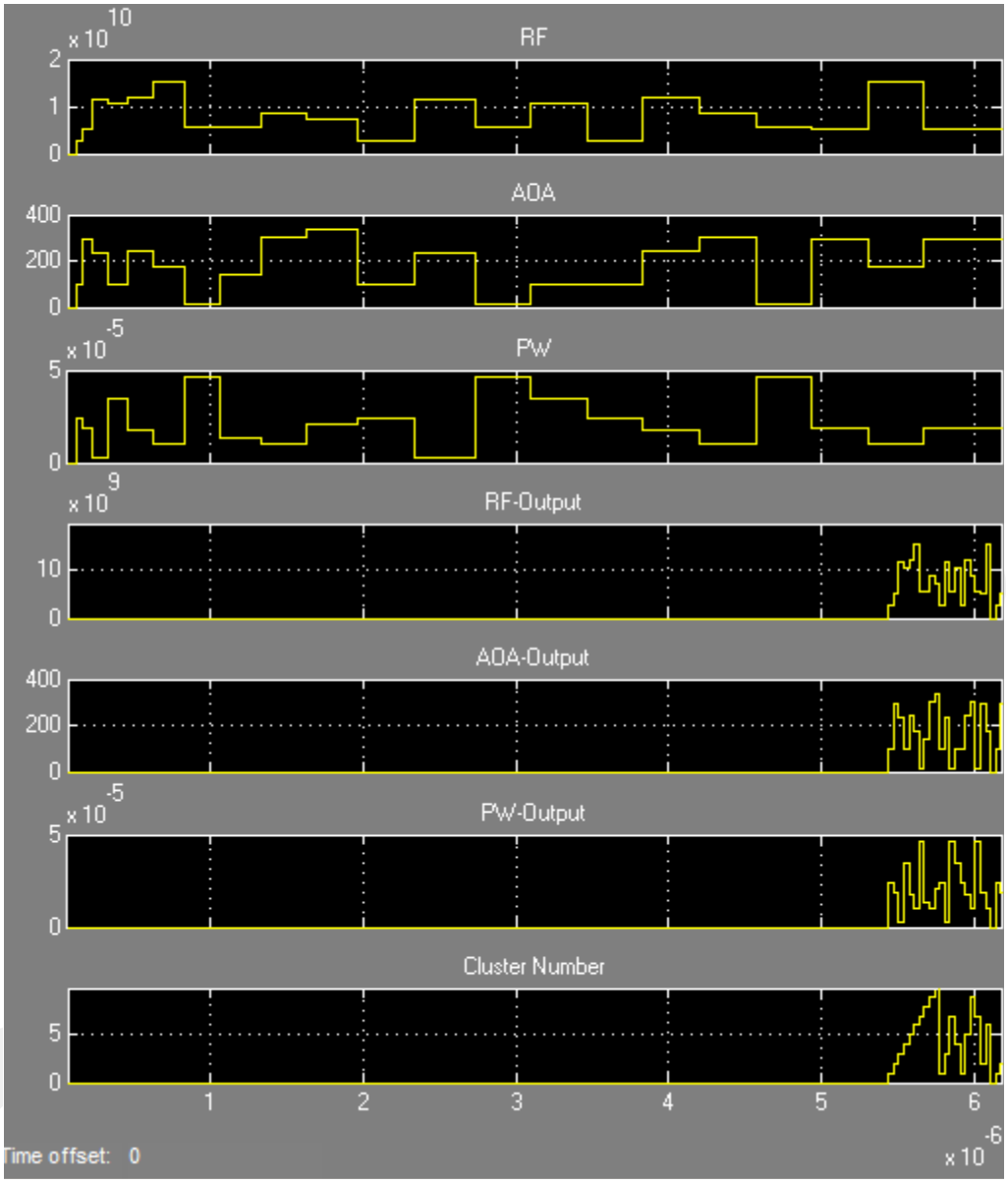

Figure 54. The Clustering result of the SMC in Simulink

In Figure 54, the inputs to the clustering design RF, AOA and PW are shown in first 3 rows respectively. Also, the outputs RF, AOA, PW and Cluster number values are shown for the last 4 rows. The scaled version of the last 4 row is also shown in Figure 55.

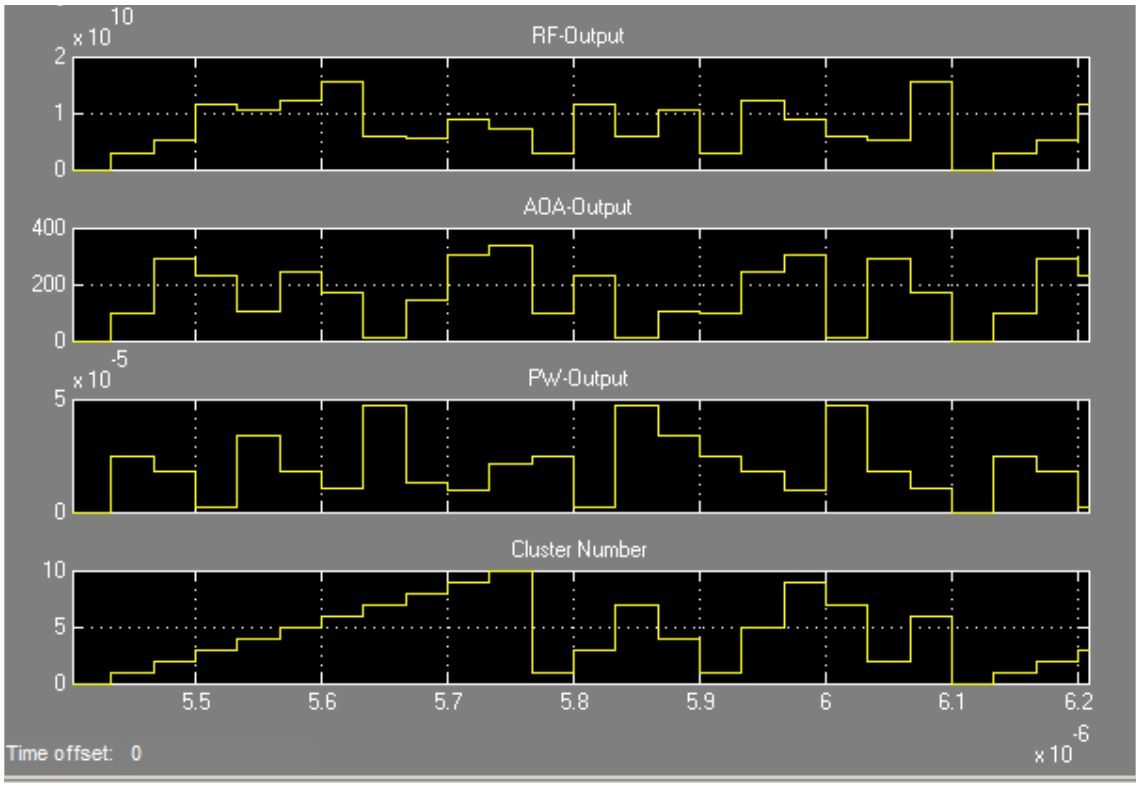

Figure 55. The scaled version of Figure 54 for the last 4 rows.

If we compare the last row which shows the Cluster Number in Figure 55, we see that the order; 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 1, 3, 7, 4, 1, 5, 9, 7, 2 and 6 respectively.

Also, in Figures 56, 57, 58, 59 and 60, the result of the Modelsim simulation is shown. If we follow the cluster numbers, again we see that they are same with the algorithm developed with m-language.

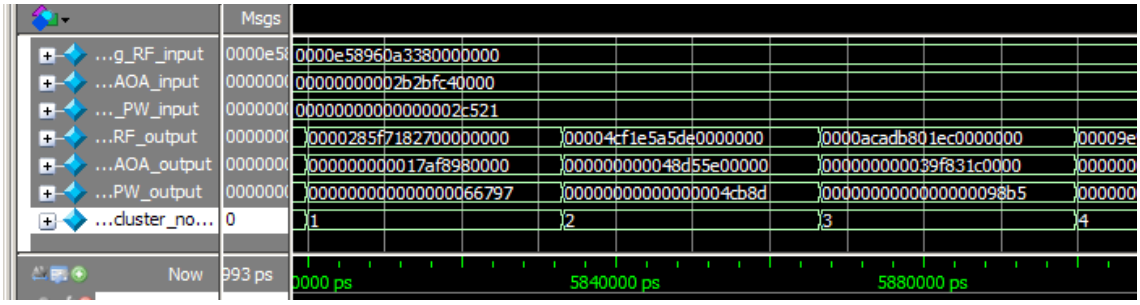

Figure 56. The result of Modelsim Simulation-Part 1-(Cluster No's:1-2-3)

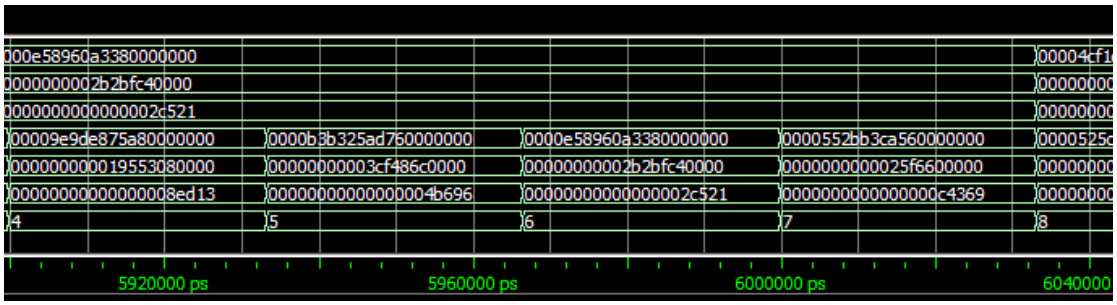

Figure 57. The result of Modelsim Simulation-Part 2(Cluster No's:4-5-6-7)

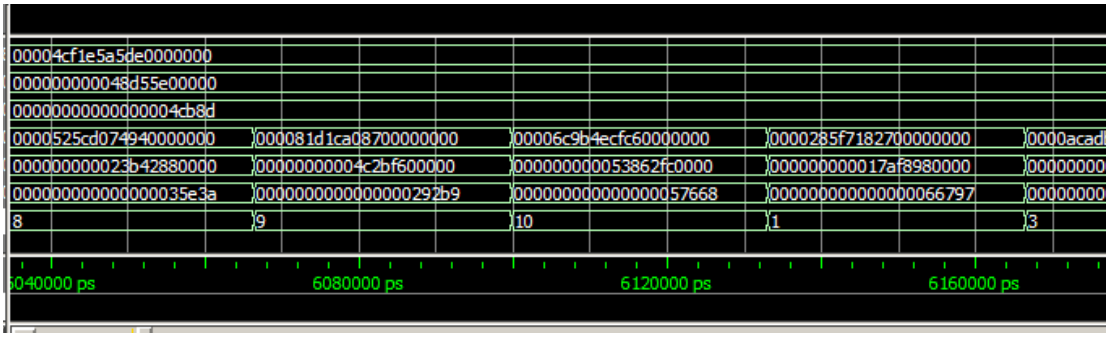

Figure 58. The result of Modelsim Simulation-Part 3(Cluster No's:8-9-10-1)

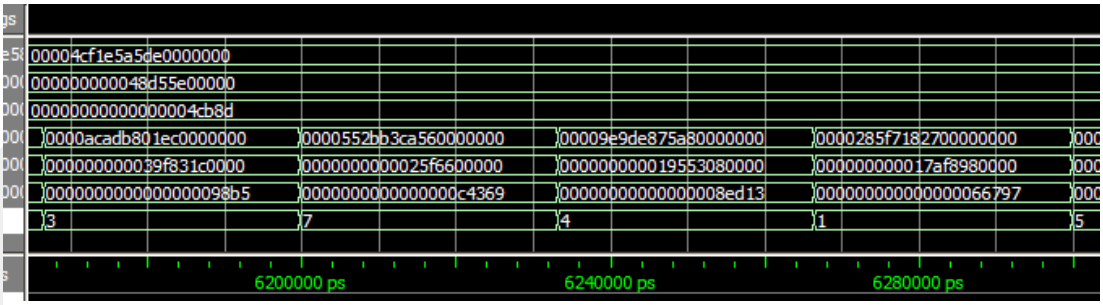

Figure 59. The result of Modelsim Simulation-Part 4(Cluster No's:3-7-4-1)

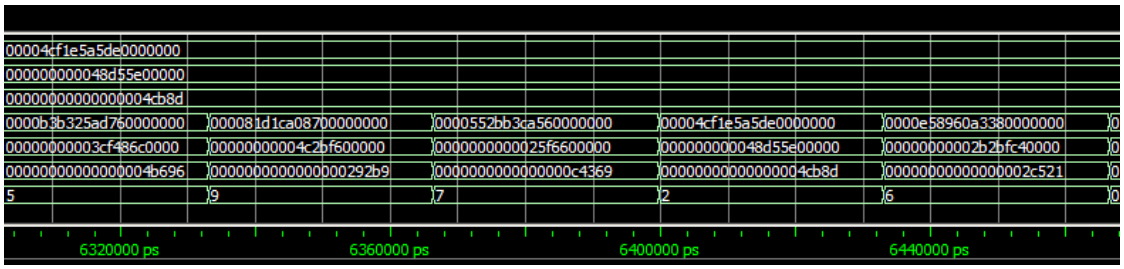

Figure 60. The result of Modelsim Simulation-Part 5(Cluster No's:5-9-7-2-6)

3.6.3.2 PRI Transform

Same 10 TOA inputs are given to both the m-language and the SMC design. The results shown in Figures 54 and 55 are not exactly the same, however, they are enough to estimate the PRI.

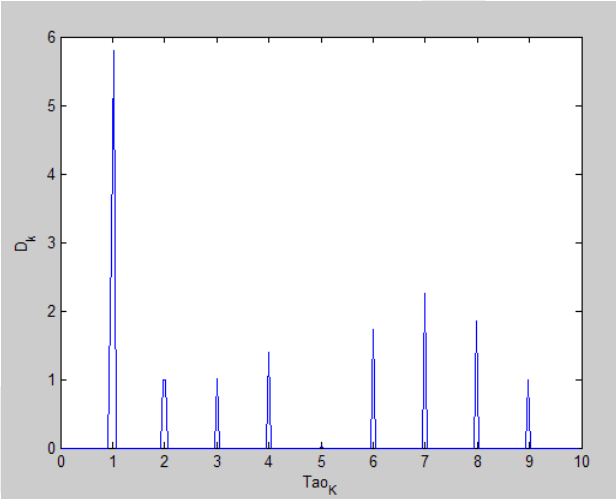

Figure 61. m-language result

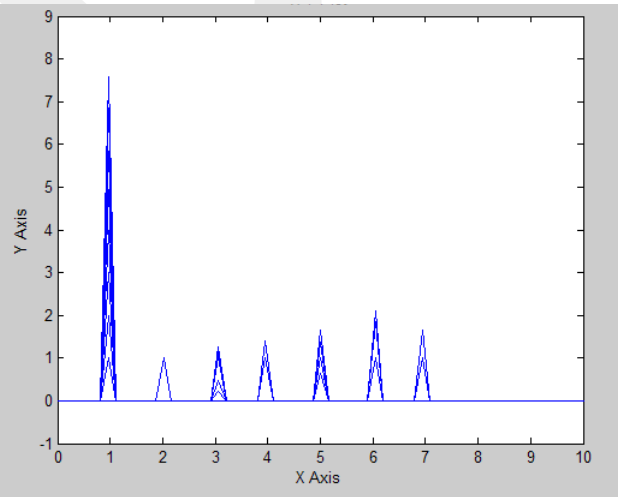

Figure 62. Simulink result

CHAPTER 4

CONCLUSIONS

In conclusion, the place of Signal Processing and Identification in EW is introduced briefly at the beginning of the study. It is aimed to realize a Signal Processing and Identification of an Intercept Receiver. The following methodology that consists of Clustering, PRI Estimation and Identification is accepted to do so. Then, partially the algorithms are developed and implemented in m-language. First, the algorithms are simulated partially and then simulated as a whole system. Also Threat Generation Algorithm is developed for simulating the system. As a final step, the whole system is tried to be implemented on FPGA. Again, the Clustering, PRI Estimation and Identification parts are designed partially. However, while trying to implement the PRI Estimation part separately, it is seen that the FPGA Board is insufficient. Optimizations like using area efficient architectures are tried. For example, instead of using Look-up Table based sine and cosine architectures, the CORDIC sine and cosine architectures are used. Since the bottleneck is the area, the trade off between the speed of the Look-up Table based sine-cosine architectures and the area efficient CORDIC is easily done by preferring the CORDIC architecture. Also, the bit widths of the signals are reduced to

gain from area. In this case, reducing the bit width does not have an effect on small TOA values. However, when TOA increases, the saturations on the results are expected. Finally, by decreasing the number of pulses that the design can process, the design is able to fit to the device. However, this time the final design was not functional because it can only process 10 pulses at once. It is just proved that the design can be implemented on FPGA and works as expected as in the m-code for the same given input. As a future work, the design can be implemented in a device that has higher capacity or by using the external memories on the FPGA board.

The difference between the PRI Transform and the improved one is impressive when jitter is added. Here, the threshold function, for extracting PRI values, is dependent on the incoming pulse train. However, there are also the α, β and γ values that scales threshold. It is also seen that these values needed to be changed according to the incoming pulse train if the algorithm is working in a whole system as discussed in Scenario2.

In Clustering algorithm, the Δ values for cluster boundaries are also important as discussed in Section 3.2. One other issue might be optimal determination of these Δ values.

REFERENCES

[1] Skolnik, Merril I., "Radar Handbook", Second Edition,1990.

[2] Wiley Richard G., "ELINT: The Interception and Analysis of Radar Signals", 2006.

[3] Güven Erhan, "Emitter Identification in Electronic Warfare by the Use of Clustering Techniques", M. S. Thesis in Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, Sept., 1994.

[4] Güvenlik Ali Rıza, "Clustering Techniques for Emitter Identification in Electronic Warfare", M. S. Thesis in Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, Sept., 1999.

[5] Aslan Mehmet Kadir, "Emitter Identification Techniques in Electronic Warfare", M. S. Thesis in Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, Dec.,2006.

[6] Olgun Muhammet Ertuğ, "Design and FPGA Implementation of an Efficient Deinterleaving Algorithm", M. S. Thesis in Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey, Dec.,2008.

[7] Kobayashi Masaaki, Nishiguchi Ken'ichi, "Improved Algorithm for Estimating Pulse Repetition Intervals", IEEE Transactions on Aerospace and Electronic Systems, Vol. 36, No. 2, pp. 407 – 421, April, 2000.

[8] Chan Y. T., Chan F., Hassan H. E., "Performance Evaluation of ESM Deinterleaver Using TOA Analysis", 14th International Conference on Microwaves, Radar and Wireless Communications, Vol. 2, pp. 341 – 350, 2002.

[9] Mardia H. K., "Adaptive Multi-dimensional Clustering for ESM", IEE Colloquium on Signal Processing for ESM Systems, pp. 5/1-5/4, April, 1988.