APPLICATION OF KALMAN FILTERING TECHNIQUE FOR SAR PROCESSING OF SPARSE SATELLITE CLUSTERS

by

SUBHASH GULLAPALLI

B. E. (With Distinction), Electronics and Instrumentation Engineering
GITAM Engineering College, Andhra University, May 2000
Visakhapatnam, India

Submitted to the Department of Electrical Engineering and Computer Science and the Faculty of the Graduate School of the University of Kansas in partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering.

Thesis Committee:

____
Dr. James Stiles: Chairperson

____
Dr. John Gauch

____
Dr. Christopher Allen

Date Submitted: ________________________

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. James Stiles for giving me the opportunity to work on this project and for guiding me all throughout my studies at The University of Kansas. Thank You! Dr Stiles, for being my mentor and guiding me towards the engineer I am today. Especially I would like to thank him for being there for me when I was unable to work in the summer, and for going out of the way to help me during that period. I would also like to thank Dr. John Gauch for spending those long hours with me while working on the parallel processing part. I would like to thank Dr. Christopher Allen for accepting to be on my committee.

I would like to thank my friends Hari Palakurthi, Raju Kumar Girimaji and Shiva Prasad Ponnaboyina for being there for me when I really needed them. I should also thank Donnis Graham, Brett Becker and all the other staff of ITTC for all the help they provided. I would also like to thank Nathan Goodman for helping me during the initial stages of my research work.

Last but never the least I would like to thank god, my mother Sarada, my father Satyanarayana, my sister Sridevi, my brother-in-law Venkata Ramana and my nephews Anudeep and Abhishek for all the love and support they provide.

Table of Contents

ABSTRACT IX

1. INTRODUCTION AND MOTIVATION 1

2. PROCESSING OF MULTIPLE-APERTURE SPACE-BORNE ARRAYS FOR WIDE-AREA SAR 8

2.1 Signal Space Representation of the Radar System 8

2.2 Single-Aperture SAR 12

2.3 Multiple-Aperture SAR 14

2.4 Sparse Arrays 15

2.5 Correlation or Matched Filter 16

2.6 Maximum-Likelihood Filter 17

2.8 MMSE Filter 19

3. KALMAN FILTER IMPLEMENTATION 22

3.1 Kalman Filter Theory 22

3.2 Kalman Filter Implementation: 24

3.3 Performance of the developed Kalman Filter 28

3.3.1 Performance of KF for random scattering coefficients 29

3.3.2 Performance of KF for a real SAR Scenario 33

3.4 Comparison between Matched, Maximum Likelihood, MMSE and Kalman Filters 35

4. PERFORMANCE CHARACTERISTICS OF THE KF 46

4.1 Variation of Kalman Gain, Error Correlation Matrix and Innovation Energy with Processing 47

4.2 Dependence of the final estimate on the Initial Conditions 55

4.2.1 Significance of the Number of divisions, L chosen 55

4.2.2 Significance of the Initial value chosen for Error Correlation Matrix 57

4.2.3 Perfect Initial Conditions 62

5. OPTIMIZATION OF DEVELOPED KALMAN FILTER 65

5.1 Optimization of Matrix Operations 65

5.2 Number of Measurements required for processing to making an Optimal Estimate (Sequential Estimation) 70

5.3 Methods Pursued to reduce Error Correlation Matrix Computations 76

5.3.1 Only Diagonal Elements in the Error Correlation Matrix are updated 77

5.3.2 Diagonal Spread is considered 80

5.3.3 Error Correlation Matrix is updated based on the Expected Measurement Correlation Matrix 82

5.3.4 Correlation of only targets closer to each other is considered 84

6. PARALLEL IMPLEMENTATION OF KF 88

6.1 C code 88

6.2 Parallel Processing 89

6.2.1 Necessity and use of Parallel Processing 89

6.2.2 Building a Parallel Virtual Machine 90

6.2.3 Working Methodology 91

6.2.4 Provisions made for huge matrix sizes 94

6.2.5 Testing and Results 94

7. CONCLUSIONS AND FUTURE WORK 98

7.1 Conclusions 98

7.2 Future Work 99

8. REFERENCES 100

LIST OF FIGURES AND TABLE

Figure 2.1 Radar Geometry for a constellation of radar satellites with the flat earth approximation [8] 8

Table 3.1 Input parameters for the Radar Simulator that are kept constant 28

Table 3.2 Input parameters for the Radar Simulator for *M*=1024 29

Figure 3.1 Performance of the developed KF for randomly chosen scattering coefficients. 32

Figure 3.2 Image of the KU football Stadium used as the Input to the Radar Simulator ... 33

Figure 3.3 KF estimate obtained for each iteration 34

Figure 3.4 KF Result for KU Football stadium Image 35

Figure 3.5 A smaller Image of KU Football stadium 36

Figure 3.6 A pictorial representation of the Expected Correlation between the responses obtained from the illuminated targets 37

Figure 3.7 Comparison of Matched, ML, MMSE and Kalman filters performance versus SNR 38

Figure 3.8 Matched, ML, MMSE, and Kalman Filters performance versus Input SNR for 13 receiver sparse random array 40

Figure 3.9 Matched, ML, MMSE and Kalman Filters performance versus number of receive apertures for a sparse, random array and moderate SNR 41

Figure 3.10 Matched, ML, MMSE and Kalman Filters processing speed versus number of receive apertures for a sparse, random array 42

Figure 3.11 Processing Speed of KF versus L 44

Figure 4.1 Normalized MSE (dB) vs. the fraction of Measurements Processed (e) for the smaller version of the KU Football Stadium Image 47

Figure 4.2 A pictorial representation of the Error Correlation Matrix after processing of half the measurement data 48

Figure 4.3 Shows the Variation of the ratio between the RMS values of the diagonal and non-diagonal elements of the error correlation matrix 50

Figure 4.4 The ratio between the absolute mean of the diagonal elements of $\mathbf{b}$ to that of the variance of noise vs. the fraction of measurements processed e 52

Figure 4.5 Innovation energy vs. the fraction of measurement data processed................. 53

Figure 4.6 Normalized MSE versus the fraction of measurement data processed for different L 56

Figure 4.7 Variation of Normalized MSE versus the fraction of measurement data processed for processing, for different s^2 values for diagonal matrix $\mathbf{K}_g(0)$ 58

Figure 4.8 Variation of Normalized MSE versus the fraction of measurement data processed for processing for different $\mathbf{K}_g(0)$ 60

Figure 4.9 Variation of Normalized MSE versus the fraction of measurement data processed for a case where the final values obtained for $\mathbf{g}$ and $\mathbf{K}_g$ are taken as the initial values and the processing is done again. 63

Figure 5.1 Required number of Significant Operations versus L 66

Figure 5.2 Processing time taken versus L for KF Implementation in Matlab.................. 67

Figure 5.3 Total Processing time taken versus L for KF Implementation in Matlab 68

Figure 5.4 Comparison between performance of optimized and unoptimized KF versus number of receive apertures in the sparse satellite cluster..69

Figure 5.5 Normalized MSE (dB) vs. the fraction of measurement data processed for random scattering coefficients 72

Figure 5.6 Innovation Energy vs. the fraction of measurement data processed for random scattering coefficients 72

Figure 5.7 RMS value of elements of Kalman Gain Matrix vs. the fraction of measurement data processed for random scattering coefficients 73

Figure 5.8 Variation of the update in present estimate as compared to the previous estimate versus the fraction of measurement data processed. 74

Figure 5.9 Trace of the Target Correlation matrix versus the fraction of measurement data processed..75

Figure 5.10 Normalized MSE (dB) versus L for a case where only the diagonal elements are updated in the $\mathbf{K}_g$ matrix..78

Figure 5.11 x The normalized Variation between $\mathbf{G}_u$ and $\mathbf{G}$ versus fraction of measurement vector processed for L=2 and L=4 79

Figure 5.12 Processing time taken versus L for a case in which only the diagonal elements are considered in the error correlation matrix $\mathbf{K}_g$ 80

Figure 5.13 Elements of $\mathbf{K}_g$ considered for a diagonal spread consisting of 129 lines adjacent to the diagonal on either side...81

Figure 5.14 Normalized MSE versus the number of adjacent diagonal lines considered on one side 82

Figure 5.15 Normalized MSE versus the number of adjacent diagonal lines considered on one side 83

Figure 5.16 Normalized MSE versus the fraction of measurement data processed, for different thresholds 84

Figure 5.17 Shows the target pixels (dark shade) supposed to be highly correlated to the concerned target pixel (‘star’) located at different parts of the region concerned 85

Figure 5.18 Points considered in the target correlation matrix for corresponding correlation assumptions 86

Figure 5.19 Normalized MSE versus the fraction of measurement used for different correlation characteristics 87

Figure 6.1 Diagram depicting the Parallel processing structure implemented in KF 93

Figure 6.2 Processing time versus L for different number of parallel processors used 95

ABSTRACT

There is currently a strong push toward improving space-borne radar technology due to several advantages provided, the most important being the ability to achieve global coverage. Yet due to the inherent tradeoff between size, weight and power, relatively few space-borne radars have been implemented. One proposed approach to mitigate this space-size-power tradeoff was to implement a cluster of satellites called a *constellation*. But the disadvantage of a constellation of formation-flying satellites is that the array formed by the constellation will be sparsely populated, three-dimensional, and non-uniformly spaced, so specific methods have to be developed for processing of the data obtained from them. Three filters were proposed so far, the matched filter, maximum-likelihood filter (ML), and minimum mean-squared error (MMSE) filter for processing. But each of these filters pose their own problems. The Matched filter is unable to eliminate measurement error due to clutter, while the ML filter is unable to minimize error due to noise, and the MMSE filter though being able to give a good estimate has inherent computational complexities. So a new method is needed to process the obtained data as efficiently as the MMSE while being able to reduce the computational problems inherent in it. In this thesis, we implemented Kalman Filter (KF), an iterative approach to MMSE, to minimize the processing load. The developed KF is tested for different real-time SAR applications and for different target scenarios. KF is fully optimized and many tradeoff scenarios are proposed between the processing load and the accuracy of the estimate. Finally, parallel processing has been implemented for distributing the computational load.

1. INTRODUCTION AND MOTIVATION

There is currently a strong push toward improving space-borne radar technology. This is due to several advantages provided by space-borne radar, the most important of which is the ability to achieve global coverage. Moreover space-borne radars do not require refueling or a monitoring crew and are also not susceptible to any military threat or bad weather conditions. Yet relatively few space-borne radars have been implemented due to the many technical challenges associated with this task. The most significant being the tradeoff between size, weight and power. In order to compensate for the extreme distances from targets, a space-borne radar has to generate and transmit sufficient energy. The relatively long wavelengths of the microwave region result in comparable apertures, which are physically large. Not only does this increase sensor weight but also additionally makes deployment of these large structures problematic.

Other factors requiring large apertures are ambiguity and resolution. For Synthetic Aperture Radar (SAR), the minimum antenna size is governed by the minimum SAR antenna area constraint. Complying with this constraint ensures the range-Doppler ambiguities are not illuminated. Moreover, as antenna beam-width and size are inversely related, a large antenna can only illuminate a small region. So, for global coverage more number of visits are required to get more information about a particular region of interest. One approach to mitigate the space-size-power tradeoff is to implement a cluster of satellites called a *constellation*. Each satellite has a coherent receiver, and the constellation flies in formation to create an antenna array.

The individual satellites in this concept are small, with relatively small apertures; hence, they are termed *microsats*. The data from each microsat and the spatial sampling of the constellation are combined to form a single, virtual radar.

Several advantages are achieved by breaking the radar system into multiple, free-flying components. The effective aperture of the system is still determined by its total energy-collecting area, but that area is now divided between many satellites. The sum of the aperture sizes of all the satellites determines the total effective aperture of the system, but it is more cost effective to launch and deploy a constellation of formation-flying microsats than to launch a similar monolithic satellite. The second advantage is in the added spatial sampling. For SAR, it is possible that the spatial samples obtained from multiple receivers can be used to distinguish between range-Doppler ambiguities since ambiguous cells have different angles of arrival. The increased illumination area results in a higher search rate without sacrificing the angular resolution necessary for detecting slow-moving targets. Other advantages of satellite constellations include graceful performance degradation and ease of replacing or upgrading satellites.

The disadvantage of a constellation of formation-flying satellites is that the array formed by the constellation will be sparsely populated, three-dimensional, and non-uniformly spaced. It may be possible to temporarily propel the microsats into a regular, well-formed pattern, but the amount of propulsion necessary to maintain a regular formation puts this option beyond any reasonable fuel budget. Therefore, the radar designer has essentially no control of the array structure formed by the constellation except, possibly, for some input about the general, overall size of the constellation. The question now becomes what is the best method for processing this data obtained from sparsely populated multiple aperture spaceborne radar.

Research has been done to develop algorithms for processing of this sparsely populated multiple aperture spaceborne radar [2, 3, 4, 6, 8]. Three filters have been proposed so far, the matched filter, maximum-likelihood filter, and minimum mean-squared error filter. Even though the matched filter is able to maximize the received energy with respect to noise, it is unable to minimize the error due to presence of clutter in target response. So, if the responses from the illuminated targets are significantly correlated, a correspondingly large error will result. Generally, for the sparse array case, the responses of some resolution cells will be significantly correlated, and thus a different linear processor must be implemented. Some earlier work has been done to develop a simulator for multi-aperture spaceborne radar [7].

The maximum likelihood (ML) estimator when applied to the present scenario is able to minimize the error due to clutter, where clutter is defined as responses from all other resolution cells. But it does nothing to minimize the error due to noise. As a result the estimate error (i.e., the SAR image) quickly degrades as measurement SNR declines.

Given that some *a priori* knowledge of the radar SNR is available, a minimum mean-squared error estimator can be implemented. This estimator is the discrete implementation of a wiener filter and minimizes the estimate error due to both noise and clutter. In other words if the matched filter maximizes signal to noise, and the

ML estimator maximizes signal to clutter, the MMSE estimator can be said to maximize signal to interference, where interference is defined as the summation of both clutter and noise energy. Accordingly, this estimator provides SAR images superior to both correlation and ML processing for all SNR.

The only disadvantage of the MMSE processor is the huge additional complexity in determining the linear estimator. Additionally, for large problems, the matrix inverse operation required to implement the MMSE estimator is very problematic. Especially in the field of radar signal processing computing the inverse of the large matrices can really slow down the processing speed. An iterative implementation of the MMSE algorithm can be developed where the data vector is split into smaller segments to reduce processing time.

So a Kalman filter (KF) algorithm, an iterative implementation of the MMSE estimator is proposed, developed, analyzed and optimized. It has been shown that the processing speed can be decreased, by breaking the data vector into an optimal number of segments. It had also been proposed, that the overall estimation error could be decreased by implementing a KF. The KF has been thoroughly analyzed for different kinds of target scenarios, for different SNRs, for different number of receivers and for different initial values for the error correlation matrix.

Much work has been done to optimize the overall KF algorithm and to decrease the processing time even further. The order of matrix multiplications has been chosen in an optimal way so as to decrease the overall computations required, thereby reducing the processing time. A number of different ways have been explored to decrease the huge number of computations required for operations related to the error covariance matrix. Simulations are done to analyze the performance of the KF for different approximations made to the error correlation matrix so as to decrease the number of computations to a great extent. It has been shown that though it is possible to reduce the processing time by making approximations to the error correlation matrix, some loss is accrued in the accuracy of the final image. For some approximations made to the error correlation matrix it is possible to obtain a trade off between the accuracy of the final estimate and the processing time.

The KF algorithm consists of 5 major matrix multiplications and 1 matrix inverse for a case where the data vector is undivided or for the MMSE case, with the matrix inverse operation constituting the dominant part of the overall processing time. But as the data vector is divided into smaller and smaller segments, the matrix multiplication operations become the dominant part. So if a procedure is implemented so as to decrease this dominance of matrix multiplications in the processing, then it will be possible to decrease the overall processing time.

It is possible to speed up matrix multiplication operations by implementing distributed or parallel processing. But it is very difficult to implement existing algorithms for speeding up of matrix inverse operations. So algorithms are developed and implemented to perform matrix multiplications over parallel multiple processors. A Parallel Virtual Machine comprising of multiple fast to medium processors has been built to test the algorithms. Finally numerous simulations have been performed to analyze and quantify the overall parallel processing of KF implementation of sparsely populated multiple aperture spaceborne radar.

In the following chapters of this thesis draft, the methodology to implement parallel processing for KF for SAR processing of data obtained from sparsely populated, irregularly spaced, radar arrays is investigated. Chapter two begins with the basic radar model that is the foundation used to build effective signal processing algorithms. Brief information is given regarding the simulator developed earlier for multi-aperture spaceborne radar. The performance of matched, maximum-likelihood and minimum mean-squared error (MMSE) filters proposed earlier for SAR processing of sparse-satellite clusters is discussed. The need for the implementation of a new filter for the specific problem is discussed.

In chapter three, a detailed description of the methodology used for implementation of KF for processing of data obtained from multi-aperture spaceborne radar is given. The performance of the developed KF algorithm for different scattering characteristics is investigated. A comparison study is made between the existing filters and the developed KF filter for different system parameters such as SNR and number of receivers is investigated. The improvement obtained in the data processing as result of the KF is discussed.

In chapter four, a detailed analysis is presented regarding the performance and characteristics of the developed KF algorithm. The variation of the filter parameters with processing is discussed. Explanation is given regarding the experiments performed to observe the dependence of the final estimate on the input parameters to the KF algorithm, and the results are discussed. Some significant conclusions are drawn regarding the optimal input parameters to the KF algorithm.

In chapter five, different methods for decreasing the overall computation time and processing load of KF are investigated. Initially the order of the various matrix operations performed in the filter implementation is chosen in an optimal way to reduce the number of required computations. Timing plots are provided to signify the importance of the number of divisions made of the measurement vector for the different iterative implementations possible. Experimental results are presented to explain the different methods pursued to reduce the number of computations taken for operations relating to the error correlation matrix. A method to quantify the amount of measurements required for optimal processing is proposed. There are still many inherent problems relating to the computational load and processing time in the overall filter implementation. The enhancements that could be made to decrease the severity of the problems are discussed.

In Chapter six, the necessity and use of parallel processing is discussed. The methodology behind the parallel implementation of KF is described in detail. An explanation is given regarding the changes made in the parallel implementation to accommodate the huge data sizes of the matrices present in the system. Finally results are shown to represent the enhancement obtained in the overall data processing. In chapter seven, conclusions and recommendations for future work are made.

2. Processing of Multiple-Aperture Space-borne Arrays for Wide-Area SAR

2.1 Signal Space Representation of the Radar System

The radar geometry is shown in Figure 2.1 The space-borne system travels in the positive *x*-direction at velocity, v , and the array phase reference at time zero is located at the origin of the coordinate system. Therefore, assuming a flat earth, the *z*-coordinate of all targets on the ground is $-h$, where h is the altitude of the array phase reference. There is a single transmitter located at the array phase reference.

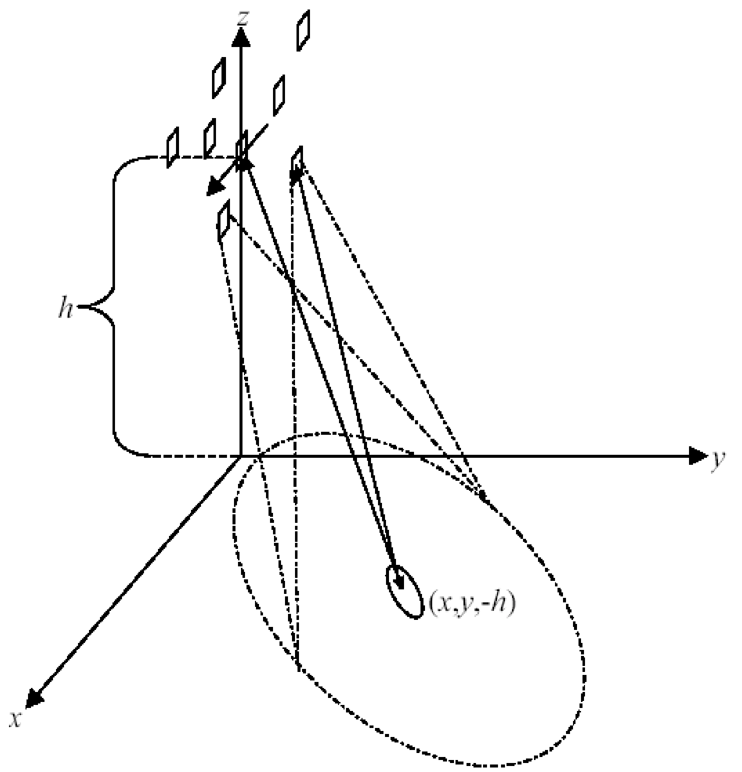

Figure 2.1 Radar Geometry for a constellation of radar satellites with the flat earth approximation [8]

In this section a signal-space representation of the radar system is presented [6]. The signal-space representation presented here helps in implementation of estimation algorithms through well-established linear algebra techniques and facilitates interpretation of the SAR filters that are used. The complex signal that a radar constellation measures, $r(\bar{x}_r, t)$, can be written as [6]

$$r(\bar{x}_r, t) = \int_A g_0(\bar{x}_r) \int_T h(\bar{x}_r, \bar{x}, t, t')\, s(t')\, dt'\, dA + n(\bar{x}_r, t). \tag{2.1}$$

Where $\bar{x}_r$ is the position vector to the receiver, $\bar{x}$ is the position vector describing surface location, $g_0(\bar{x})$ is the complex reflectance per unit area at $\bar{x}$, $h(\bar{x}_r, \bar{x}, t, t')$ is a complex, time-variant function impulse response describing the propagation from the moving radar to the surface and back, $s(t')$ is the complex representation of the transmitted signal, T is the time over which the transmitted signal exists, A is the radar's illumination area, and $n(\bar{x}_r, t)$ is complex noise. The integration in time is performed over the length of the transmit signal and is represented by $r(\bar{x}_r, \bar{x}, t)$ to get

$$r(\bar{x}_r, t) = \int_A g_0(\bar{x})\, r(\bar{x}_r, \bar{x}, t)\, dA + n(\bar{x}_r, t). \tag{2.2}$$

$r(\bar{x}_r, \bar{x}, t)$ is basically the complex response received by a receiver positioned at $\bar{x}_r$ due to an imaginary scatterer of unit magnitude positioned at $\bar{x}$ .Therefore, the response from a differential area on the ground, which depends on its position as well as the transmit signal, is characterized by $r(\bar{x}_r, \bar{x}, t)$ to within a multiplicative constant. The signal received at each receiver can be viewed as a superposition of the responses from each point of the illuminated surface ($\rho(\bar{x}_r,\bar{x},t)$), weighted by the complex scattering amplitude of the surface. If the integration in (2.2) is approximated by a summation, the received signal becomes

$$r(\bar{x}_r,t) = \sum_i \gamma_0(\bar{x}_i)\rho(\bar{x}_r,\bar{x}_i,t)\Delta A + n(\bar{x}_r,t). \tag{2.3}$$

where ΔA defines a section of surface area that is less than or equal to the resolution of the radar. The index i includes all discrete areas ΔA that are illuminated and $\bar{x}_i$ is the position vector to the center of the ith discrete area. If the signal is sampled in space according to the position of the receiver in the constellation and in time according to the signal's bandwidth, the mth sample at the nth receiver becomes

$$r(\bar{x}_r^n,t_m) = \sum_i \gamma_0(\bar{x}_i)\rho(\bar{x}_r^n,\bar{x}_i,t_m)\Delta A + n(\bar{x}_r^n,t_m). \tag{2.4}$$

Finally, the entire set of measurements can be represented using matrix-vector notation

$$\mathbf{r} = \mathbf{P}\boldsymbol{\gamma} + \mathbf{n} \tag{2.5}$$

where

$$\left.\begin{aligned}
\mathbf{r} &= [r(\bar{x}_r^1,t_1) \quad r(\bar{x}_r^1,t_2) \quad \cdots \quad r(\bar{x}_r^N,t_{BT})]^\dagger \\
\boldsymbol{\gamma} &= [\gamma_1 \quad \gamma_2 \quad \cdots \quad \gamma_M]^\dagger \\
\mathbf{n} &= [n(\bar{x}_r^1,t_1) \quad n(\bar{x}_r^1,t_2) \quad \cdots \quad n(\bar{x}_r^N,t_{BT})]^\dagger \\
\gamma_i &= \gamma_0(\bar{x}_i)\Delta A
\end{aligned}\right\} \tag{2.6}$$

The elements of $\mathbf{r}$ and $\mathbf{n}$ are the time sampled values of $r(t)$ and $s(t)$ for each receiver. $(\cdot)^\dagger$ is the matrix or vector transpose, BT is the received time-bandwidth product of a single receiver, and N is the number of receivers in the constellation.

$$\left.\begin{aligned}\mathbf{P} &= [\mathbf{r}_1 \quad \mathbf{r}_2 \quad \cdots \quad \mathbf{r}_M] \\ \mathbf{r}_i &= [r(\bar{x}_r^1,\bar{x}_i,t_1) \quad r(\bar{x}_r^1,\bar{x}_i,t_2) \quad \cdots \quad r(\bar{x}_r^N,\bar{x}_i,t_{BT})]^\dagger\end{aligned}\right\} \tag{2.7}$$

The *response vector*, $\mathbf{r}_i$, is the full set of measurements obtained by a radar at a discrete set of space and time locations, again due to a unit reflectance scatterer. M is the number of resolution cells of size ΔA. If the number of resolution cells M exceeds the number of measurements, NBT, then any estimate of the complex RCS (Radar Cross Section) vector $\mathbf{g}$ will contain error due to ambiguities.

The radar response can therefore be represented as a summation of the responses from each resolution cell $\mathbf{r}_i$, weighted by the complex scattering amplitude of each cell g_i plus noise. This representation helps in direct application of linear algebraic techniques to the radar problem.

Given that the radar parameters (e.g., position, velocity, wavelength, transmit signal, antenna pattern) are known accurately, the response vectors $\mathbf{r}_i$ are known *a priori*. The SAR problem is therefore to estimate the values in the RCS vector $\mathbf{g}$, given some measurement vector $\mathbf{r}$ contaminated by noise. As seen in (2.5) the radar process can be approximated in a linear fashion. So an estimator is required that could be applied as a linear process. A weight vector, or filter, is found for each resolution cell. When the inner product of the received measurements is taken with each of the weight vectors, the estimated RCS vector $\hat{\mathbf{g}}$ is

$$\hat{\mathbf{g}} = \mathbf{W}\mathbf{r} \tag{2.8}$$

where
$$\hat{\mathbf{g}} = [\hat{g}_1 \quad \hat{g}_2 \quad \cdots \quad \hat{g}_M] \tag{2.9}$$

and $$\mathbf{W} = [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \cdots \quad \mathbf{w}_M]^{\mathbf{H}} \tag{2.10}$$

w_i is the weight vector for the ith resolution cell, and $(\cdot)^{\mathbf{H}}$ denotes the conjugate-transpose operation. It is also important to note that this is the type of processing traditionally done in SAR, where $\mathbf{w}_i$ is typically matched to measurements received from the ith resolution cell. Although some algorithms require more computation for finding each $\mathbf{w}_i$, the data-dependent process of calculating inner products is equivalent for all linear estimators.

2.2 Single-Aperture SAR

It could be observed from (2.8) that a necessary condition for estimating $\mathbf{g}$ is that the dimension of $\mathbf{r}$ must be equal to or more than that of $\mathbf{g}$. In other words, to estimate the scattering of N number of illuminated resolution cells (or image pixels), a minimum of N number of independent measurements must be acquired by the radar system. In spotlight mode, SAR focuses on a particular area for some time, T. During that time, the maximum rate at which independent complex samples can be collected depends on the signal bandwidth B. Therefore, the maximum number of independent, complex samples that be collected is equal to BT, also known as the time-bandwidth product. The problem is that bandwidth and observation time are fixed by the resolution requirements of the system. The range resolution requirement determines bandwidth, and the azimuth resolution requirement determines observation time. Since it is only possible to image unambiguously as many targets as there are independent samples, and since the number of independent samples is related through bandwidth and time to particular range and azimuth resolutions, the maximum imaging area is fixed. A simple example for a side-looking, spotlight SAR is presented in the following. Suppose the resolution requirements imposed on the radar are Δx and ΔR in azimuth and range, respectively. Range resolution is given by [12]

$$\Delta R = \frac{c}{2B} \tag{2.11}$$

where c is the speed of light. For a 90^0 side-looking geometry where the azimuth extent is small compared to the range, the azimuth resolution can be approximated as

$$\Delta x \approx \frac{\lambda_0 R_0}{2v}\frac{1}{T} \tag{2.12}$$

where λ_0 is the wavelength at the center operating frequency, R_0 is the average target range, and v is the along-track velocity of the radar platform. The area per pixel is then the product of the azimuth and range resolutions

$$\Delta x \Delta R = \frac{c\lambda_0 R_0}{4v}\frac{1}{BT}. \tag{2.13}$$

The maximum area is the time-bandwidth product multiplied by the area per pixel

$$A_{\max} = BT\,\Delta x \Delta R = \frac{c\lambda_0 R_0}{4v}. \tag{2.14}$$

As can be seen from the right hand side of (2.14), the maximum area that can be imaged is determined by range, wavelength, and platform velocity.

One solution appears to be increasing the time-bandwidth product. However (2.11) and (2.12) clearly show that the resolution dimensions are inversely proportional to bandwidth and time. For example, by increasing bandwidth and time each by a factor of two, it is possible to get four times as many independent samples, but the resolution pixels would be half the original size on each side. This would result in four times as many samples, but the area of each pixel would be four times, smaller, and the total area would remain the same.

2.3 Multiple-Aperture SAR

A necessary requirement for increasing SAR map area is to increase the number of independent samples, or amount of information, that is collected without modifying resolution cell size in the process. We have seen that this cannot be met by increasing bandwidth or illumination time. So isn't there a way around this? Lets see, the basic requirement is to increase the amount of information collected, and we know that it not possible to increase the information collected by a single receiver. So, the only way that this problem could be solved is by adding more number of receivers. Since sensor resolution is determined by time (i.e., delay and Doppler) information, it is evident that additional spatial information can be used to increase the dimension of measured vector $\mathbf{r}$. By adding more antenna apertures to the SAR system, each with its own receiver, angle of arrival information can be collected. If N is the total number of receive apertures, then the number of independent samples available to the system is now NBT . The maximum area is again the product of the number of independent samples and the area per pixel

$$A_{\max} = NBT\Delta x\Delta R = N\frac{c\lambda_0 R_0}{4v} \tag{2.15}$$

which is N times larger than was possible with a single receiver. Furthermore, it is noted that the angle of arrival information is unique from the time and frequency information, making it possible to discriminate range-Doppler ambiguities.

It should be noted that the minimum aperture area requirement that is associated with single aperture SAR is still retained in Multi-Aperture SAR. The advantage of the Multi-apertures is that this limit on aperture area does not limit the size of the illuminated area or the extent of the resulting image.

2.4 Sparse Arrays

The microsat concept calls for placing each receive aperture on its own, smaller satellite. Furthermore, the orbital dynamics of formation flying require the satellites to have significant, random spacing between them. Therefore, the microsat array is sparsely populated, and different spatial processing must be applied. It may be possible to temporarily propel the microsats into a regular, well-formed pattern, but the amount of propulsion necessary to maintain a regular formation puts this option beyond any reasonable fuel budget. So the apertures are randomly placed in three dimensions. It is assumed that the apertures have the same azimuth and elevation angles at boresight; therefore, the satellites' illumination patterns on the ground are assumed to be identical.

2.5 Correlation or Matched Filter

The vector representation of the correlation filter for the *i*th resolution cell is the weighted conjugate transpose of its measurement vector

$$\mathbf{w}_i^{corr} = \frac{\mathbf{r}_i}{\left|\mathbf{r}_i\right|^2} \tag{2.16}$$

where the superscript denotes that the filter is a correlation filter. If the matched filters for all resolution cells are placed into the columns of a matrix as shown in (2.10), then the matched-filter estimator, W_{corr}, is given by

$$\mathbf{W}_{corr} = \mathbf{D}^{-1}\mathbf{P}^{\mathbf{H}} \tag{2.17}$$

where $\mathbf{P}$ is described in (2.7), and $\mathbf{D}$ is a diagonal matrix defined as

$$\mathbf{D} = \begin{bmatrix} \mathbf{r}_1^{\mathrm{H}}\mathbf{r}_1 & 0 & \cdots & 0 \\ 0 & \mathbf{r}_2^{\mathrm{H}}\mathbf{r}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbf{r}_M^{\mathrm{H}}\mathbf{r}_M \end{bmatrix}. \tag{2.18}$$

When the matched filter is applied to the measurement received, the estimate of the i th target is

$$\hat{g}_i = \left(\mathbf{w}_i^{corr}\right)^{\mathrm{H}}\mathbf{r} = g_i + \sum_{j,j\neq i}\left(\mathbf{w}_i^{corr}\right)^{\mathrm{H}}\mathbf{r}_j g_j + \left(\mathbf{w}_i^{corr}\right)^{\mathrm{H}}\mathbf{n}. \tag{2.19}$$

As could be seen in (2.19), the estimate error in the matched filtering operation comes from the last two terms. The last term represents error due to noise.

The matched filter minimizes this term, by maximizing the received energy with respect to noise. The matched-filter vector has the smallest magnitude of any filter vector that gives g_i as its expected result. Therefore, the matched filter has the least noise power at its output of any linear filter.

The second term in (2.19) represents the error due to the correlation of the weight vector with all other illuminated targets. The matched filter does nothing to minimize this term, and therefore, if the responses from the illuminated targets are significantly correlated, a correspondingly large error will result. Hence, it could be seen that the matched filter does not provide optimal estimates for cases that are clutter limited rather than noise limited. However, it is this lack of dependence on clutter that also makes the matched-filter vectors the least computationally expensive to generate.

Generally, for the sparse array case, the responses of some resolution cells will be significantly correlated. As matched filter is unable to eliminate error due to clutter so it becomes very ineffective for this case. Thus, a different linear processor must be implemented.

2.6 Maximum-Likelihood Filter

The Maximum Likelihood (ML) estimator provides the estimate of $\mathbf{g}$ that maximizes the likelihood function $p\left({}^{\mathbf{r}}\!/_{\mathbf{g}}\right)$, where $p\left({}^{\mathbf{r}}\!/_{\mathbf{g}}\right)$ is the conditional probability density (pdf) of $\mathbf{r}$ given $\mathbf{g}$ and is given as

$$p\left(\mathbf{r}/\mathbf{g}\right) = \frac{1}{\sqrt{|2p\mathbf{K}_n|}} \exp\left[-\frac{1}{2}(\mathbf{r} - \mathbf{Pg})^H \mathbf{K}_n^{-1} (\mathbf{r} - \mathbf{Pg})\right] \tag{2.20}$$

Using (2.20), the maximum-likelihood (ML) estimator is obtained by maximizing the argument of the exponential function

$$\max_{\mathbf{g}} -\frac{1}{2}(\mathbf{r} - \mathbf{Pg})^{\mathbf{H}} \mathbf{K}_n^{-1} (\mathbf{r} - \mathbf{Pg}). \tag{2.21}$$

The ML estimator then becomes

$$\mathbf{W}_{ml} = \left(\mathbf{P}^{\mathbf{H}}\mathbf{K}_n^{-1}\mathbf{P}\right)^{-1} \mathbf{P}^{\mathbf{H}}\mathbf{K}_n^{-1}. \tag{2.22}$$

If it is assumed that the noise samples are independent, then the noise covariance matrix is diagonal

$$\mathbf{K}_n = s_n^2 \mathbf{I} \tag{2.23}$$

where $\mathbf{I}$ is the identity matrix. The ML estimator then reduces to

$$\mathbf{W}_{ml} = \mathbf{P}^{\sim 1} \tag{2.24}$$

where $(\times)^{\sim 1}$ denotes the pseudo inverse operation. The inverse in this case is a pseudo inverse, since the dimension of $\mathbf{r}$ exceeds that of $\mathbf{g}$.

The estimate of the *i*th pixel's RCS due to the ML filter is then

$$\hat{g}_i = \left(\mathbf{w}_i^{ml}\right)^{\mathbf{H}} \mathbf{r} = g_i + \left(\mathbf{w}_i^{ml}\right)^{\mathbf{H}} \mathbf{n}. \tag{2.25}$$

Comparing the forms of (2.25) and (2.19), it is seen that the clutter term is absent in (2.25). The last term in (2.25) however, becomes important. The pseudo inverse operation is basically a de-convolution operation, and generally results in inferior performance for the single aperture SAR. This is due to the fact that for a single aperture radar the dimension of $\mathbf{r}$ is not significantly more than that of $\mathbf{g}$ resulting in not so well conditioned $\mathbf{P}$ matrix. If $\mathbf{P}$ is ill conditioned (close to singular), then the pseudo inverse operation is not dependable. However, for multi-aperture SAR, the dimension of the measurement vector $\mathbf{r}$ can significantly exceed that of $\mathbf{g}$ so that $\mathbf{P}$ is generally well conditioned.

The accuracy of the estimate obtained by ML filter is largely dependent on the SNR. For high SNR cases the processor finds a weight vector that is orthogonal to the responses of all other pixel targets, thus eliminating the clutter. However, since this orthogonal filter does nothing to minimize noise, the processing performance diminishes as SNR drops. Thus, SNR and the condition of $\mathbf{P}$ matrix become crucial factors in determining the accuracy of the estimate obtained through the ML filter. There is also a huge increase in the computational load required to calculate $\mathbf{W}$ due to the pseudo inverse operation.

An ideal processor therefore would maximize the Signal-to-Interference Ratio (SIR), where interference is defined as the sum of both clutter and noise energy.

2.8 MMSE Filter

Given that *a priori* knowledge of the radar SNR is available, a minimum mean-squared error estimator can be implemented. The MMSE filter maximizes the SIR and acts as an ideal processor. The MMSE filter therefore could be said to be the mathematically optimum compromise between the correlation and ML filters.

The derivation of the linear MMSE filter is based on the orthogonality principle, and could be finally written as [15]

$$\mathbf{W}_{mmse} = E\left[\boldsymbol{\gamma}\boldsymbol{\gamma}^{\mathbf{H}}\right]\mathbf{P}^{\mathbf{H}}\left(\mathbf{P}E\left[\boldsymbol{\gamma}\boldsymbol{\gamma}^{\mathbf{H}}\right]\mathbf{P}^{\mathbf{H}} + E\left[\mathbf{n}\mathbf{n}^{\mathbf{H}}\right]\right) \tag{2.26}$$

If the elements of the RCS vector, $\boldsymbol{\gamma}$, are assumed to be independent with identical statistics, and $E\left[\mathbf{n}\mathbf{n}^{\mathbf{H}}\right]$ is recognized as the noise covariance matrix, $\mathbf{K}_n$, then (2.26) reduces to

$$\mathbf{W}_{mmse} = \gamma_t^2\mathbf{P}^{\mathbf{H}}(\gamma_t^2\mathbf{P}\mathbf{P}^{\mathbf{H}} + \mathbf{K}_n)^{-1} \tag{2.27}$$

where the expected value of the squared RCS magnitude for each target is $\gamma_t^2 = E\left[\gamma_i^{\mathbf{H}}\gamma_i\right]$

Some important insight gained from (2.27) about the behavior of the MMSE filter. First, in a low-noise or zero-noise case, $\mathbf{K}_n$ will be negligible and $\mathbf{W}_{mmse}$ becomes

$$\mathbf{W}_{mmse} \approx \gamma_t^2\mathbf{P}^{\mathbf{H}}\frac{1}{\gamma_t^2}\left(\mathbf{P}^{\mathbf{H}}\right)^{-1}\mathbf{P}^{-1} = \mathbf{P}^{-1} \tag{2.28}$$

which is the same for the ML filter. In the low noise case, therefore, the MMSE filter maximizes SCR. However, in high-noise case, $\mathbf{K}_n$ dominates and $\mathbf{W}_{mmse}$ becomes

$$\mathbf{W}_{mmse} \approx \gamma_t^2\mathbf{P}^{\mathbf{H}}\mathbf{K}_n^{-1} = \frac{\gamma_t^2}{\sigma_n^2}\mathbf{P}^{\mathbf{H}} \tag{2.29}$$

It could be seen from (2.29) that the MMSE filter vector is in similar direction to that of the Matched-filter vector for low-SNR case. An important difference, however, is that the MMSE filter becomes inversely proportional to the noise variance. Hence the magnitude of the MMSE filter approaches zero, as the noise variance approaches infinity. Therefore it is inherent in the equation for the MMSE filter that in the presence of overwhelming noise, it is best to estimate the RCS values not by the measurements but by the statistical properties of the targets.

Though MMSE filter gives good results at both low SNR and low SCR scenarios, the major problem with it is the huge computation and processing load introduced in the system due to the inverse. The problem is hugely aggravated in radar signal processing where huge amounts of data makes the computation of inverse a very difficult and time consuming process. Taking this into consideration, an iterative implementation of the MMSE algorithm, known as Kalman Filtering is developed and investigated in the coming sections. Moreover, as there is a possibility in Kalman Filtering to model the uncertainty in the parameter that is to be estimated, it is expected that there will be an overall decrease in the error of the estimate.

3. KALMAN FILTER IMPLEMENTATION

The main motivation factor behind the development and implementation of KF to data obtained from sparse aperture radars is to decrease the computation load inherent in the MMSE filtering while being able to obtain similar or even better estimates in all types of SNR and SCR scenarios. KF is basically an iterative implementation of the MMSE filtering, thereby reducing the complexity involved with calculating inverse for a huge matrix. In this section a brief review of the Kalman Filter Theory along with a detailed explanation of the implementation of KF for the present scenario is given. The performance of the developed KF along with a comparison to the filters proposed earlier is also provided.

3.1 Kalman Filter Theory

R.A. Fisher introduced the idea of maximum likelihood estimation and this has provided the platform for future developments in estimation theory [16]. Kolmogorov in 1941 and Wiener in 1942 independently developed a linear minimum mean-square estimation technique that received considerable attention and provided the foundation for the subsequent development of Kalman filter theory [16]. Kalman published his first paper on discrete-time, recursive mean square filtering in 1960 [17]. A summary of the Kalman filter problem and its solution is provided in this section. The system is composed of two essential ingredients, the *state* or *process equation* and the *measurement* or *observation equation*.

The state equation models the expected variation in the parameter $\mathbf{f}(i)$ that is to be estimated, during the period of time of the measurement process

$$\mathbf{f}(i) = \mathbf{C}(i)\mathbf{f}(i-1) + \mathbf{m}(i) \tag{3.1}$$

where $\mathbf{C}(i)$ known as the 'state transition matrix', models the expected variation in the value of $\mathbf{f}(i)$ from instant $i-1$ to i. $\mathbf{m}(i)$ known as process noise expresses the uncertainty in the modeling of this expected variation and is modeled as guassian noise.

The observation equation relates the obtained measurements to its state and is of the form,

$$\mathbf{y}(i) = \mathbf{B}(i)\mathbf{f}(i) + \mathbf{v}(i) \tag{3.2}$$

where $\mathbf{B}(i)$ describes the relationship between the signal vector and the observation vector. $\mathbf{v}(i)$ represents the measurement errors that occur at each observation time and is modeled as guassian noise.

The Kalman filtering problem, namely, the problem of jointly solving the state and observation equations for the unknown state in an optimal manner may be formally stated as follows:

The entire observed data, consisting of the observations $\mathbf{y}(1), \mathbf{y}(2), \cdots \mathbf{y}(i)$, *is used to find, for each* $b \geq 1$, *the minimum mean-square estimate of the state* $f(a)$.

The problem is called filtering if $a = i$, *prediction if* $a > i$, *and smoothing if* $1 \leq a < i$ [11].

In the next section a detailed explanation is given about the application of the KF problem to SAR processing of Multi-Aperture spaceborne radar and the corresponding KF solution.

3.2 Kalman Filter Implementation:

In this section a detailed explanation is given for the implementation of Kalman filtering for radar processing.

From (2.5) we have

$$\mathbf{r} = \sum_{m}^{M} g_m \mathbf{r}_m + \mathbf{n} \tag{3.3}$$

The Kalman filter is an iterative implementation of the MMSE estimator. So, it can be implemented by operating on sections of the radar response vector $\mathbf{r}$, rather than on the entire length of the vector.

So the radar response vector $\mathbf{r}$ is divided into L smaller vectors. For example $\mathbf{r}$ could be divided into smaller vectors, each with a dimension of 3 as shown below.

$$\mathbf{r} = \left[\underbrace{r_1, r_2, r_3}_{\mathbf{r}(1)}, \underbrace{r_4, r_5, r_6}_{\mathbf{r}(2)}, \ldots\ldots, \underbrace{r_{3(L-1)+1}, r_{3(L-1)+2}, r_{3L}}_{\mathbf{r}(L)} \right] \tag{3.4}$$

The normalized response vectors can equivalently be segmented into smaller sections:

$$\mathbf{r}_m = \left[\mathbf{r}_m^{\dagger}(1), \mathbf{r}_m^{\dagger}(2), \ldots, \mathbf{r}_m^{\dagger}(L) \right]^{\dagger} \tag{3.5}$$

Therefore $\mathbf{P}$ can also be defined as

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}_1(1) \\ \mathbf{p}_2(2) \\ \vdots \\ \mathbf{p}_M(L) \end{bmatrix} \tag{3.6}$$

Finally, the noise vector can likewise be segmented:

$$\mathbf{n} = \left[\mathbf{n}(1)^{\dagger}, \mathbf{n}(2)^{\dagger}, \ldots \mathbf{n}(L)^{\dagger}\right]^{\ddagger} \tag{3.7}$$

Now, the parameter that is to be estimated in this case is $\boldsymbol{\gamma}$ and $\mathbf{r}$ is the observation vector. Basing on (3.1) the state equation for this application can be written as

$$\boldsymbol{\gamma}(l) = \mathbf{A}(l)\boldsymbol{\gamma}(l-1) + \mathbf{u}(l) \qquad \textit{State Equation} \tag{3.8}$$

where l represents the iteration number or the section of data (out of a total of L sections of data) on which the processing is being done. So, l varies from 1 to L. $\mathbf{A}(l)$ is the state transition matrix described in (3.1) and $\mathbf{u}(l)$ is process noise and represents the uncertainty in $\mathbf{A}(l)$.

The equation (3.5) gives a linear relationship between the measurement vector and the state vector and is similar to the observation equation (3.2). Therefore, the observation equation for this case can be written as

$$\mathbf{r}(l) = \mathbf{P}(l)\boldsymbol{\gamma}(l) + \mathbf{n}(l) \qquad \textit{Observation Equation} \tag{3.9}$$

These measurements are obtained over time, space, and frequency. It is generally assumed that the state vector comprising of the scattering coefficients γ_m are *approximately constant* with respect to time, space, and frequency over the extent of the radar measurement. In other words the scattering from the illuminated targets can be assumed to be constant. Therefore state transition matrix $\mathbf{A}(l)$ will be an identity matrix and the state equation of (3.8) can be written as:

$$\mathbf{g}(l/l-1) = \mathbf{g}(l-1/l-1) + \mathbf{u}. \tag{3.10}$$

Basing on (3.9) and (3.10) it is possible to describe the Kalman filtering process. After processing $\mathbf{r}(l-1)$, information is available about $\hat{g}(l-1/l-1)$ and the error correlation matrix $\mathbf{K}_g(l-1/l-1)$. The way in which this information could be used to update the estimate is given below. Initially the state-error correlation matrix (it is referenced as error correlation matrix in the thesis document) is updated as follows

$$\begin{aligned}\mathbf{K}_g(l/l-1) &= \mathrm{E}\left\{(\mathbf{g}(l)-\hat{\mathbf{g}}(l/l-1))(\mathbf{g}(l)-\hat{\mathbf{g}}(l/l-1))^{\mathbf{H}}\right\} \\ &= \mathbf{A}(l)\mathbf{K}_g(l-1/l-1)\mathbf{A}(l)^{\mathbf{H}} + \mathbf{K}_u(l).\end{aligned} \tag{3.11}$$

Now as $\mathbf{A}(l)$ is taken as an identity matrix, therefore $\mathbf{K}_g(l/l-1)$ becomes

$$\mathbf{K}_g(l/l-1) = \mathbf{K}_g(l-1/l-1) + \mathbf{K}_u(l). \qquad \textit{Step 1} \tag{3.12}$$

where $\mathbf{K}_u(l)$ the correlation matrix of process noise $\mathbf{u}$

$$\mathbf{K}_u(l) = \mathrm{E}\left\{\mathbf{u}(l)\mathbf{u}(l)^{\mathbf{H}}\right\}. \tag{3.13}$$

The gain matrix $\mathbf{G}(l)$ is computed so that the mean square error is minimized, which basing on the orthogonality theorem is minimum when the error in the estimate is orthogonal to the measurement data.

$$\mathbf{G}(l) = \mathbf{K}_g(l/l-1)\mathbf{P}(l)^{\mathbf{H}}\left[\mathbf{P}(l)\mathbf{K}_g(l/l-1)\mathbf{P}(l)^{\mathbf{H}} + \mathbf{K}_n(l)\right]^{1} \qquad \textit{Step 2} \tag{3.14}$$

where $$\mathbf{K}_n(l) = \mathrm{E}\left\{\mathbf{n}(l)\mathbf{n}(l)^{\mathbf{H}}\right\}. \tag{3.15}$$

The next step involves the computation of error between the expected measurement, which could be obtained from the previous estimate, and the observed measurement. This error obtained $\mathbf{u}(l)$ is termed as the *innovation* and the process is termed as the *innovation process* and is as follows

$$\mathbf{u}(l) = \mathbf{r}(l) - \mathbf{P}(l)\hat{g}(l-1/l-1) \qquad \textit{Step 3} \qquad (3.16)$$

The innovation $\mathbf{u}(l)$ associated with the observed data $\mathbf{r}(l)$ representing the essence of the Kalman Filtering Theorem [17], is orthogonal to all the past observations and so could be used to make an optimal estimate of $\mathbf{g}$.

The estimate is updated basing on the Kalman Gain and innovation as follows

$$\hat{g}(l/l) = \hat{g}(l-1/l-1) + \mathbf{G}(l)\mathbf{u}(l). \qquad \textit{Step 4} \qquad (3.17)$$

Finally the error correlation matrix is updated as follows

$$\mathbf{K}_g(l/l) = [\mathbf{I} - \mathbf{G}(l)\mathbf{P}(l)]\mathbf{K}_g(l/l-1). \qquad \textit{Step 5} \qquad (3.18)$$

Steps 1, 2, 3, 4 and 5 are repeated until all the radar measurements are used.

The initial $g(0)$, $\mathbf{K}_g(0)$, and $\mathbf{K}_u(0)$have to be set to initiate the filtering process.

The implementation is initially done in Matlab and is later implemented in C for optimal memory utilization. Later PVM (Parallel Virtual Machine) software is used to parallelize the most computationally intensive part of the implementation.

3.3 Performance of the developed Kalman Filter

In this section the test results of the developed KF for a general scattering scenario is presented. The radar data is obtained from a multi-aperture spaceborne

Radar System Height	183000 m
Center frequency of transmit signal	10 GHz
Radar System Velocity	7800 m/s
Tilt of apertures to target	0.7855 (45^0)
Look angle of antenna pattern from normal to aperture	0^0
Pulse Repetition Frequency	20.645kHz

Table 3.1 Input parameters for the Radar Simulator that are kept constant

radar simulator developed earlier [7]. The system parameters given as input to the radar simulator reflect the physical parameters of an actual Low earth orbiting (LEO) satellite and hence present a realistic condition under which the simulator operates. Table 3.1 shows the input parameters that are fixed for all the simulations done in the thesis investigation.

A number of parameters like *M*, the number of resolution cells considered, the number (*N*) and position of apertures, the number of frequency points and the number of time samples can be varied in the radar simulator. Table 3.2 shows the input parameters to the radar simulator for a specific *M* value of 1024. Where the number of receivers can again be varied. Different types of target scenarios can be given as input to the radar simulator by means of an image with the pixels representing the scattering characteristics of the targets. The receiver configuration is chosen to be sparse and randomly spaced. The parameters that are chosen remain constant for all experiments presented in the document.

Number of Receivers in the Constellation	13
Number of pixels in target along x axis	32
Number of pixels in target along y axis	32
Number of samples taken	400
Length of apertures in along-track dimension	0.13 m
Length of apertures in across-track dimension	0.18 m

Table 3.2 Input parameters for the Radar Simulator for *M*=1024

3.3.1 Performance of KF for random scattering coefficients

Will the developed KF be able to make a good estimate of the scattering coefficients $\mathbf{g}$ irrespective of the complex nature of $\mathbf{g}$ for measurements obtained from a sparsely located multiple aperture spaceborne radar? To answer this question the KF is tested for a scenario where the scattering coefficients are chosen randomly. Each of the scattering coefficients is given a random magnitude and a random phase. The magnitudes are chosen as a random value out of a 256-level gray scale. An image is generated by substituting the complex values of $\mathbf{g}$ for the corresponding pixel values. This image representing the scattering coefficients of all target pixels is given as input to the radar simulator. The numbers of pixels considered in the image are 32 by 32 or 1024 (= *M*). The time bandwidth product of the received signal was 400; therefore, only about 0.39 of the original 32 by 32 input can be unambiguously imaged by a single aperture system.

Guassian noise was added to the obtained radar measurement vector $\mathbf{r}$. The signal to noise ratio (SNR) is taken to be 30dB, where signal is the radar response vector. The SNR is defined as the ratio of the average signal power in a single data sample to the average power of a single noise sample. The radar measurements obtained after adding the Guassian noise is given as input to the developed KF. The normalized vectors $\mathbf{r}_m$ representing the normalized radar response obtained from *m*th scatterer position are also obtained from the radar simulator. The total measurement vector consisting of 5200 ($13(= N)' \ 400(= BT)$) vectors is divided into 10 ($= L$) smaller vectors.

The input parameters to the KF are chosen as follows:

Basing on the assumption that the reflectance values from pixel (1 resolution cell) to pixel are uncorrelated and the mean scattering value is zero, the initial error correlation matrix becomes $\mathbf{K}_g(0) = \mathrm{E}\{\mathbf{g}\mathbf{g}^{\mathrm{H}}\} = s_g^2\mathbf{I}$ where s_g^2 is the expected value of the squared reflectance magnitude for each target. The average pixel magnitude for a 256-level gray scale is 128. So the expected value of the squared reflectance magnitude for each target will be 128^2 for this particular case.

The process noise correlation matrix $\mathbf{K}_u(0)$, which expresses the uncertainty in the state transition matrix is neglected, on the assumption that the scattering obtained from each target for different space, time and frequencies remains constant. The validity of this assumption is reviewed in later sections of the report.

Now the expected value of the average of the complex scattering coefficients is zero and so the initial value for $\mathbf{g}(0)$ is taken to be a null vector.

The performance of the KF is expressed in terms of the variation in error vs. the fraction of data processed ε as it gives information regarding both the final estimate and the rate of convergence. The error criterion chosen is the mean-squared error (MSE) of the pixel magnitudes normalized by the image's mean-squared pixel magnitude

$$\mathbf{MSE} = \frac{(\hat{\mathbf{g}} - \mathbf{g})^{\mathbf{H}} (\hat{\mathbf{g}} - \mathbf{g})}{\mathbf{g}^{\mathbf{H}} \mathbf{g}}. \tag{3.19}$$

Figure 3.1 shows the variation of the Normalized MSE with the fraction of measurement vector processed. The fraction of measurement vector processed, ε is given by

$$e = \frac{l}{L}. \tag{3.20}$$

where l, is the number of data vectors processed by KF out of a total of L vectors. The KF uses the information contained in the radar measurements to make a good estimate of $\mathbf{g}$. The Kalman Gain $\mathbf{G}(l)$, for each iteration, (l) is computed in a way so as to reduce the MSE. It could be seen from Figure 3.1 that the MSE initially decreases rapidly with ε. This could be attributed to the fact that the initial information contained in the measurements helps in eliminating the clutter associated with the measurements to resolve the range and Doppler ambiguities. But once these ambiguities are resolved and the clutter eliminated the information obtained from the new measurements is used by the filter to eliminate the noise. The slow rate of decrease of MSE at the end of the processing could be attributed to the elimination of the noise.

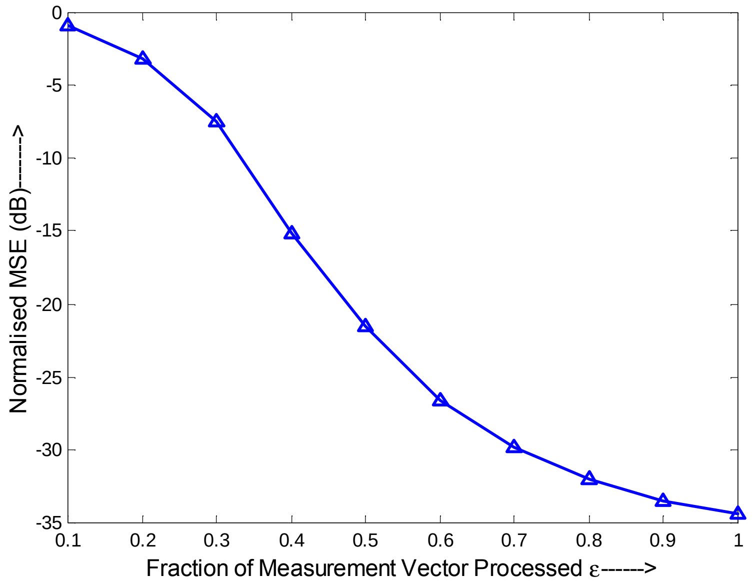

Figure 3.1 Performance of the developed KF for randomly chosen scattering coefficients.

The final estimate $\mathbf{g}(L)$ after all the measurements are processed is quite close to the actual $\mathbf{g}$ as could be seen from the low value of MSE at the end. From this, we could conclude that the developed KF is independent of the complex nature of $\mathbf{g}$ and is able to make a good estimate by using the measurements obtained from sparsely located multi-aperture spaceborne radar.

3.3.2 Performance of KF for a real SAR Scenario

In this section the performance of the developed KF can be seen in action. Figure 3.2 shows the magnitude of the image given as input to the radar simulator. It was taken from a photograph of the football stadium of the University of Kansas, in Lawrence, Kansas. The photograph was scanned, changed to a 256-level grayscale image, and cropped to be 64 by 64 pixels. The pixel intensities were given a random phase, and the resulting set of complex pixel values was used as the vector of scattering coefficients, $\mathbf{g}$ in the radar simulations. The image was chosen because it realistically represents what could be seen in a SAR scenario. A picture of higher resolution is not chosen due to the memory and computation constraints.

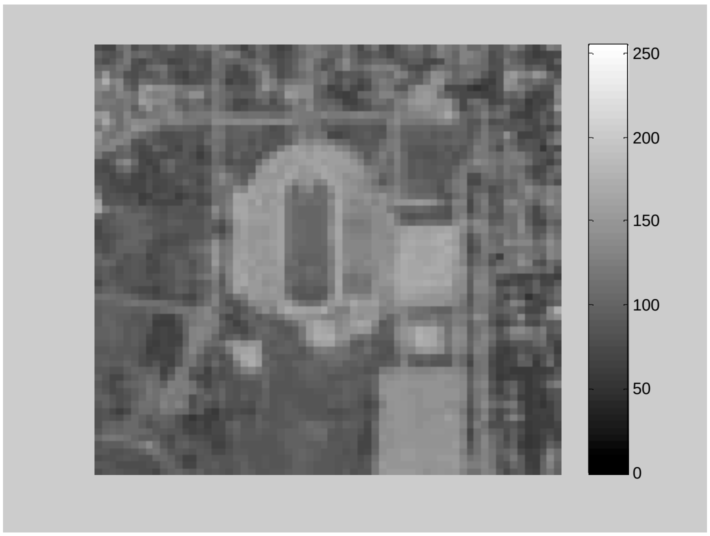

Figure 3.2 Image of the KU football Stadium used as the Input to the Radar Simulator

The time bandwidth product of the received signal was 1600; therefore, only about 0.39 of the original 64 by 64 input can be unambiguously imaged by a single aperture system. The improvement in the estimate of the scattering versus the fraction of measurements used in the KF processing can be viewed in the images given in Figure 3.3. The fraction indicated at the top of each image, is the value of the fraction of measurements processed (e) by the KF to obtain that particular estimate.

It could be also seen from the figures that there is a lot of improvement in the estimate in the initial stages of the processing. This rapid improvement in the estimate in the early stages could be attributed to the reduction of interference between the targets present in the radar measurement. As the clutter associated with the

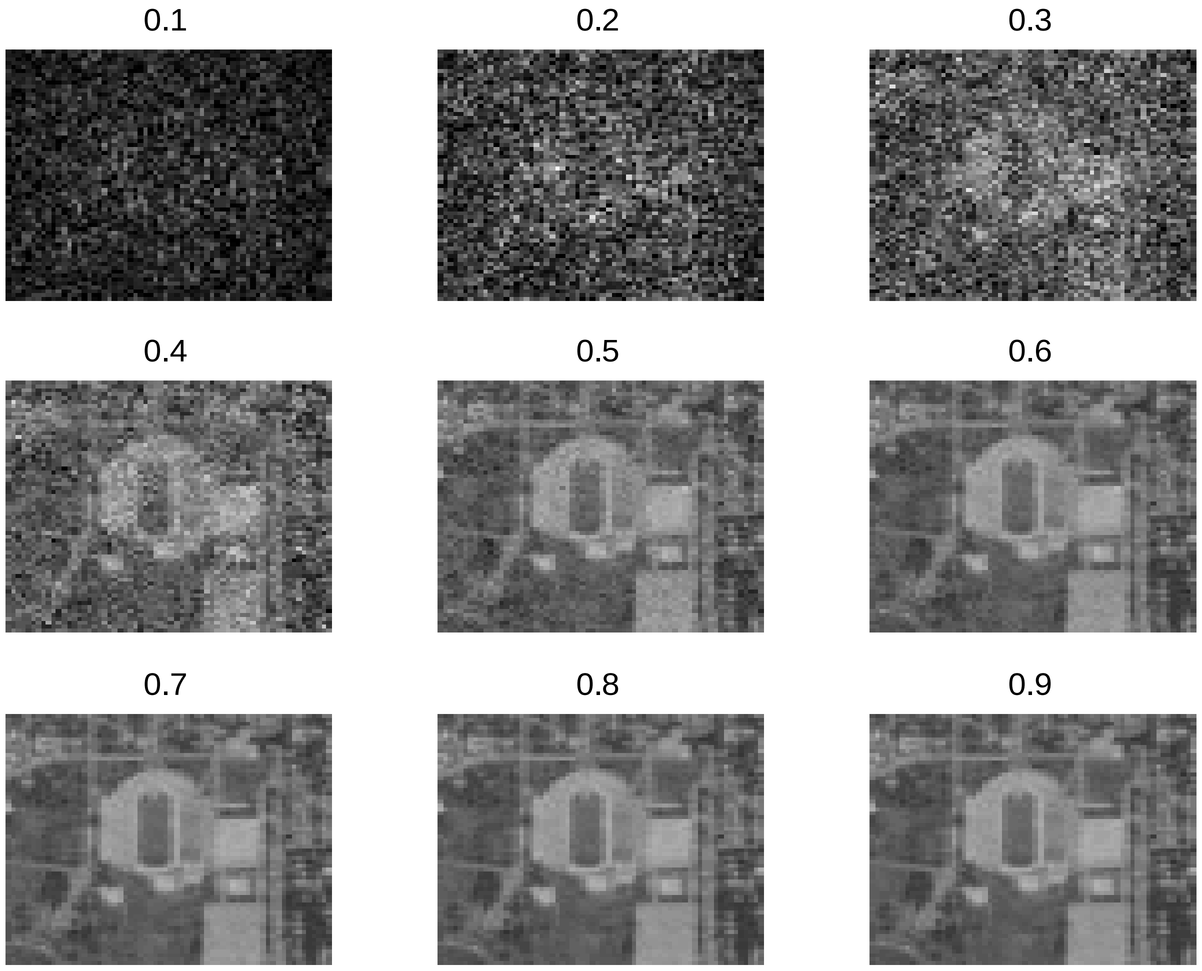

Figure 3.3 KF estimate obtained for each iteration

measurements is eliminated to resolve the range and Doppler ambiguities a huge improvement is made in the estimate as could be seen in the pictures obtained after the initial iterations. It could also be seen that not much of an improvement is made in the last three stage of the processing. The final estimate is as shown in Figure 3.4 and could be seen to be very close to the actual image shown in Figure 3.2.

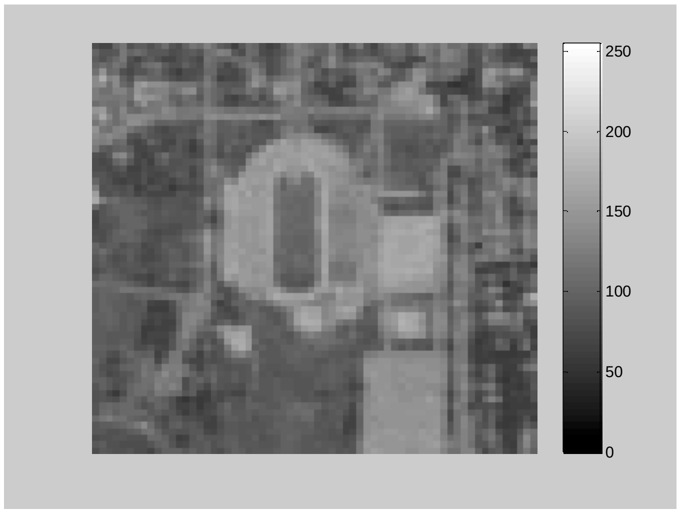

Figure 3.4 KF Result for KU Football stadium Image

3.4 Comparison between Matched, Maximum Likelihood, MMSE and Kalman Filters

A detailed comparison is made between the performance of the developed KF and the three filters, the Matched filter, the ML estimator and MMSE developed in earlier research works. A detailed review of the methodologies of operation for all the three filters is already given in chapter two. The performance of the filters is tested for different system parameters such as SNR and the number of receivers in the constellation. Finally the computational load of all the four filters is compared for different number of receivers in the constellation.

Figure 3.7 shows the results for the four filters when applied to three different SNR cases: low SNR (-10dB), moderate SNR (10dB) and a high SNR (30dB). The target image is as shown in Figure 3.5. It was taken from a smaller photograph of the football stadium of the University of Kansas, in Lawrence, Kansas.

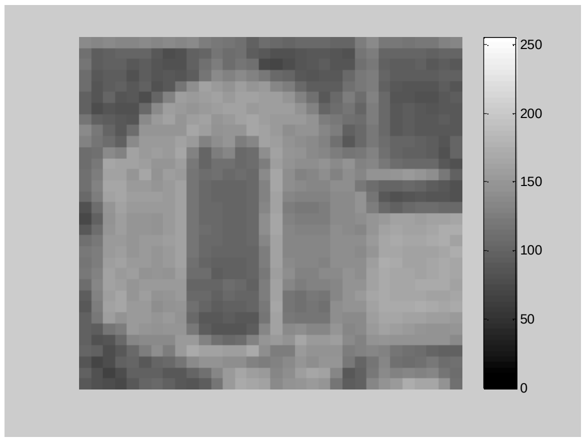

Figure 3.5 A smaller Image of KU Football stadium

The photograph was scanned, changed to a 256-level grayscale image, and cropped to be 32 by 32 pixels. Each pixel intensity was given a random phase, and the resulting set of complex pixel values was used as the vector of scattering coefficients, $\mathbf{g}$ in the radar simulations.

The apertures for Figure 3.5 are in a sparse randomly located 13-element array. The time bandwidth of the signal was 400, and so the total number of measurements taken for 13 receivers is 5200. Before analyzing the performance of the filters let us look at the expected correlation $\mathbf{a}$ between the responses obtained from the illuminated targets. This expected correlation $\mathbf{a}$ gives detailed information about the clutter that could be expected in the measurements and so helps in analyzing the performance of the filters. $\mathbf{a}$ is obtained as follows.

$$\mathbf{a}(i,j) = \frac{\mathbf{r}_i^{\dagger}\mathbf{r}_j}{\left|\mathbf{r}_i^{\dagger}\right|\left|\mathbf{r}_j\right|}. \tag{3.21}$$

where $\mathbf{r}_i$ and $\mathbf{r}_j$ represent the normalized response vectors for target pixels i and j respectively and $|*|$ is defined as the absolute magnitude.

The absolute magnitudes of the individual elements of $\mathbf{a}$ are plotted to obtain the picture shown in figure 3.6. '$\mathbf{a}$' is basically the expected correlation between the

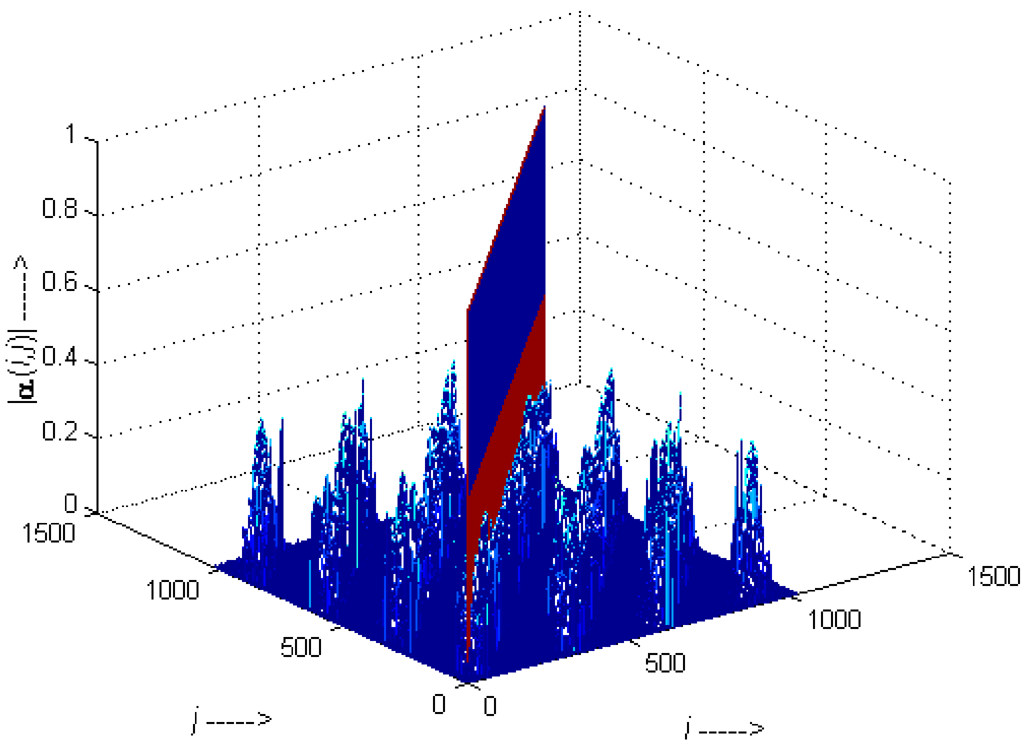

Figure 3.6 A pictorial representation of the Expected Correlation between the responses obtained from the illuminated targets

responses obtained for two unit ground scatterers for the present scenario. The off diagonal elements in $\mathbf{a}$ represent the cross correlation between the responses obtained from the illuminated target and so signify the clutter component present in the normalized response. It could be seen that many of the off diagonal elements have significant values indicating that there is a significant presence of clutter in the measurements obtained for the scenario of sparse arrays being considered.

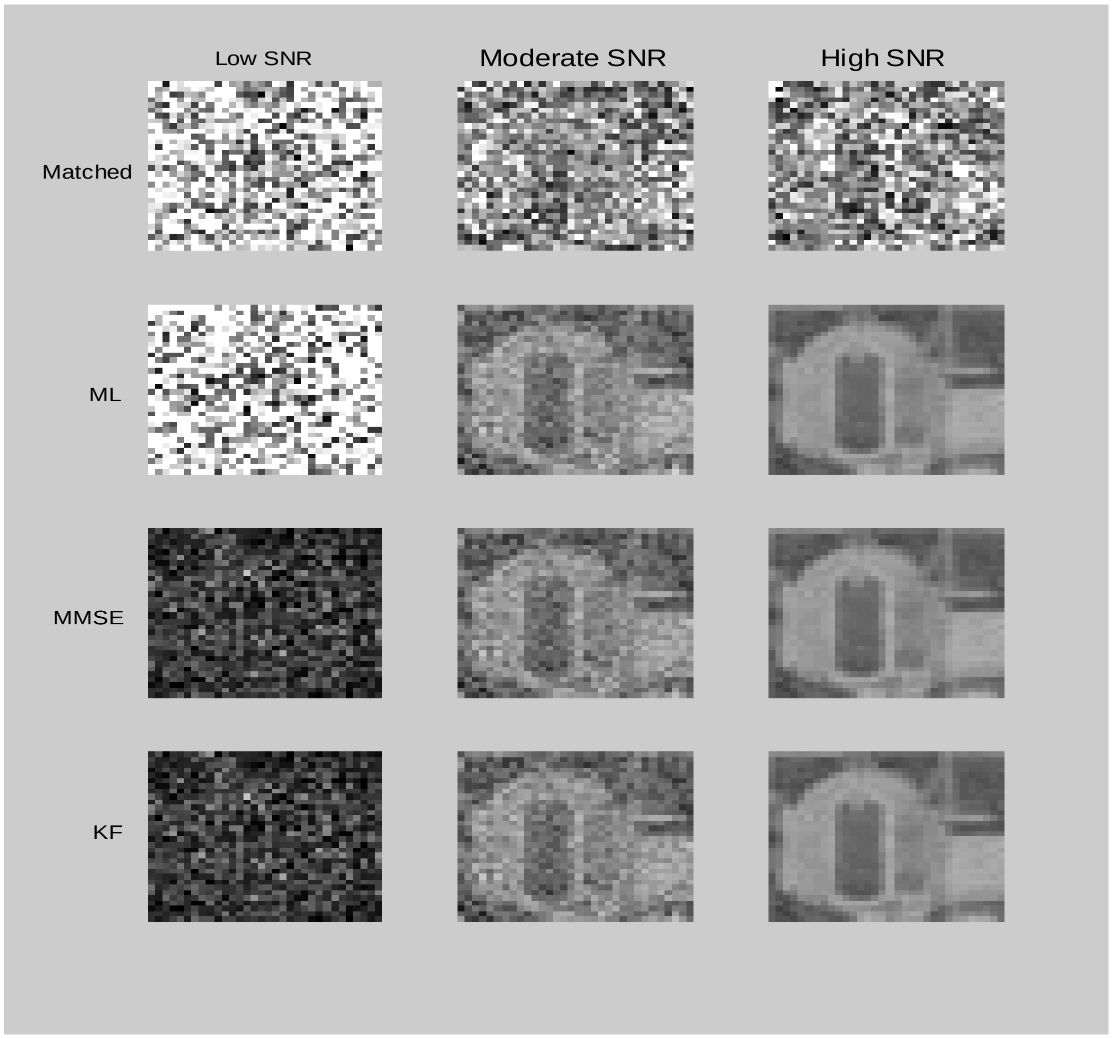

Figure 3.7 Comparison of Matched, ML, MMSE and Kalman filters performance versus SNR

Once information is obtained about the clutter present in the measurements, the filters could be analyzed more thoroughly. It could be seen from Figure 3.7 that, for all the SNR scenarios considered, MMSE and KF give better estimates than the Matched filter. This could be easily understood as the matched filter, though able to maximize the signal to noise ratio is highly clutter limited. Even though the matched filter estimate of the scattering coefficients improves with SNR, it is still unable to make an optimal estimate in any of the cases. This could be attributed to the fact that the matched filter is unable to eliminate the error in the estimate due to clutter. As the clutter present in the measurements for this scenario is significant (as seen in figure 3.6) the matched filter is unable to give good estimates.

It is also seen that the KF gives better estimates than ML for the LOW and Moderate SNR scenarios, while for high SNR all the three, ML, MMSE, and KF seem to give the same results. This is due to the fact that, as explained earlier, the ML filter though being able to minimize signal clutter, is unable to reduce the noise contained in the signal.

The major thing that could be seen from the figure is that the developed KF gives the same estimate as given by MMSE for all the SNRs considered. This could mean that the initial assumption made regarding the process noise correlation matrix $\mathbf{K}_u(0)$ could be wrong, or it might be due to some other reasons. A detailed investigation is done to find out the reasons behind this, the results of which will be presented in the later chapters.

The effect of SNR on performance can be clearly seen in the Figure 3.8. The results presented clearly validate the conclusions stated about the performance of the filters. It could be once again seen that though KF performs better than matched and ML filters it gives the same result as MMSE.

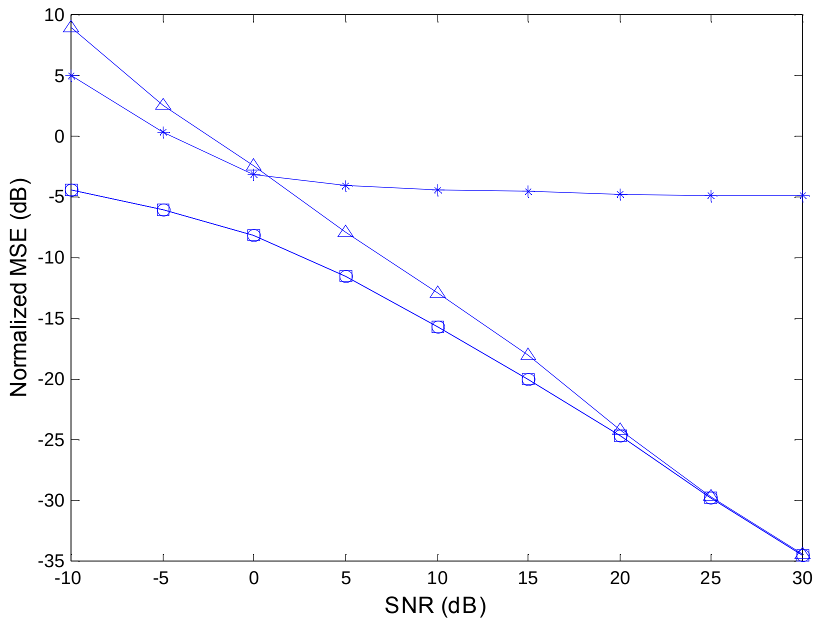

Figure 3.8 Matched, ML, MMSE, and Kalman Filters performance versus Input SNR for 13 receiver sparse random array

The rapid increase in error as SNR decreases for the ML filter and the flattening of the matched filter curve for high SNR could also be seen. In the case of ML, MMSE filters and KF it could be seen that the error continues to decrease as the SNR rises.

As the number of receivers in the constellation is increased, more number of measurements could be obtained, thereby providing more information to the filters for resolving the ambiguities present in the system. It is therefore expected that the estimation error decreases with increase in number of receivers. Figure 3.9 validates this expected result. The estimation error versus number of receivers is relatively flat for the matched filter because it cannot use the additional information being provided to eliminate the clutter. It only uses the additional information to maximize the gain on the target.

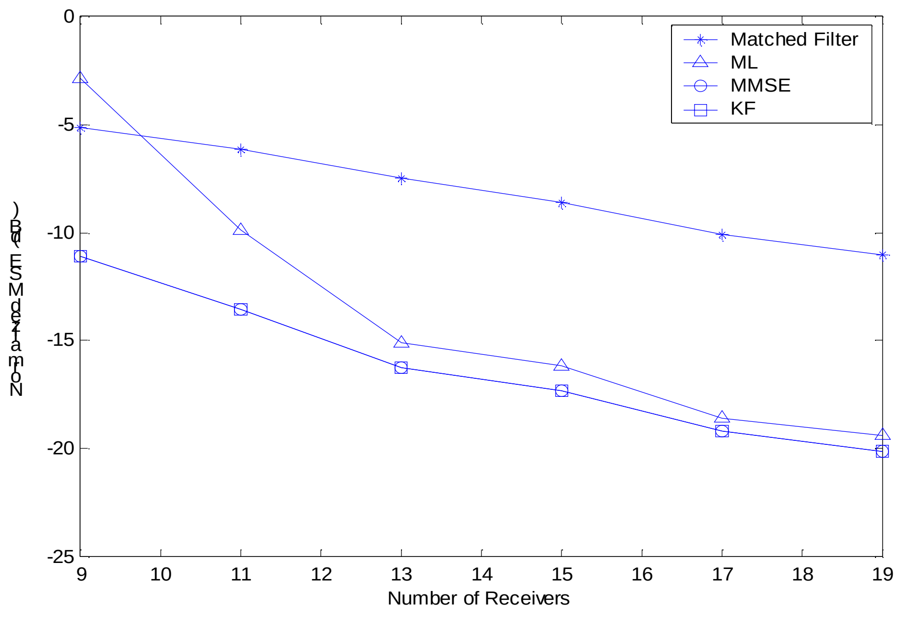

Figure 3.9 Matched, ML, MMSE and Kalman Filters performance versus number of receive apertures for a sparse, random array and moderate SNR

A significant improvement in estimate could be seen in ML, MMSE and KF as the number of receivers are increased from 9 to 13. But there doesn't seem to be much improvement when the number of receivers is varied from 13 to 19. This improvement in the estimate occurs because each additional aperture increases the amount of signal energy collected.

The most important reason behind the development and implementation of KF has been to decrease the processing load inherent in the MMSE filter. The decrease in processing load is achieved by means of reducing the dimension of the huge inverses that need to be calculated in the MMSE scenario. The Figure 3.10 shows this improvement obtained. The processing times for KF for different number of receivers is obtained for a fixed L value of 10, where L is the number of smaller vectors the

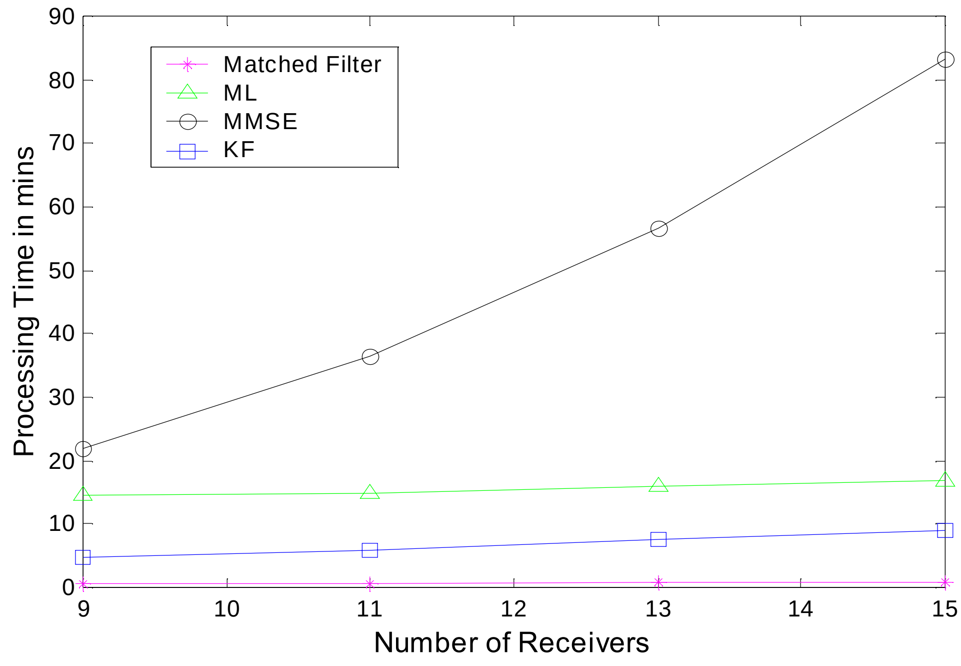

Figure 3.10 Matched, ML, MMSE and Kalman Filters processing speed versus number of receive apertures for a sparse, random array

radar measurement vector is divided into. The processing is done in Matlab on an Intel Pentium III processor offering a processing speed of 697 MHz. The processor is part of a multiprocessor system, which is used to test the parallel version of KF the details of which will be presented in Chapter 6. The results validate the main reason behind the development of the KF. It could be seen that as the number of radar measurements are increased, the processing time for MMSE increases huge fold. But in the case of the KF, there is only a slight change in the processing time. It could also be seen that the processing time for KF is even less than that taken by ML filter. This could be attributed to the fact that the ML filter, like MMSE filter has to compute the inverse (pseudo) of a huge matrix. It could also be seen that of all the filters Matched filter processing is the fastest, but as it doesn't give an optimal estimate it cannot be considered as an option.

We have seen in the previous case that the developed KF takes less processing time than MMSE for a fixed L value of 10. This improvement is basically obtained due to the fact that instead of performing a single iteration involving operations on huge matrices multiple iterations are performed involving operations on smaller matrices. So is there an optimal value of L for which the processing time is least? Figure 3.11 gives an answer to the above question. It shows the processing time for KF versus$L$. The processing times are obtained for 9 apertures, for a total of 3600 (=9*400) measurements. It could be seen from the figure that the time is dependent on$L$ and is actually minimum (=358secs) for aL value of 6.

So the developed KF gives the same estimate as given by MMSE but it greatly reduces the processing time and computational load. Further analysis is made of the mode of operation of KF to optimize its performance by finding ways to obtain an even better estimate, at a much faster speed.

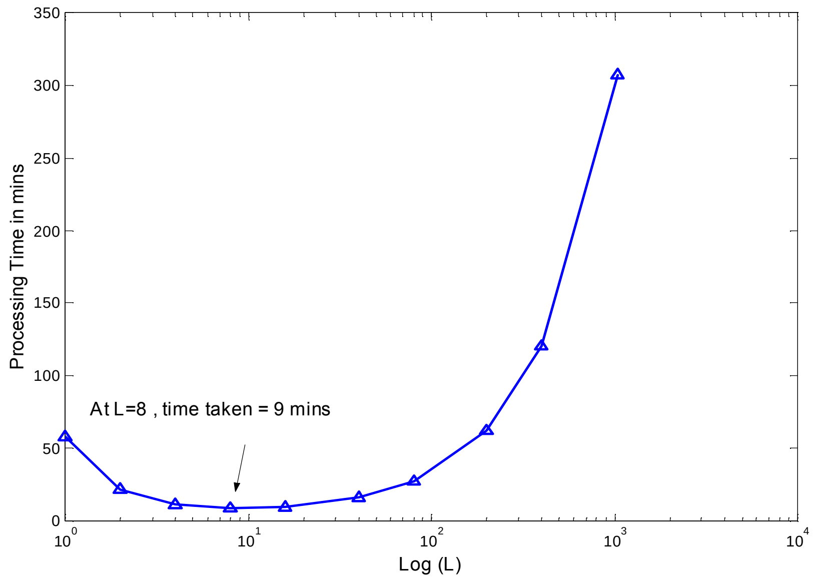

Figure 3.11 Processing Speed of KF versus L

Even though the KF decreases the overall computational load existent in MMSE filter, it still consists of a huge number of computations. The computations pertaining to the operations performed on the target correlation matrix constitute the dominant factor in the overall computation load. Numerous ways are pursued to decrease these computations.

But initially various simulations were performed to analyze the dependence of the performance of KF on the initial values chosen for the error correlation matrix $\mathbf{K}_g(0)$, and the process noise correlation matrix $\mathbf{K}_u(0)$. The simulations results are also analyzed to answer a number of questions. For examples, how does the diagonal and non-diagonal elements of the error correlation matrix vary with iterations? Out of clutter and noise which one plays a major part in the computation of the Kalman Gain? How does the innovation energy (energy contained in $\mathbf{u}(l)$) vary with iterations? In the next chapter a detailed presentation is made of the experiments performed and the results obtained to answer the above questions.

4. PERFORMANCE CHARACTERISTICS OF THE KF

To perform any optimization on the developed KF, it is necessary to gain knowledge about its functioning and performance. In this section, a detailed analysis is made on the performance characteristics of the KF. Experiments were performed to analyze the variation of kalman gain $\mathbf{G}$, error correlation Matrix $\mathbf{K}_g(0)$ and innovation energy, versus the amount of data processed. From this we can obtain detailed knowledge about the functioning of the KF. Experiments are also conducted to quantify the dependence of KF performance on the initial values given for error Correlation Matrix, and process noise correlation matrix $\mathbf{K}_u(0)$.

The following scenario is considered for the experiments presented in this and later chapters. The target image is as shown in Figure 3.5. The photograph was scanned, changed to a 256-level grayscale image, and cropped to be 32 by 32 pixels. Each pixel intensity was given a random phase, and the resulting set of complex pixel values was used as the vector of scattering coefficients $\mathbf{g}$ in the radar simulations. The apertures were placed in a sparse randomly located 13-element array. The time bandwidth of the signal was 400, and so the total number of measurements taken for 13 receivers is 5200. A higher number of apertures than the required minimum are considered so as to obtain a clear picture of the dependence of KF on the various other parameters. The number of iterations or the number of divisions made to the measurement vector is 10 (= L).

4.1 Variation of Kalman Gain, Error Correlation Matrix and Innovation Energy with Processing

Expressions for the kalman gain $\mathbf{G}$, and error correlation matrix $\mathbf{K}_g$ are given earlier in Section 3.2. The initial value given for $\mathbf{K}_g(0)$ is $s_g^2\mathbf{I}$ where s_g^2 is the expected value of the squared reflectance magnitude for each target and $\mathbf{I}$ is an identity matrix. The $\mathbf{K}_u(0)$, which expresses the uncertainty in the state transition matrix is neglected, and the initial values for the scattering coefficients vector $\mathbf{g}(0)$ are taken as zeros. The reasons for the above three initializations were explained in section 3.3.1. The signal to noise ratio is taken as 30dB.

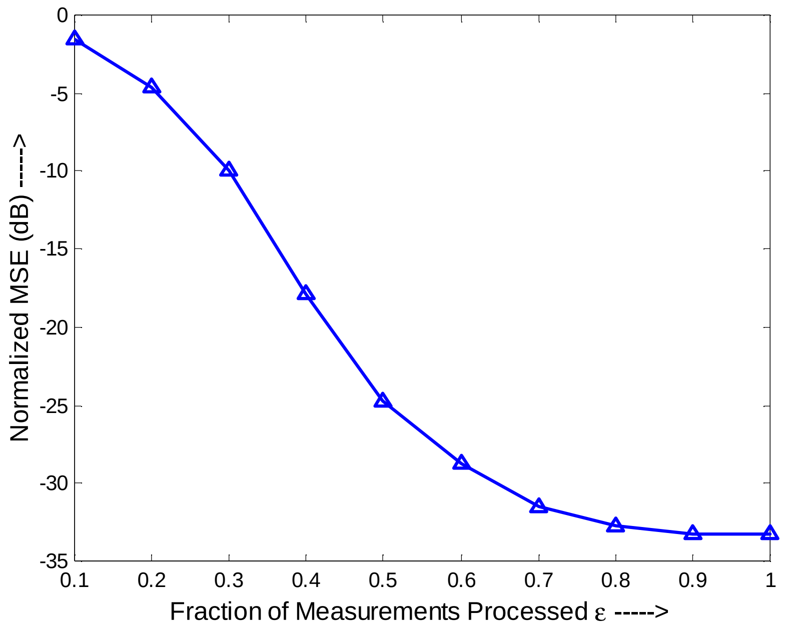

Figure 4.1 Normalized MSE (dB) vs. the fraction of Measurements Processed (e) for the smaller version of the KU Football Stadium Image

The expected correlation between the responses of the illuminated targets for the present scenario was shown earlier in Figure 3.6. Figure 4.1 shows the variation of the Normalized MSE (in dB) vs. the fraction of measurement vector (e) processed. The variation of the MSE with processing is discussed while explaining about the variations of the other matrices in the KF.

Now, the initial value of the error correlation matrix $\mathbf{K}_g(0)$ was taken as an identity matrix with the diagonal elements scaled to the expected correlation between the scattering coefficients. It could be expected that due to the significant presence of clutter in the measurements, the non-diagonal elements are not going to remain

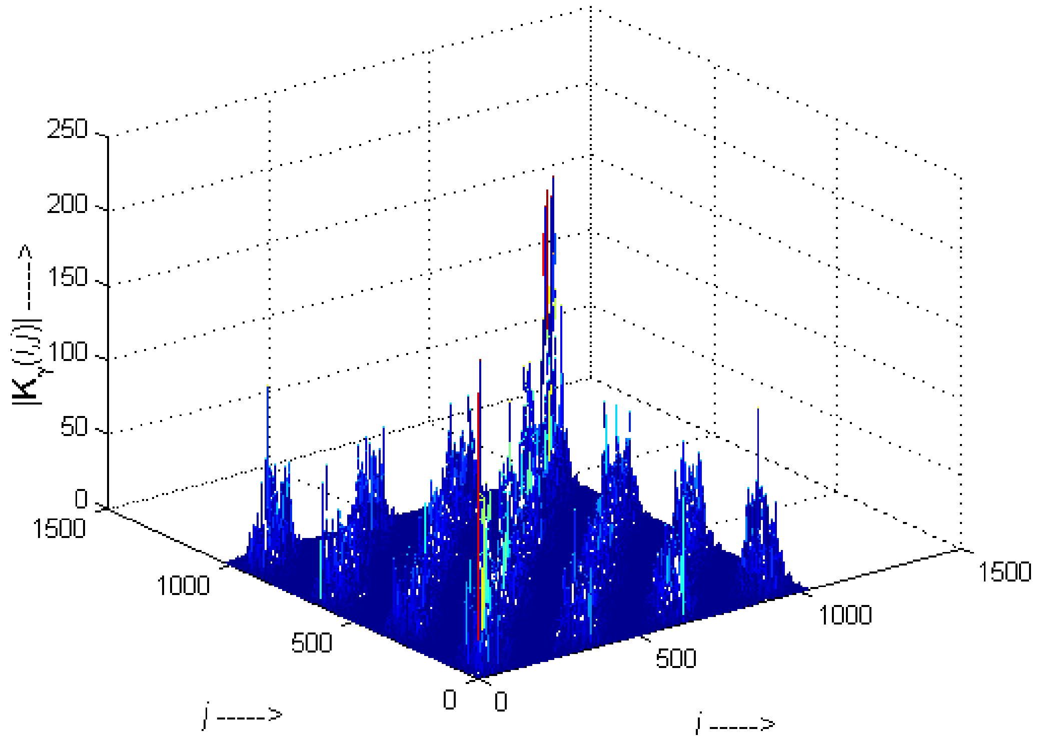

Figure 4.2 A pictorial representation of the Error Correlation Matrix after processing of half the measurement data

negligible. So how do the diagonal and non-diagonal elements vary with processing? Or in other words how is the clutter scenario present in the system, reflected in $\mathbf{K}_g(l)$? To get answers to some of the questions $\mathbf{K}_g(l)$ is plotted for $l = 5$ out of a total of 10 (L) iterations (after processing of 50% of the data) and is shown in Figure 4.2.

It could be seen that along with the diagonal elements there is a significant presence of cross diagonal elements in the updated error correlation matrix. This demonstrates that there is significant presence of interference or clutter in the obtained radar measurements. The error correlation matrix basically adjusts itself to the clutter scenario present in the measurements. A more detailed analysis of the variation of the diagonal elements and non-diagonal elements is presented in Figure 4.3.

The variation in magnitudes of the diagonal and non-diagonal elements of the error correlation matrix is represented in terms of their respective RMS (root mean square) values. The RMS values of the diagonal elements and non-diagonal elements of $\mathbf{K}_g(l)$ are obtained as follows

$$\left(K_g(l)\right)_{DRMS} = \sqrt{\frac{\sum_{i=1}^{M} \left(\mathbf{K}_g(i,i)\right)^2}{M}} \tag{4.1}$$

where the subscript DRMS represents the RMS value of the diagonal elements of a matrix

$$\left(K_g(l)\right)_{NDRMS} = \sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1,j\neq i}^{M}\left(\mathbf{K}_g(i,j)\right)^2}{M}} \tag{4.2}$$

where the subscript NDRMS stands for the RMS of the non-diagonal elements of a matrix.

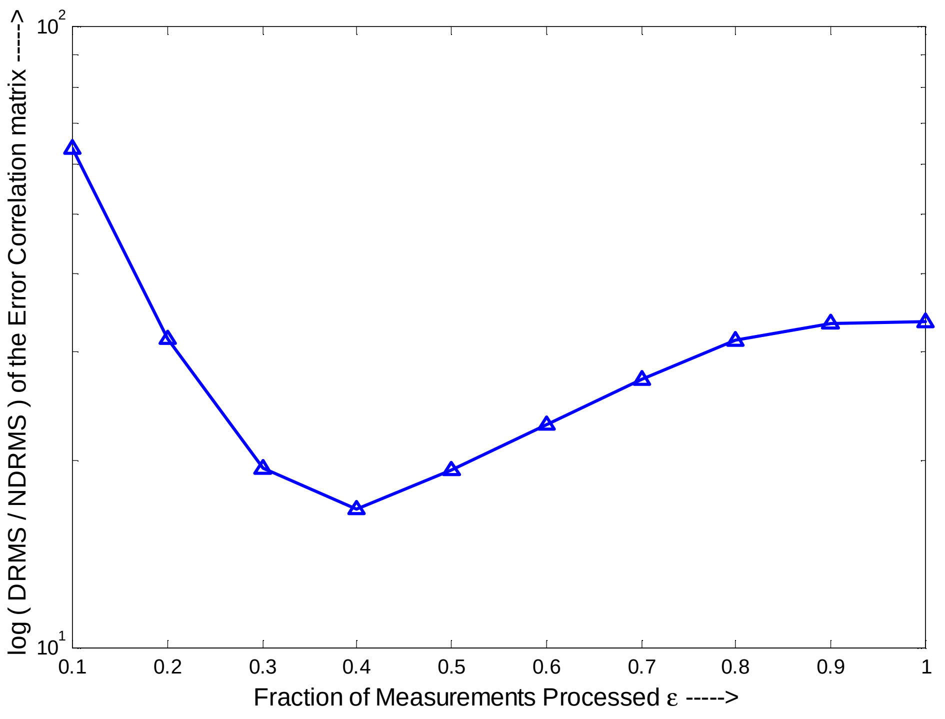

Figure 4.3 Shows the Variation of the ratio between the RMS values of the diagonal and non-diagonal elements of the error correlation matrix

Figure 4.3 shows the variation of the ratio between the RMS values of the diagonal elements and non-diagonal elements $\left(\left(K_g(l)\right)_{DRMS}/\left(K_g(l)\right)_{NDRMS}\right)$. The error in the estimate decreases with processing of new measurements and so along with that it could be expected that all the elements in the error correlation matrix decrease with processing. It could be seen that the cross diagonal elements too play a significant part in the overall processing or in other words even though the RMS value of the non-diagonal elements is very less when compared to the that of the diagonal elements they are huge in number and so still contribute to the estimation process. Almost throughout the processing the ratio between the DRMS and NDRMS of the error correlation matrix varies adjusting to the clutter (cross correlation components) present in the obtained measurements. But it could be seen that during the last stages of the processing there is not much of a change in the ratio as the filter is able to adjust to the clutter actual clutter scenario.

There is an interesting fact to observe about the way in which the Kalman gain $\mathbf{G}(l)$ is updated during each iteration. We know that the Kalman Gain $\mathbf{G}$ is updated so as to minimize the mean square error of the estimate and is obtained by (3.14)

$$\mathbf{G}(l) = \mathbf{K}_g\left(l/l-1\right)\mathbf{P}(l)^{\mathbf{H}}\left[\mathbf{P}(l)\mathbf{K}_g\left(l/l-1\right)\mathbf{P}(l)^{\mathbf{H}} + \mathbf{K}_n(l)\right]^{-1}$$

Now as the magnitudes of the normalized response vectors are very low so the magnitude of elements in the Kalman gain matrix depend significantly on the inverse term. Lets say $\mathbf{b}$ represents the resultant matrix of the product $\mathbf{P}(l)\mathbf{K}_g\left(l/l-1\right)\mathbf{P}(l)^{\mathbf{H}}$.

Then $\mathbf{b}$ basically represents the clutter while $\mathbf{K}_n(l)$ represents the variance of the noise. We have also seen that the diagonal elements of the error correlation matrix are much larger than the don-diagonal elements. Then the ratio

$$R_1 = \frac{\left(\sum_{i=1}^{M} \left|b\left(i,i\right)\right| \Big/ M\right)}{VarianceofNoise} \qquad (4.3)$$

tells us whether the clutter term $\mathbf{b}$ or the noise term $\mathbf{K}_n$ plays a major part in computation of the Kalman Gain. Figure 4.4 shows the variation of R_I obtained for each iteration. It could be seen that for the last three iterations the value of R_I is in the orders of 10^{-1} to 10^{-3} which means that the noise term represented by $\mathbf{K}_n$ dominates and plays the major role in computation of $\mathbf{G}$.

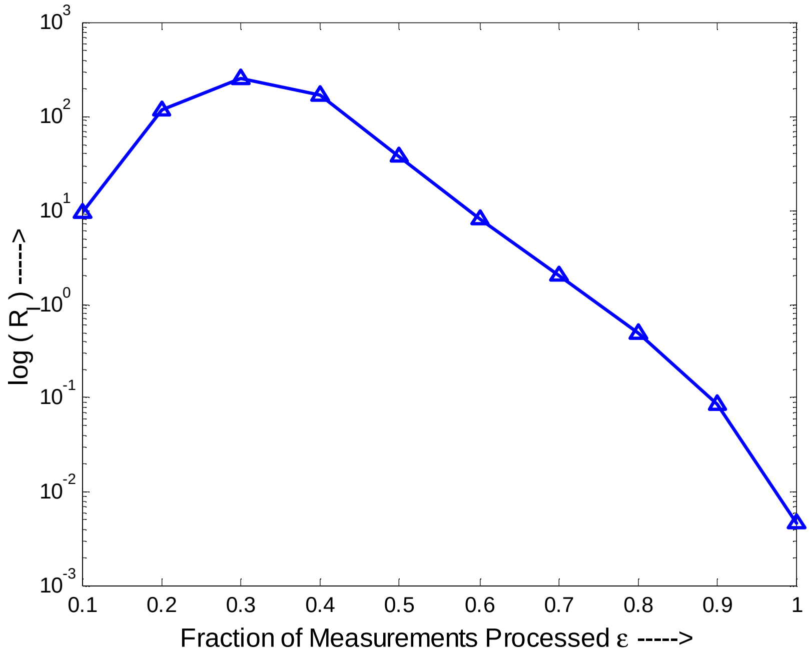

Figure 4.4 The ratio between the absolute mean of the diagonal elements of $\mathbf{b}$ to that of the variance of noise vs. the fraction of measurements processed e

So, basically at the initial stages of the processing the KF mainly eliminates the error due to clutter while at the final stages it mainly nulls the guassian noise present in the system. That is the reason why there is not much of a variation in the MSE at the end of the processing as the KF basically works or eliminating the small error present in the estimate due to noise. So we have seen that the Kalman Gain $\mathbf{G}(l)$

computed during the last stages of the processing doesn't depend on the error correlation matrix. Will this information be useful to reduce the number of computations in the error correlation matrix? We will know about that in chapter 5.

A novel feature in Kalman Filtering is the concept of innovation. Innovation tells about the error between the expected measurement, which could be obtained from the previous estimate, and the observed measurement. It basically gives an idea about the new information present in the latest measurement data. The normalized value of innovation energy present in the new measurement can be computed as follows

$$y = \frac{\left|\mathbf{u}(l)\mathbf{u}(l)^{\dagger}\right|}{\left|\mathbf{r}(l)\mathbf{r}(l)^{\dagger}\right|} \tag{4.4}$$

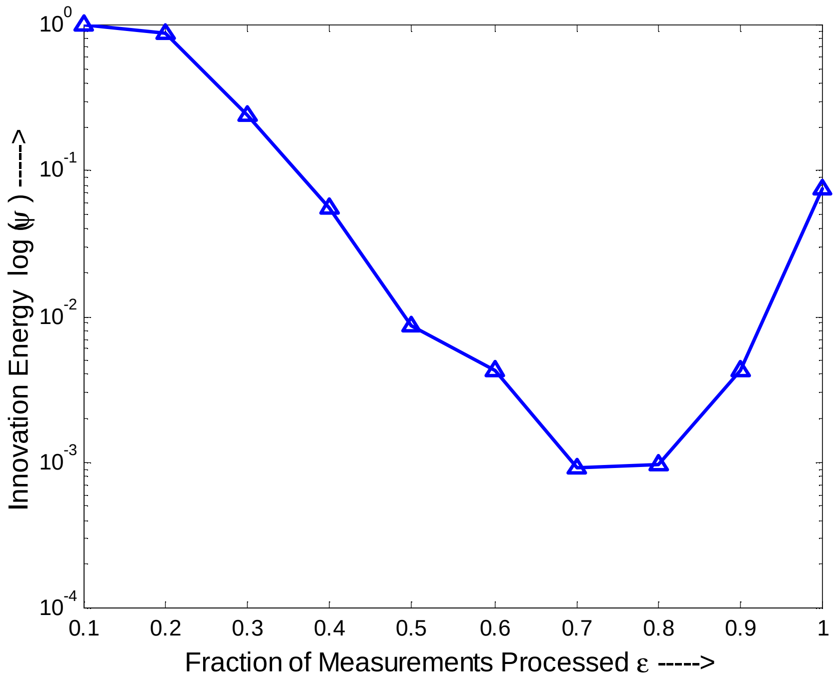

Figure 4.5 Innovation energy vs. the fraction of measurement data processed

It could be seen from Figure 4.5 that the variation in innovation energy is more of a random change depending on the obtained new measurement. For example measurements obtained from one receiver might be more informative than some other receiver depending on their position in the constellation. Moreover some measurements can be more noise prone than some others depending on the time of observation.

As explained a lot of information is obtained about the variation of various parameters in the KF from the above simulations. This knowledge helped to develop strategies for optimization of the KF to still reduce the processing load. The developed strategies, their implementation and performance will be presented in the next chapter.

In all the experimental scenarios considered until now the initial values for $\mathbf{K}_g$, $\mathbf{K}_u$, and $\mathbf{g}$ have been kept constant. So how far are these initial assumptions optimal? How will variation in the initial conditions affect the performance of the KF? Will the KF be able to converge to a better estimate if the final estimate and the final error correlation matrix obtained after processing the measurement data are taken as the initial conditions and the whole processing is redone over the same measurement data? The answers to the above questions are presented in the next section. Basically this analysis is performed to quantify the dependence of the KF performance on the initial conditions and if possible find more optimal initial values for these parameters so as to obtain a better estimate.

4.2 Dependence of the final estimate on the Initial Conditions

In this section different set of values are given to $\mathbf{K}_g(0)$, $\mathbf{K}_u(0)$, and L as part of initiating the KF and its performance is analyzed. It is often observed that for many estimation problems the accuracy of the estimate increases with L. The reason for this increase in accuracy with L is due to the fact that as the number of iterations are increased the KF is able to accommodate the variation in the signal (to be estimated) during the observation time (modeled by state transition matrix) and the process noise ($\mathbf{u}$) more clearly, thereby giving a better estimate. But, as observed in the section 3.4, the developed KF is giving the same estimate as given by MMSE. Is this because L taken is too small or is it due to any of the earlier assumptions made while implementing the KF. The next section gives details of the experiment performed to answer the above questions.

4.2.1 Significance of the Number of divisions, L chosen

Figure 4.6 shows the performance of the KF for different values of L. The number of aperture used is 9. It could be seen that the obtained estimate is not dependent on L. Not only is the final estimate equal for all L, but it could be seen that the rate of convergence of the estimate per iteration is also same. So there isn't any improvement obtained as the L value is increased. This might be due to the initial assumption made regarding the state transition matrix $\mathbf{A}(l)$ and the process noise correlation matrix $\mathbf{K}_u$. Both the above matrices are neglected basing on the on the assumptions that the scattering obtained from each target for different space, time and frequencies remains constant. This assumption is not right as in a real scenario the scattering from each target is not constant for different space, time and frequencies. In reality the scattering varies rapidly with frequency and is very much different for different frequencies. But this assumption is necessitated basing on the fact that the radar simulator [7] developed earlier assumes that the scattering remains constant.

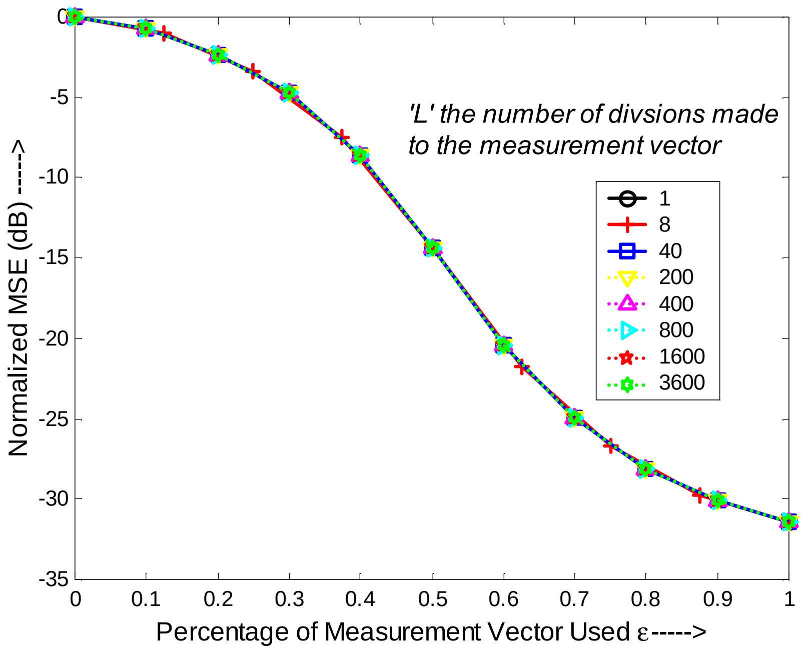

Figure 4.6 Normalized MSE versus the fraction of measurement data processed for different L

It will be really laborious and difficult to incorporate this real variation scenario in scattering in the radar simulator. Moreover as the main aim of the thesis investigation is to develop algorithms, which decrease the processing, and computation load inherent in SAR processing, more effort was put into decreasing this computational load than to improve the already good estimation results being provided by MMSE. But in a real scenario the developed KF is bound to provide better results than MMSE and the accuracy should increase with L [9].

4.2.2 Significance of the Initial value chosen for Error Correlation Matrix

So how robust and stable is the developed KF? Is it not going to converge to an optimal estimate if it is not initiated properly? How do the initial parameters chosen for $\mathbf{K}_g(0)$ affect the final estimate and the rate of convergence of the filter? Experiments are done to analyze the significance of $\mathbf{K}_g(0)$ for optimal performance of the filter. These experiments and the final conclusions made are presented in this section. The basic reason being that by analyzing this dependence of the performance of KF on $\mathbf{K}_g(0)$, it will be helpful to choose a particular value for $\mathbf{K}_g(0)$ which increases the accuracy and the rate of convergence.

4.2.2.1 Diagonal matrix

It is still assumed that the reflectance values from pixel to pixel are uncorrelated and so $\mathbf{K}_g(0)$ is taken as an identity matrix. The diagonal elements are then scaled to different random values (s^2). So, basically a random value is taken for the expected value of the squared reflectance magnitude for each target. The initial value for $\mathbf{K}_u(l)$ is taken as zero as in the previous scenarios. The target scenario is same as the one in section 4.1 and the number of apertures considered is 13.

Figure 4.4 shows the variation of normalized MSE versus the fraction of measurement data processed for different values of s^2. The actual expected value of the squared reflectance magnitude for each target, s_g^2 is 129.81^2. It could be seen that for values of s^2 which are closer to the average value, the rate of convergence is high and for those farther from this average value have lower convergence rates. It could also be seen that if the chosen s^2 is far too smaller than the average value than the KF is unable to converge at all. While for large s^2 values it could be seen that even though the filter goes haphazard at the initial stages, it is still able to converge at the same estimate as that obtained for a s^2 value of s_g^2.

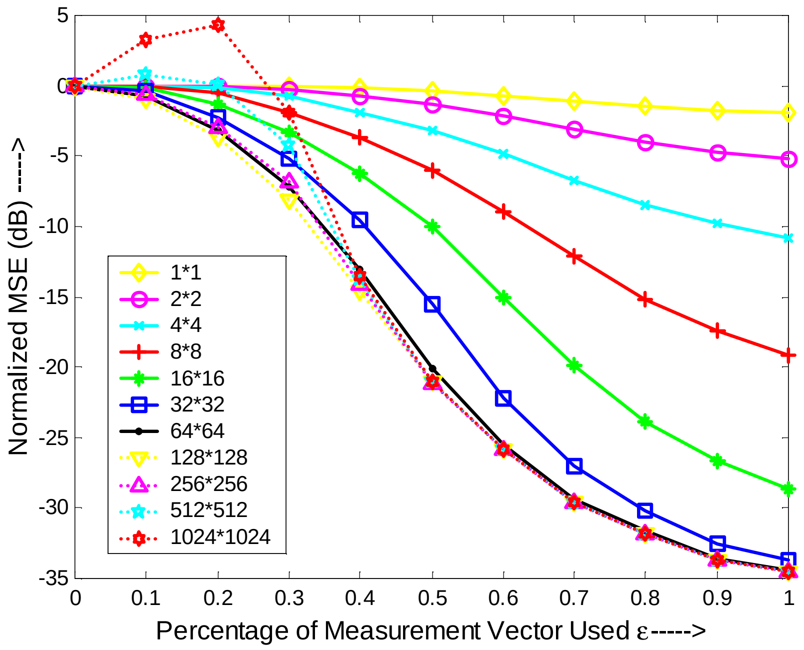

Figure 4.7 Variation of Normalized MSE versus the fraction of measurement data processed for processing, for different s^2 values for diagonal matrix $\mathbf{K}_g(0)$

Two important criteria could be deduced from the performed experiment regarding the initialization of the error correlation matrix. The number one criterion is that the error correlation matrix has to be initialized so as to comprise the whole uncertainty region of the estimated parameter. If the error correlation matrix is initialized to, too small uncertainty region (lesser than the region comprising of the expected value of the squared reflectance magnitude) then the KF is unable to converge. The second one is that the rate of convergence of the KF depends on how well the boundaries of the uncertainty region comprising the estimated parameter are defined.

4.2.2.2 Random Correlation Matrix

This experiment is really going to test the stability and robustness of the developed KF. $\mathbf{K}_g(0)$ is taken as a random correlation matrix. A correlation matrix $\mathbf{d}$ is obtained as shown below. Initially a random complex vector $\mathbf{h}$ of M (number of resolution cells) elements is generated as follows

$$\mathbf{h}(i) = V + jV \tag{4.5}$$

where V is random number between 0 and 1. Then

$$\mathbf{d} = \mathbf{h}^{\mathrm{H}}\mathbf{h} \tag{4.6}$$

Now, the performance of the KF is tested for three different cases of $\mathbf{K}_g(0)$.

Case I: The initial error correlation matrix is taken as $\mathbf{K}_g(0)$, where

$$\mathbf{K}_g(i,j) = V\sigma_g^2\mathbf{d}(i,j). \tag{4.7}$$

where V is random number between 0 and 1, s_g^2 is the expected value of the squared reflectance magnitude for each target, and $\mathbf{K}_g(i,j)$ represents the element at ith row and jth column.

So basically the obtained matrix $\mathbf{K}_g(0)$ is a random correlation matrix and no assumptions are made about the correlation between the reflectance values from pixel

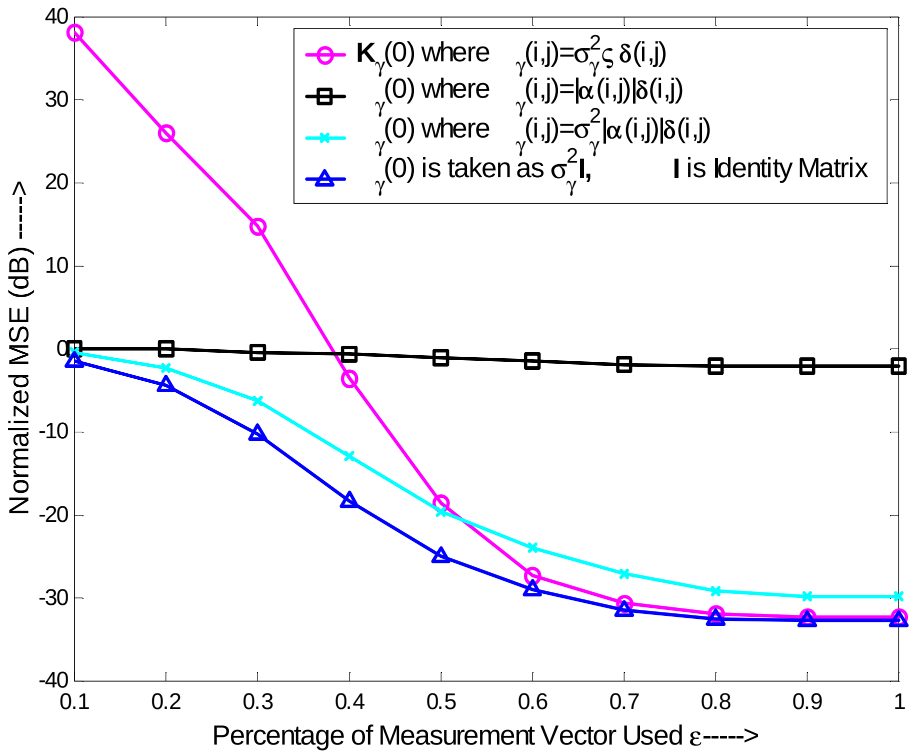

Figure 4.8 Variation of Normalized MSE versus the fraction of measurement data processed for processing for different $\mathbf{K}_g(0)$

to pixel. It could be seen from Figure 4.8 that though the filter goes astray initially it later on converges to an optimal estimate. This further proves the point made in the

the last section that the error correlation matrix has to be initialized so as to comprise the whole uncertainty region comprising the reflectance from pixels. The performance of the KF for $\mathbf{K}_g(0) = s_g^2\mathbf{I}$ is given for comparison purposes.

Case II: The initial error correlation matrix is taken as $\mathbf{K}_g(0)$, where

$$\mathbf{K}_g(i,j) = \left|\mathbf{a}(i,j)\right| \mathbf{d}(i,j) \tag{4.8}$$

where $\mathbf{a}$ is the expected correlation between the normalized responses obtained from the illuminated targets given in (3.21).

Basically this initialization of the $\mathbf{K}_g(0)$ is done to seen how the filter performs if it is assumed that the correlation between the reflectance values obtained from pixel to pixel is taken to be similar to the expected correlation between the responses obtained from the targets $\mathbf{a}$. It could be seen from the figure that the KF is unable to converge for this initialization. This is because $\mathbf{K}_g(0)$ is initialized to a very small region of uncertainty, which doesn't include the estimate.

Case III: The initial error correlation matrix is taken as $\mathbf{K}_g(0)$, where

$$\mathbf{K}_g(i,j) = s_g^2 \left|\mathbf{a}(i,j)\right| \mathbf{d}(i,j) \tag{4.9}$$

It could be seen that the basic difference between (4.8) and (4.9) is that in the second case the error correlation matrix $\mathbf{K}_g$ is initiated to a larger region of uncertainty. It could be seen from the figure that the performance of the filter improves rapidly with changes made. It could also be seen that the final estimate obtained for Case III is actually less accurate than the one obtained for Case I. In Case I no sort of assumptions are made regarding the correlation scenario of the reflectance values from pixel to pixel. While in case III the cross correlation components of the reflectance values from pixel to pixel are assumed to be less. But still estimate for case I is more accurate than Case I. This might be due to the factor that there are ample numbers of measurements for the KF for Case I to converge. Had there been like 50% of measurements than Case III would perform better than Case I. The initial value for $\mathbf{K}_g(0)$ in Case III might be a good one in a real scenario as it takes into consideration the expected correlation between targets.

4.2.3 Perfect Initial Conditions

We have seen in section 4.1 that though the innovation energy decreases gradually with processing it is still has random characteristics and depends on the new measurement obtained. As the filter processes measurements more and more information is obtained and according to that the accuracy of the estimate is improved. Now is the KF able to extract all the available information contained in a measurement sample or is some information left out? So will the KF be able to extract this lost information (supposing some information is lost)? To obtain answers for the above question an experiment was performed, the details of which are as follows.

Initially KF is done on the obtained measurements with the initial error correlation matrix taken as $\mathbf{K}_g(0) = s_g^2\mathbf{I}$ and the mean scattering value as zero. The filtering process is continued until all the measurements are used to converge to a final estimate. Then the final error correlation matrix and the final estimate obtained at the end of the processing are saved. These saved values are taken as the initial conditions and the KF is redone on the same measurements.

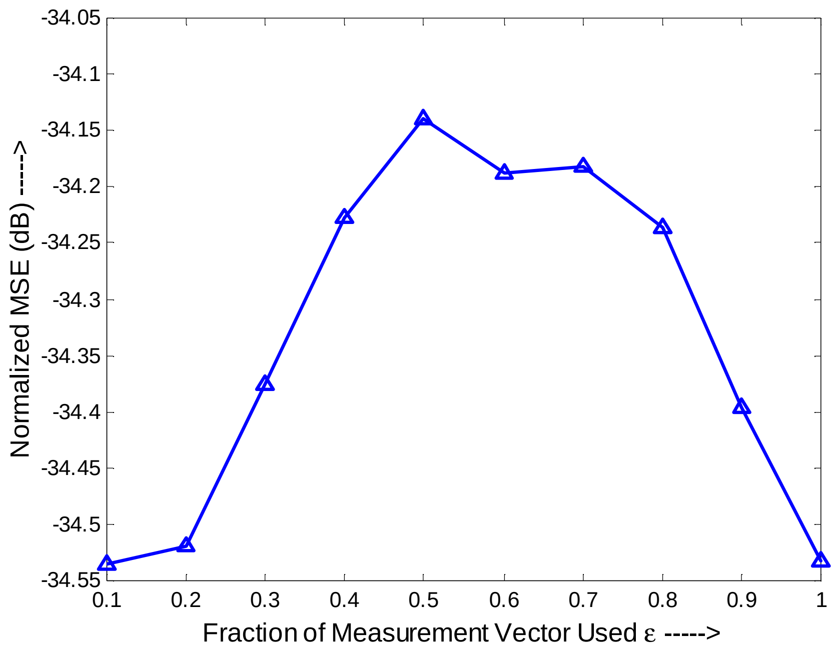

Figure 4.9 Variation of Normalized MSE versus the fraction of measurement data processed for a case where the final values obtained for $\mathbf{g}$ and K_g are taken as the initial values and the processing is done again.

From Figure 4.9 it could be seen that the KF is unable to converge to a better estimate. This shows that the KF has already extracted all the information that is available in the measurements and there is no more innovation energy left in the measurement vectors that could be used to converge to a better estimate.

From these experiments, a lot of information is gained about the working, performance and stability of the developed KF. The first thing that is learnt is that initially the clutter present in the system dominates and the Kalman Gain is computed so as to minimize the error in the estimate due to this clutter. In the final stages the measurement noise dominates and the Kalman gain is updated to reduce the error in the estimate due to this noise. Information is also obtained about the variation of target correlation matrix, and innovation energy with processing. It will be shown in the later chapters how this information has been useful in optimizing the developed KF. It was also seen that the developed Kalman filter is very stable and the final estimate is not dependent on the initial values set for the error correlation matrix if the uncertainty region of the estimate is properly represented in the initial error correlation matrix.

The KF implementation is fully done on matrices and so it is expected that matrix operations consume a major portion of the overall processing time. So, it becomes important initially to quantify the processing times consumed by each of these operations in the KF. This will help in developing optimization techniques tailored to the specific application.

5. OPTIMIZATION OF DEVELOPED KALMAN FILTER

5.1 Optimization of Matrix Operations

The KF operation consists of a number of matrix operations, for that matter, all the operations performed are related to matrices. So if the total processing time has to be decreased then the order in which these operations are performed has to be perfectly optimized. From Figure 3.11 it is also known that the overall processing time depends on the number of divisions made to the measurement vector (L) or the number of iterations performed.

The Traditional measure of the efficiency of a numerical algorithm is based on a *flop count*. *Flop* is an abbreviation for "floating point operation". The flop count is simply the number of flops that the algorithm would execute when used to solve a particular problem. Flops count are usually given in terms of the problem "size", in the present case, it is the size of the matrices. The numbers of flops required for multiplication and inverse are a lot higher than that for addition, subtraction, and transpose and so while measuring, the numbers of flops for the latter operations are not considered.

The number of operations required for calculating the inverse of a square matrix of order P is P^3. Some algorithms have been developed earlier, which implement matrix inverse in lesser number of operations. For example Winograd and Strassen developed algorithms for implementation of matrix inverse in lesser number of operations. But these algorithms are difficult to implement and lots of extra effort has to be put to develop parallel versions for them.

The number of operations required for multiplying two matrices (of order R by S and S by T), are $R \times S \times T$ multiplications and $R \times T \times (S - 1)$ additions. As explained earlier the addition operations will be discarded.

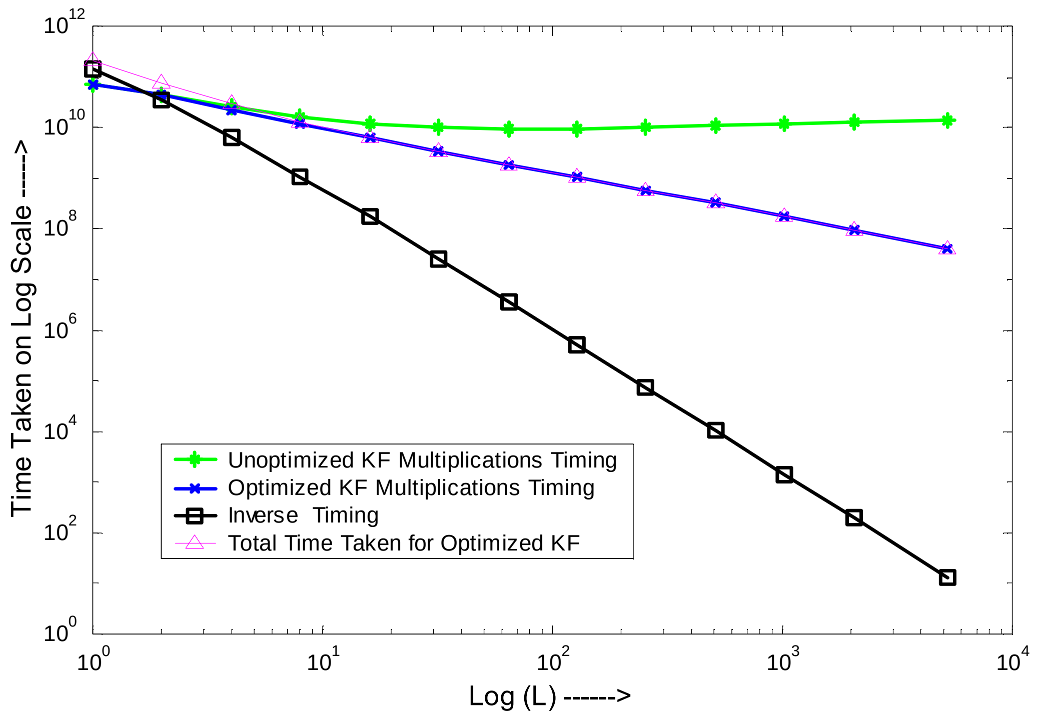

Figure 5.1 Required number of Significant Operations versus L

Now basing on the above facts, the order of matrix operations in the KF implementation are chosen in a particular way so as to reduce the overall number of flops required for processing. The total number of operations required versus *L,* for the KF Implementation for a resolution of 1024 (32 by 32) and for 5200 (13 by 400) measurements obtained are as shown in Figure 5.1. It shows that the number of required operations decreases with L, or its states that the processing time decreases with L. But in real time software implementation the variation in processing time will be as shown in Figure 5.2. It shows that the processing time decreases initially as L

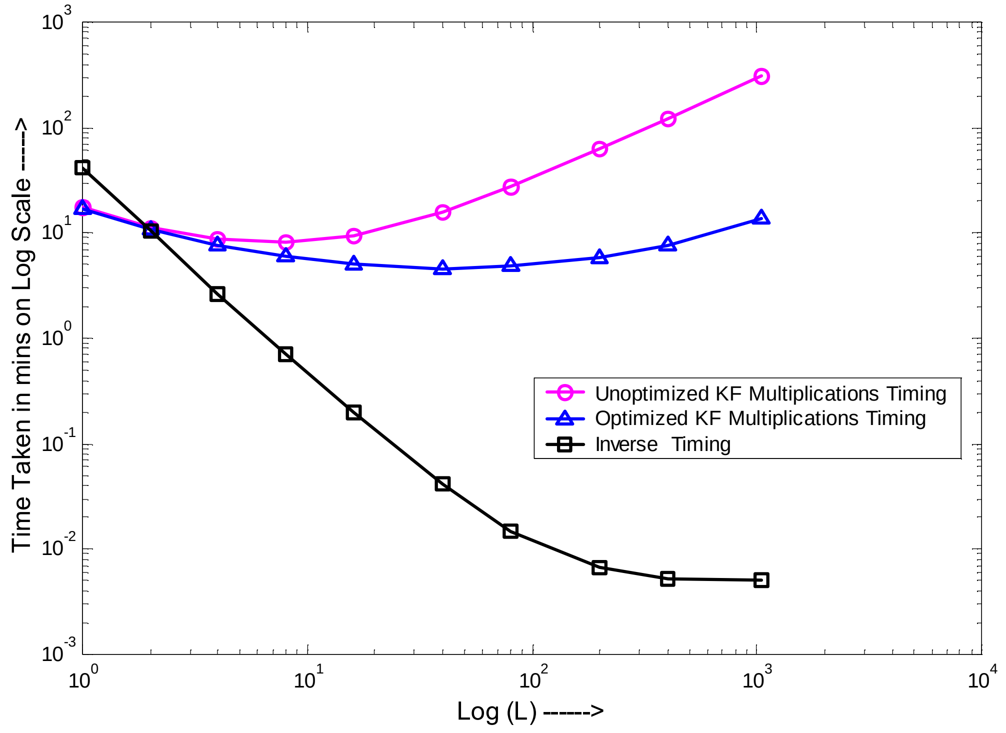

Figure 5.2 Processing time taken versus L for KF Implementation in Matlab

is increased but later on keeps on increasing. This is due to the fact that apart from the processing time taken for executing the operations, a lot of time is also taken for calling on subroutines and accessing memory. As L is increased the number of subroutine calls increases and the time taken for memory access operations also increases. If high speed processors are used then it will be better to reduce the number of calls made to the subroutines and opt for a point where the inverse times and multiplications times intersect (for an *L* value of 2 in the Figure 5.2). It should be noted that as the required resolution increases there will be a more significant change in the processing times as *L* increase.

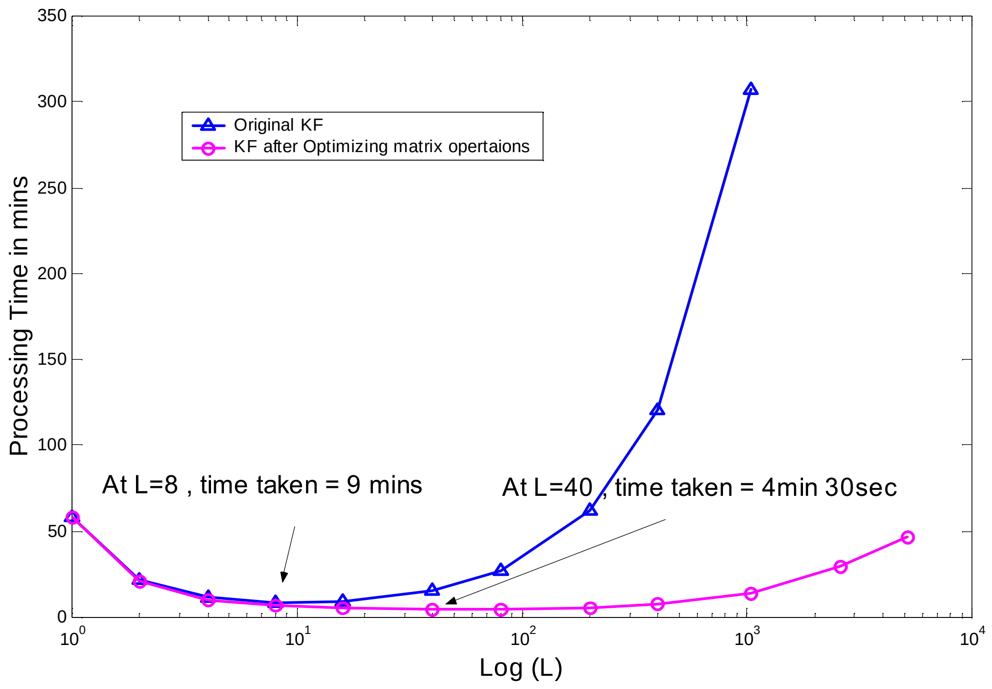

Figure 5.3 Total Processing time taken versus L for KF Implementation in Matlab

It also shows that the time taken for matrix multiplications dominates the overall processing time. So if the matrix multiplication operations are speeded up then the overall processing time also decreases. Work has been done to implement this and will be presented in later chapters. Figure 5.3 shows the improvement made in processing efficiency and also shows that the least time is taken when *L* =40, as compared to *L* =8 for the earlier implementation. So now it will be possible to obtain faster estimates while inverting smaller matrices, thereby reducing further the computational problems involved with inverting huge matrices.

It could also be seen that the time taken for higher L is far less for the optimized KF when compared to the un-optimized KF. Figure 5.4 shows the comparison between times taken for optimized KF versus un-optimized KF versus the number of receivers.

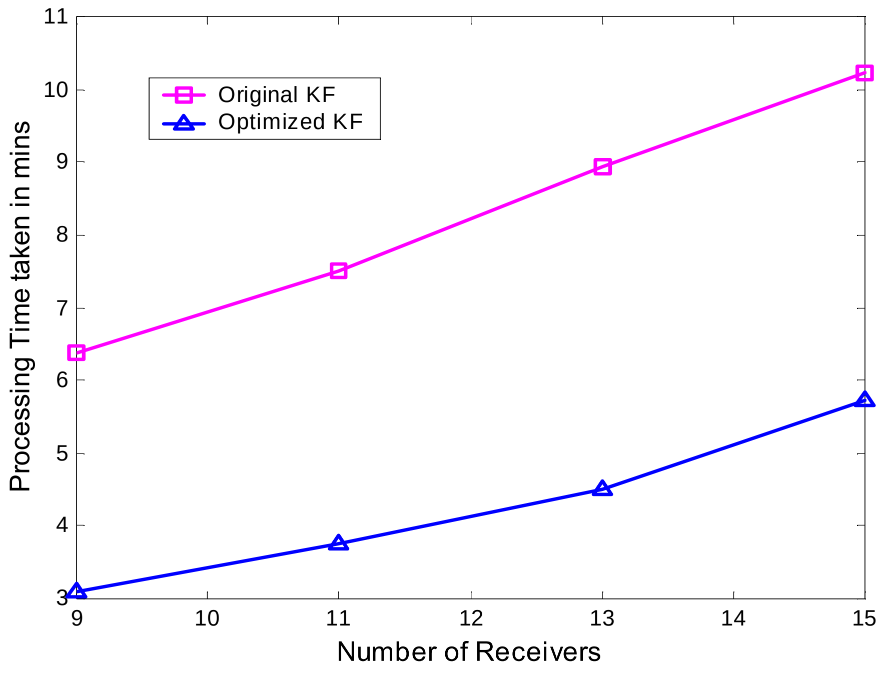

Figure 5.4 Comparison between performance of optimized and unoptimized KF versus number of receive apertures in the sparse satellite cluster

Further modifications could be done in the real time (software or hardware) implementation of the KF to further optimize it. But is there a way to improve the processing efficiency by changing the very way in which the KF is implemented? A number of experiments have been performed and the results analyzed to answer this question. The experiments and their results are presented in the next sections.

5.2 Number of Measurements required for processing to making an Optimal Estimate (Sequential Estimation)

It could be seen from Figure 3.3 that KF is able to give good results for random complex scattering coefficients. It could also be noticed from the above figure that the Normalized MSE reaches –25dB even before processing of 60% of the data. This means that if a way is found to quantify this position where the filter reaches an optimal solution then the processing could be stopped at that position thereby reducing the processing time. Though there is a loss in accuracy of the estimate, a huge increase in processing speed could be achieved. In this section investigation results of the experiments performed to find a solution to quantity this behavior are presented. The basic methodology is to choose a parameter in the filter, which varies in a similar fashion as the Normalized MSE, so that processing could be stopped when the parameter reaches a preset limit. For this reason the variation of the parameters in KF with processing of data is analyzed and the important results observed are presented here.

It is also necessary that the variation in the chosen parameter is stable for all scattering scenarios. So experiments are performed for four types of scattering coefficients. The scattering coefficients are simulated by generating M (number of resolution cells) random complex numbers. These numbers are generated for four cases as follows

i. They are real and equal

ii. The phase is kept constant white randomizing their magnitudes

iii. Their magnitudes are kept constant while randomizing their phase

iv. Both phase and magnitudes are randomized

The magnitudes are chosen as a random value out of a 256-level gray scale. An Image is generated by substituting the random sets of the above-generated numbers of $\mathbf{g}$ for the corresponding pixel values. This image representing the scattering coefficients of all target pixels is given as input to the radar simulator. Then Kalman Filtering is applied on the obtained measurements. Variations in the parameters of the KF are saved while the filtering process is being performed.

First of all how does the Normalized MSE vary with processing for all the above cases? Is it going to vary in a similar fashion for all the cases or is it going to be different? Figure 5.5 shows that it varies in a similar fashion for all the cases. The KF converges to the final estimate in a similar way given that the number of receivers in the constellation are fixed (fixed number of measurements taken and fixed resolution). So a parameter has to be found in the filter, which sort of emulates this variation in Normalized MSE and flattens out when there is not much of an improvement in the estimate.

A number of parameters could be expected to vary in a similar fashion. For example the innovation energy contained in the latest measurement (dealt in section 4.1) could be expected to decrease as the filter processes more and more number of measurements. But we have seen that in Figure 4.5 that though the innovation energy decreases gradually, it still has a small random component attached to it depending on the obtained measurement. Figure 5.6 demonstrates this hypothesis.

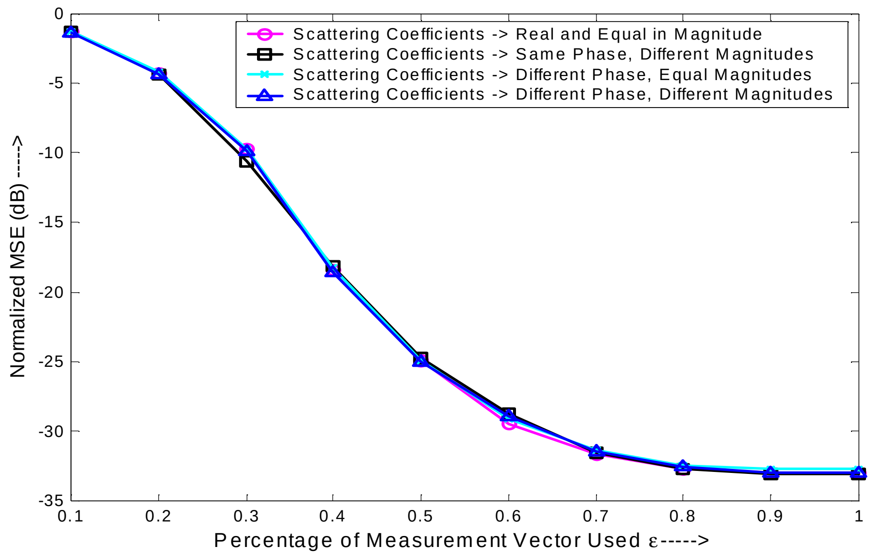

Figure 5.5 Normalized MSE (dB) vs. the fraction of measurement data processed for random scattering coefficients

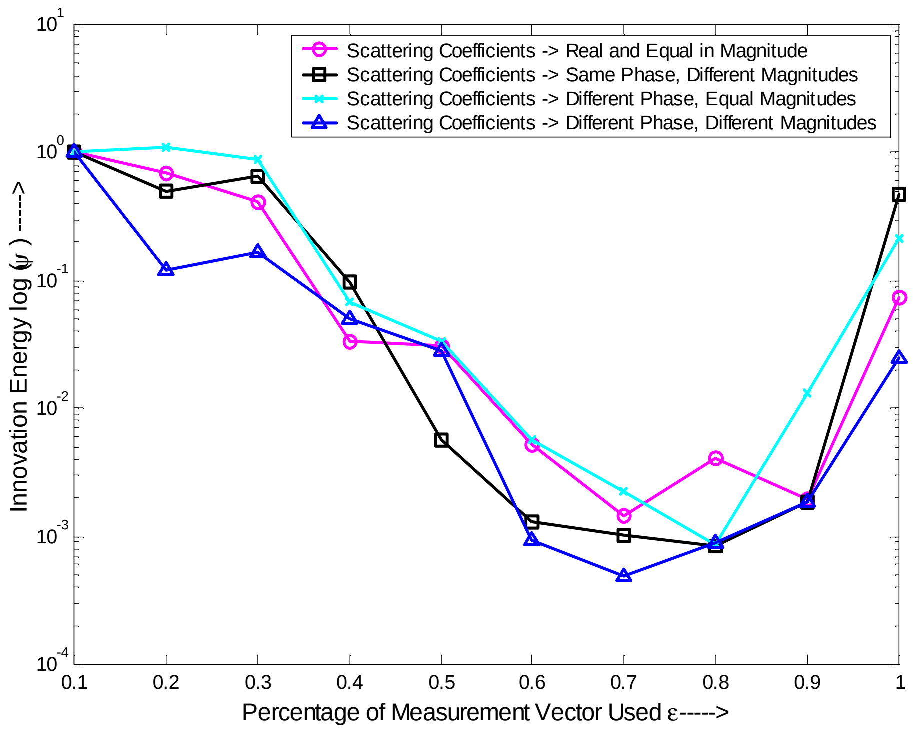

Figure 5.6 Innovation Energy vs. the fraction of measurement data processed for random scattering coefficients

Kalman gain could be considered basing on the assumption that as the filter converges to a better estimate the gain required to reduce the estimate error decreases. The Kalman gain computed for each iteration is quantified in terms of the RMS value of all its elements as

$$\mathsf{G}=\sqrt{\frac{\sum_{i=1}^{M}\sum_{j=1}^{NBT/L}\left|\mathbf{G}(i,j)\right|^{2}}{(MNBT/L)}}. \tag{5.1}$$

Finally the obtained 'G' for all iterations are normalized. It could be seen from Figure 5.7, that though the Kalman gain decreases gradually the flattening out at the

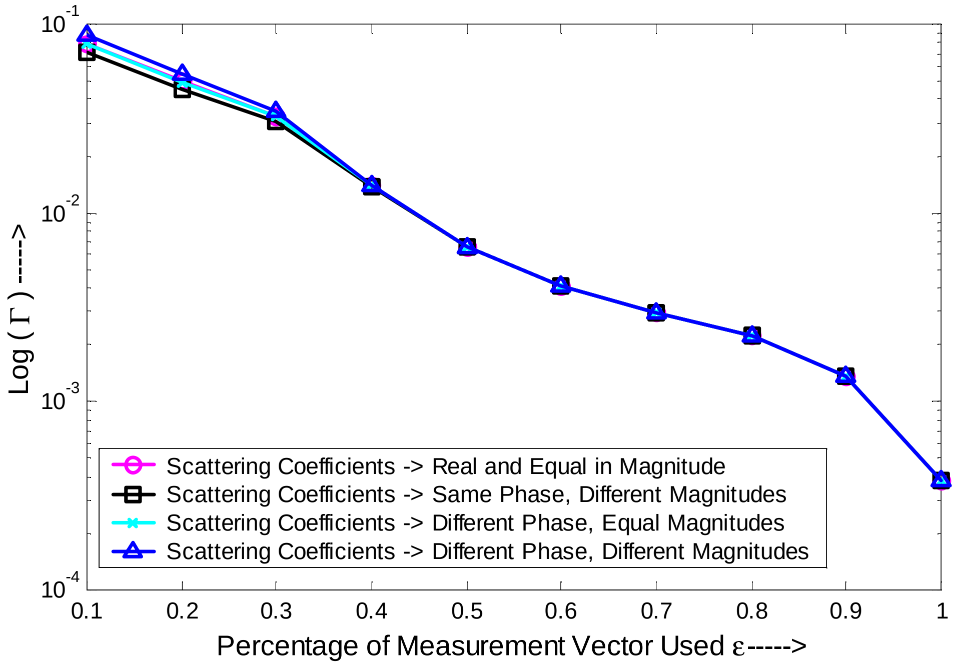

Figure 5.7 RMS value of elements of Kalman Gain Matrix vs. the fraction of measurement data processed for random scattering coefficients

end is not very obvious. Moreover it could be seen that (for the last iteration) it too has a random component depending upon the obtained measurement.

Another parameter that could be considered is 'm' which quantifies the update made in the present estimate when compared to the estimate obtained for the previous iteration.

$$m = \frac{\left(\hat{\mathbf{g}}(l) - \hat{\mathbf{g}}(l-1)\right)^{\mathrm{H}}\left(\hat{\mathbf{g}}(l) - \hat{\mathbf{g}}(l-1)\right)}{\hat{\mathbf{g}}(l)^{\mathrm{H}}\hat{\mathbf{g}}(l)} \tag{5.3}$$

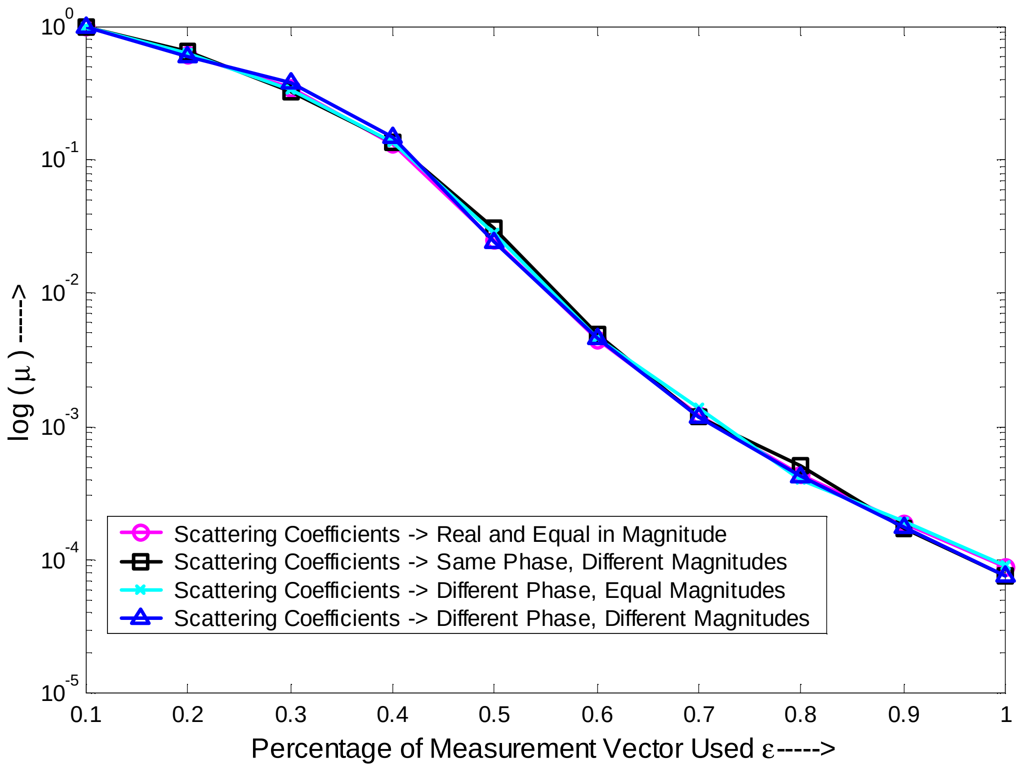

Figure 5.8 Variation of the update in present estimate as compared to the previous estimate versus the fraction of measurement data processed.

From Figure 5.8 it could be seen that the variation in m is constant for all the cases considered. But the problem with it is that, it doesn't flatten out at the end, or in other words it keeps on varying throughout the processing.

One other parameter that could be considered is the trace of the error correlation matrix $\mathbf{K}_g(l)$, updated for each iteration. The obtained trace is normalized as follows

$$\text{Normalized trace of } \mathbf{K}_g(l) = \frac{\overset{M}{\underset{i=1}{å}} \left|\mathbf{K}_g(i,i)\right|}{s_g^2} \qquad (5.4)$$

As the $\mathbf{K}_g(0) = s_g^2\mathbf{I}$ so the above value will be 1 for the initial error correlation matrix. It could be seen from Figure 5.9 that the trace of the Error Correlation Matrix

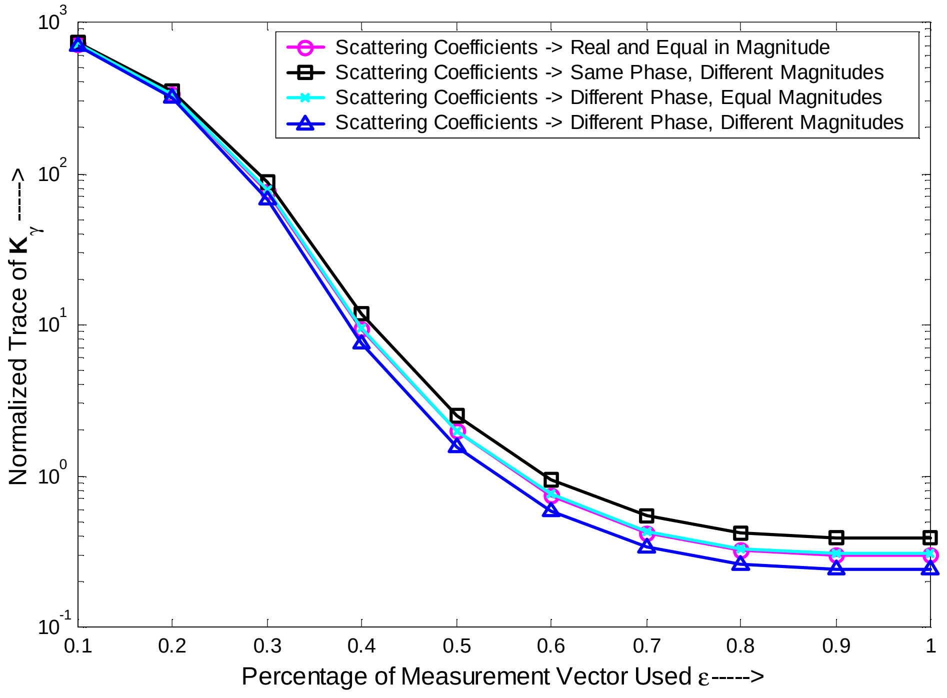

Figure 5.9 Trace of the Target Correlation matrix versus the fraction of measurement data processed.

varies in a way very similar to that of the Normalized MSE. It decreases gradually and then at the end sort of flattens out. Therefore this measure could be used to quantify the amount of data required for obtaining an optimal solution. The filter processing could be stopped at a prefixed value for the trace of $\mathbf{K}_g(l)$ by incorporation of a logic control quite easily. As the time taken for processing of each iteration is the same, the processing speed increases by 40% for this particular case. In general, the gain in speed of processing depends on the target clutter and measurement noise. More experiments have to be performed to verify the above sequential estimation process for different configurations of satellite constellation and noise scenarios. It should be noted that there is no possibility for this kind of an implementation in Matched, ML and MMSE filters.

5.3 Methods Pursued to reduce Error Correlation Matrix Computations

The processing time consumed by operations defining the Error Correlation matrix, $\mathbf{K}_g$, in the KF constitutes a major part of the overall processing time. For example for M=32^2 (32 by 32) the total number of elements in the Target Correlation Matrix are more than a million, and if M is increased to 256^2 for a fixed target region then the number of elements increases to a whopping 4.2 billion. Even initializing a matrix of such a huge order requires huge RAM and operations involving that will create huge complexities in the implementation. However huge the measurement vector may be, it can be divided into smaller vectors of optimal size. A similar sort of thing could be done for the Target Correlation Matrix too, but that itself will add a lot more complexity in the implementation, and will also increase the time taken for read, write memory operations.

So, a detailed investigation is carried on to find out if a fewer number of elements in the target correlation matrix could be considered while being able to maintain its important features intact. The first thing that could be tried out is by considering only the diagonal elements of the target correlation matrix. In other words this is based on the assumption that the measurements obtained from a target pixel is uncorrelated to other target pixels.

5.3.1 Only Diagonal Elements in the Error Correlation Matrix are updated

The performance of KF is tested for a scenario where only the diagonal elements are updated in the error correlation matrix $\mathbf{K}_g$. It is tested for different values of L. The Figure 5.10 shows the performance of the KF versus L in terms of the accuracy of its final estimate.

As could be seen from the figure that for smaller values of L the KF is able to converge to a good estimate. But as L increases the performance of the KF degrades. It was shown earlier in Figure 4.2 that the non-diagonal elements of the target correlation matrix too have significant values. So how could it be possible to obtain a good estimate when the non-diagonal elements are neglected in the processing? A lot of analysis was made to find answers to this question and finally were able to make some significant conclusions.

The first thing that was found was that even though only the diagonal elements were considered, the Kalman Gain produced is quite close to the one obtained by considering all the elements.

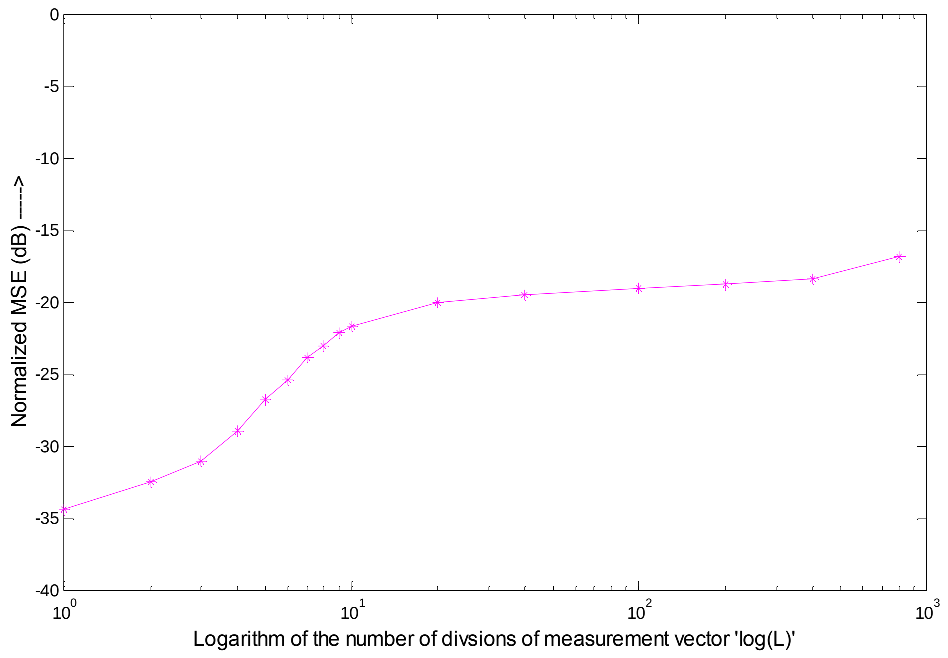

Figure 5.10 Normalized MSE (dB) versus L for a case where only the diagonal elements are updated in the $\mathbf{K}_g$ matrix

The Normalized difference between the Kalman Gain obtained when only the diagonal elements are considered in target correlation matrix $\mathbf{G}_u$, and the normal kalman gain obtained when all the elements are considered $\mathbf{G}$ is given by

$$x = \frac{\sum_{i=1}^{M} \sum_{j=1}^{M} \left| \mathbf{G}_u(i,j) - \mathbf{G}(i,j) \right|^2}{\sum_{i=1}^{M} \sum_{j=1}^{M} \left| \mathbf{G}(i,j) \right|^2} \tag{5.5}$$

It could be seen in figure 5.11 that for L=2, the computed Kalman Gain for this case is somewhat similar to the normal scenario. This could be explained on the basis of two reasons. One being the fact that during the initial stages of processing the diagonal elements of the target correlation matrix are significantly larger than the non-diagonal elements. So during the initial stages of processing the effect of neglecting the non-diagonal elements is not realized.

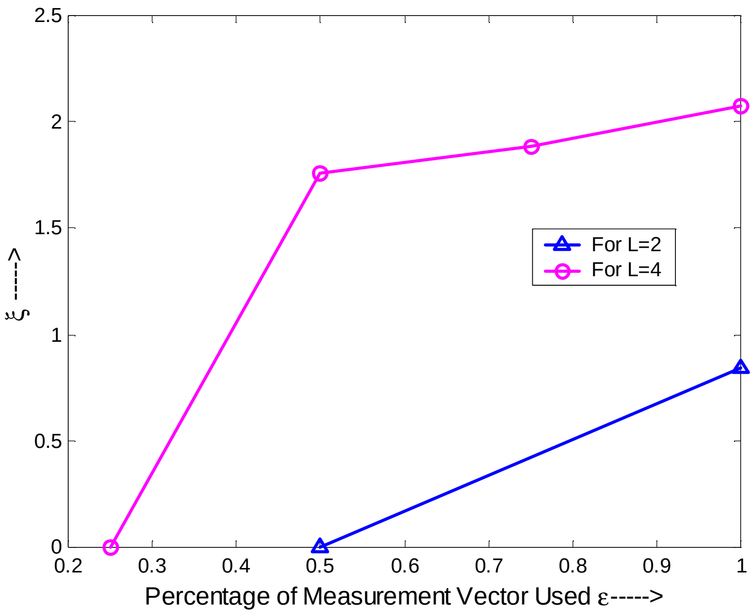

Figure 5.11 χ **The normalized Variation between $\mathbf{G}_u$ and $\mathbf{G}$ versus fraction of measurement vector processed for *L*=2 and *L*=4**

The other reason being that during the final stages of the processing (as explained in section 4.1 the variance of the noise plays a major role in the computation of Kalman Gain and so the variation made in the target correlation matrix doesn't make an effect. But as *L* increases it is seen that the diagonal elements do not decrease with processing as in the case where all the elements are considered. This causes more variations in the Kalman Gain computed.

So for smaller values of *L* it is possible for the KF to converge to an optimal estimate even though the correlation between reflectance values obtained for pixel-to-pixel are neglected. But for smaller values of L the complexity in the system due to the matrix inverse computation is still present. Basing on the accuracy required and the processing power available a tradeoff could be reached to decide upon a particular L value. But the overall processing time reduces to a great extent as only a fraction of

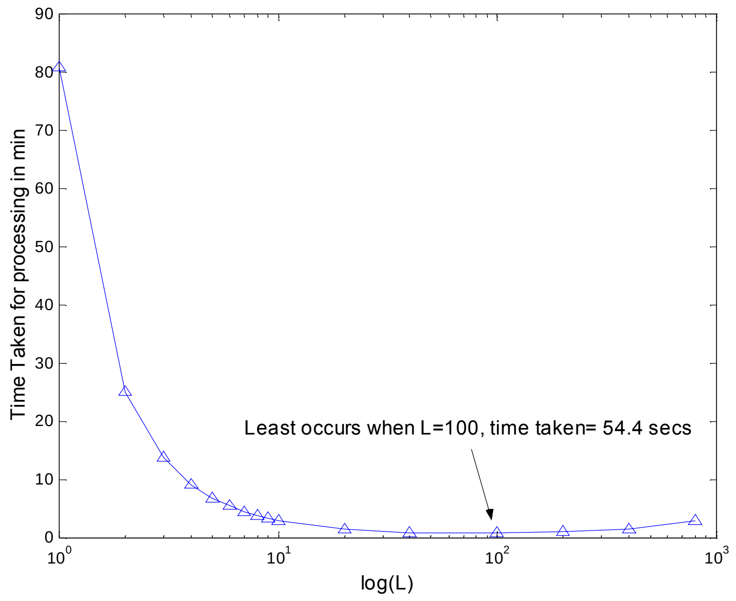

Figure 5.12 Processing time taken versus L for a case in which only the diagonal elements are considered in the error correlation matrix $\mathbf{K}_g$

the total elements present in the target correlation matrix need to be computed. Figure 5.12 shows this huge reduction in processing time.

5.3.2 Diagonal Spread is considered

As there is a high probability that the reflectance values obtained from neighboring resolution cells are highly correlated it was expected that further improvement could be obtained in the accuracy of the estimate by considering elements adjacent to the main diagonal. Figure 5.13 shows the elements considered in $\mathbf{K}_g$ for a case in which elements lying in 64 cross diagonal lines adjacent to the main diagonal are considered on one side, i.e. a total of 128, (64*2) cross diagonal lines are considered. It could be seen from Figure 5.14 that as the considered spread of the diagonal increases the MSE decreases. So this offers an effective solution to the tradeoff between accuracy and processing time. But it should be noted this improvement in accuracy is limited and if the spread is increased to more than a certain extent than non linear processing error occurs (the last element in Figure 5.14, 256*2+1 = 513 adjacent diagonals) and the KF goes astray.

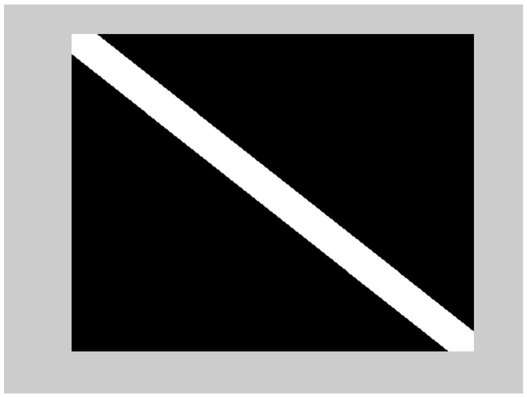

Figure 5.13 Elements of $\mathbf{K}_g$ considered for a diagonal spread consisting of 129 lines adjacent to the diagonal on either side

It could also be noted that the final estimate is going to depend on the value of L as explained in the previous section.

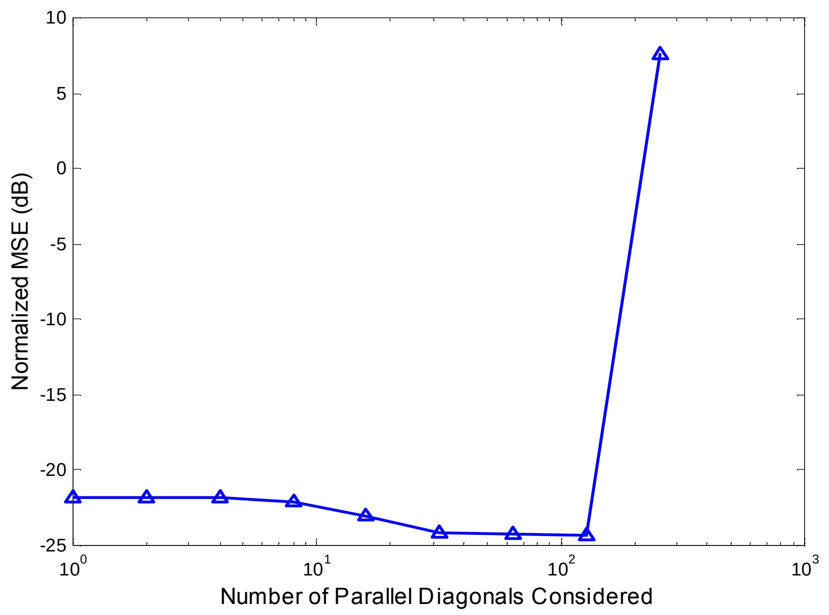

Figure 5.14 Normalized MSE versus the number of adjacent diagonal lines considered on one side

5.3.3 Error Correlation Matrix is updated based on the Expected Measurement Correlation Matrix

The expected correlation between the responses obtained from the illuminated targets $\mathbf{a}$ could be computed from the normalized response vectors and was given in (3.21). The question is whether the KF will be able to converge to an optimal estimate if only the corresponding elements in $\mathbf{a}$ signifying higher correlation are updated in the Error Correlation Matrix $\mathbf{K}_g$. In other words, it is desired to see how far the expected measurement correlation matrix is close to the correlation between the reflectance values obtained from pixel to pixel. If the KF is able to reach a good estimate than new procedures could be implemented to update only those points thereby reducing the processing time.

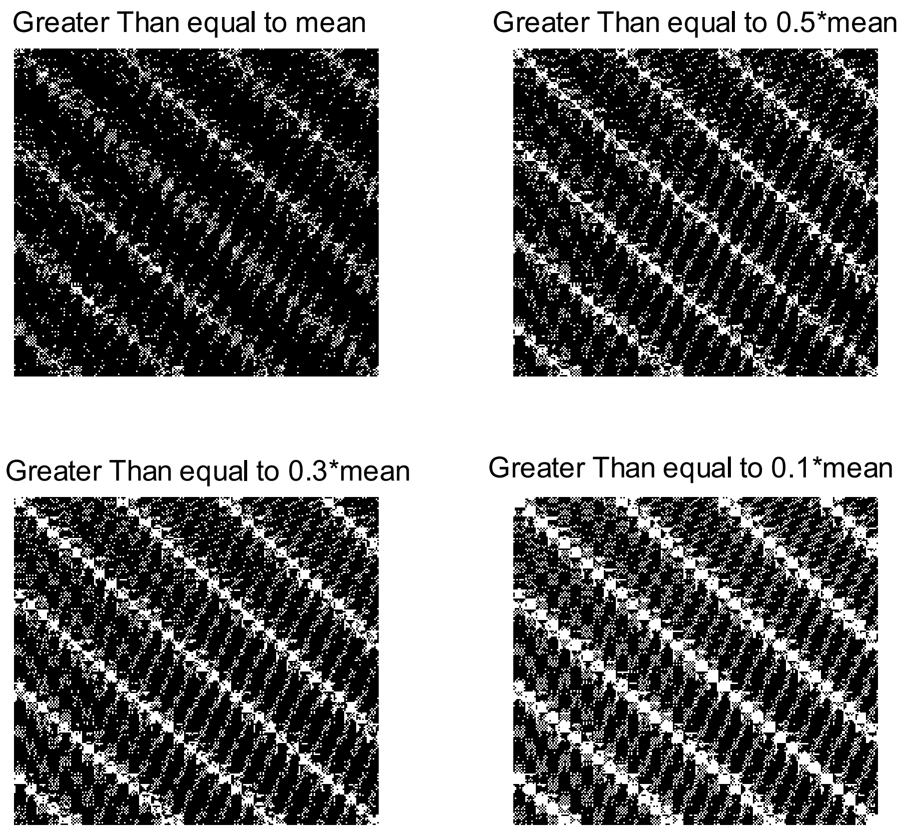

Figure 5.15 Normalized MSE versus the number of adjacent diagonal lines considered on one side

Experiments were performed to answer this question. Four experiments were performed by considering only those elements in $\mathbf{K}_g(l)$ which have significant values in a . The significant elements in a are chosen by comparing them to a preset threshold. The threshold is obtained in terms of the absolute mean value of all the elements in the a . Figure 5.15 shows the thresholds used and the points updated in the target correlation matrix.

Figure 5.16 shows the convergence of the KF for corresponding thresholds used to choose the elements (where L = the percentage of elements considered out of a total number of M^2 elements in $\mathbf{K}_g(l)$). It could be seen that the KF is unable to give a good estimate. So basically this method cannot be pursued to reduce the computation load as it affects the estimation process.

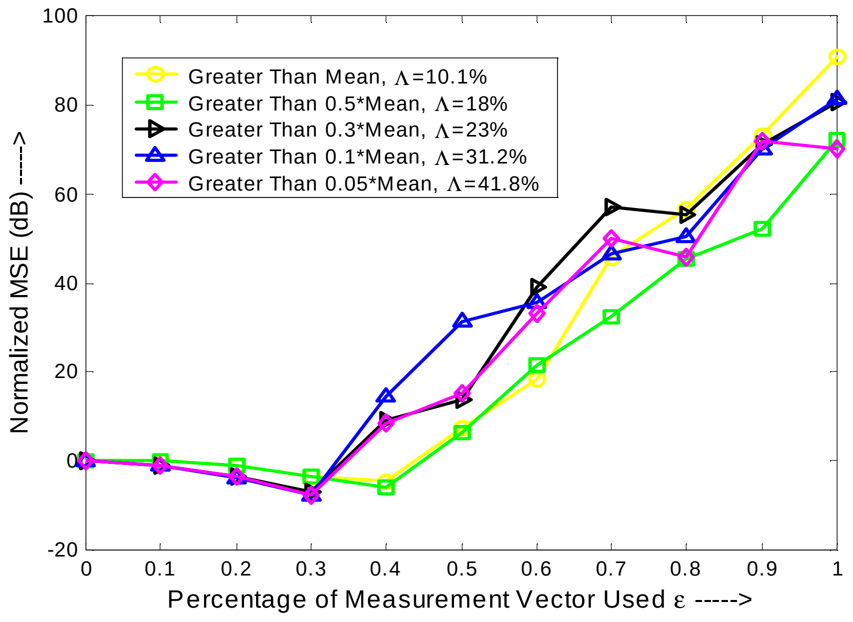

Figure 5.16 Normalized MSE versus the fraction of measurement data processed, for different thresholds

5.3.4 Correlation of only targets closer to each other is considered

It is highly probable that the scattering characteristics of target pixels very close to each other are similar. For example in the picture of the football stadium all the target pixels constituting the structure of the stadium exhibit similar scattering characteristics. Similarly all the target pixels constituting of grass exhibit similar scattering characteristics. If SAR pictures are taken of huge grasslands or of large forests than the scattering scenario exhibited by many target pixels will be quite similar. Based on this logic the KF performance is tested by considering only those points in the target correlation matrix, which are expected to have high correlation. Assuming different kinds of correlation scenarios different experiments are performed. For example say, a target pixel has high correlation with target pixels that are positioned at a distance less than two times the resolution considered for each target pixel. The situation is depicted in Figure 5.17.

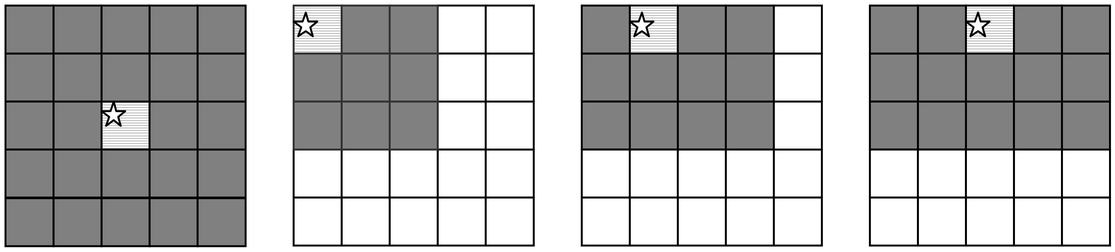

Figure 5.17 Shows the target pixels (dark shade) supposed to be highly correlated to the concerned target pixel ('star') located at different parts of the region concerned

The first picture form the left in the figure shows a scenario where the target pixel is well inside the target boundaries. The dark shaded squares in the figures are supposed to be highly correlated to the square consisting of the 'star'. The other pictures depict scenarios for target pixels lying at the edge of the target boundary. The targets considered to be highly correlated to the particular target pixel are dark shaded. The other boundaries of the target pixel represent the area outside the concerned target region.

Figure 5.18 shows the points updated in the target correlation matrix for corresponding correlation scenarios. For example, the plot on the top-left shows the points chosen for consideration in the target correlation matrix for a scenario where it is assumed that a target pixel is highly correlated to its just neighbor target pixels and

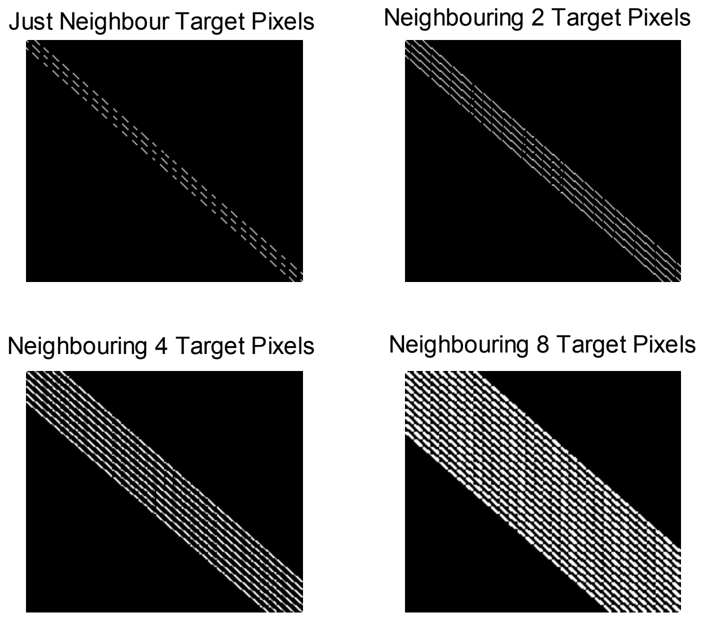

Figure 5.18 Points considered in the target correlation matrix for corresponding correlation assumptions

is uncorrelated to the other target pixels. So as could be seen in the plot a very few number of points are considered. It is very hard to predict the actual dependencies of the target pixel correlations, it depends on the area where the target the SAR is mapping. Figure

Figure 5.19 shows the performance of the KF for different correlation dependencies. It could be seen that as the number of neighboring targets supposed to be highly correlated with the desired target pixel is increased the MSE decreases. But as this number is increased further the KF gives random estimates.

If different target correlations are characterized earlier, then that information could be used to optimize the KF. If certain approximations could be made to characterize the target correlations, then it will help in choosing the significant points in the $\mathbf{K}_g$ matrix.

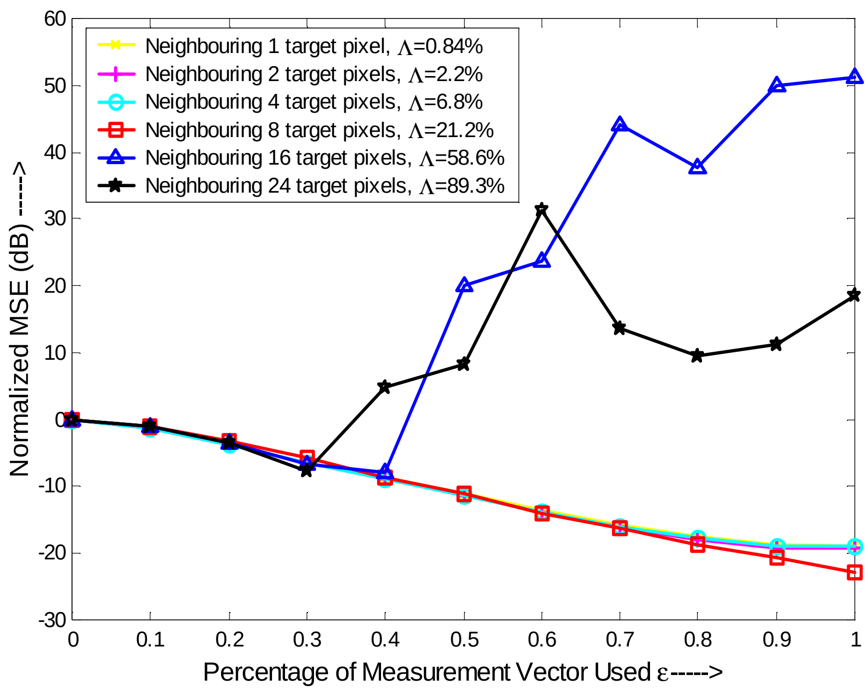

Figure 5.19 Normalized MSE versus the fraction of measurement used for different correlation characteristics

In all the scenarios considered above the accuracy of the estimate is traded for higher processing speeds and reduced computational load. But is it possible to maintain the accuracy of the estimate while being able to speed up the process? If there are any solutions for this, are they economically and technologically viable? Much research and development is being done to develop faster processors, and more optimal software's. Could these be used to fasten the processing? To get answers for the above questions the thesis investigation has ventured into the world of "parallel processing". The details and results of which are given in the next chapter.

6. PARALLEL IMPLEMENTATION OF KF

The scenarios considered in earlier chapters for KF implementation and testing represent a scaled down versions of the real SAR scenario. In a practical scenario, there is requirement for a finer resolution, which in turn needs large number of measurements. As showed in section 5.1 the matrix multiplications play a major part in the overall processing and if they could be done faster, the overall speed of the filter also improves linearly. Moreover, as the RAM required for implementing KF in Matlab is high, so the implementation should be redone using a programming language, which requires lesser RAM. Matlab allocates 8 bytes of memory for storing either float or double variables. If a more flexible and procedural language like C is chosen, then the memory efficiency will improve. Again, as the inbuilt functions in Matlab (matrix multiplication and inverse function calls) are replaced by user-defined functions, it is possible to obtain a clearer picture of the optimization scenario.

6.1 C code

The main reasons behind opting for a procedural language like C is to improve the processing efficiency and memory utilization. It also facilitates introduction of new modules for future variations in the functionality of the code and debugging. A lot of highly tested and popular software procedures are already available in C for implementation of parallel processing using multithreads or multiple processors. Most of the parallel processing softwares available for Matlab introduce parallelism by compiling Matlab scripts into a parallel C code. Moreover, the validity of these available softwares have not been proven as they have come quite recently.

The implementation was done in a modular fashion for easier understanding of the flow of the algorithm. Subroutines were developed for various matrix operations. Lot of care has been taken to reduce the time taken for memory access operations and other program latencies. For example all the matrices are initialized by allocating memory in a regular organized fashion. A lot of time will be required for memory access operations if the matrices were allocated memory randomly.

6.2 Parallel Processing

6.2.1 Necessity and use of Parallel Processing

There are many other approaches to speed up the implementation of Kalman Filter apart from parallel processing. The most significant one is to make the standalone single-processor design larger (e.g., increase the amount of memory it can directly address, and for SAR applications the requirement is huge) and more powerful (e.g., increase its basic word length and computational precision) and faster (e.g., by using smaller-micron etching technology, packing more transistors into less space, and coupling everything with larger and faster communications pathways). But these upgrades will make it very expensive.

Parallel processing offers higher and reliable performance, at affordable prices. The basic logic in parallel processing is to divide an unmanageable large task into smaller tasks, which are more manageable. The divided smaller tasks could then be run on multiple processors. For example many of the research institutions and small companies have access to large number of small to medium scale CPU's offering reasonable speed and RAM. These institutions could use parallel processing as a tool to solve large problems, as they cannot afford to buy a super computer necessary for solving large problems. Moreover most of the available processors are underutilized or not utilized at all. Programs are not run on a CPU at all times of the day and even the usual programs that are run on a CPU utilize less than 20% of the available processing power. Hence, all this wasted processing power could also be used to solve problems, which usually require high-speed processors. In some cases multiple processors solve a large problem faster than a single high-speed processor.

6.2.2 Building a Parallel Virtual Machine

PVM version 3.4.4 [14] has been used for building a parallel virtual machine. PVM allows programmers to exploit a wide variety of computer types. A single parallel virtual machine comprising of a collection of heterogeneous computer systems could be built using PVM. The main components of a PVM system are a daemon (a program that runs in the background while performing the required task), called *pvmd3* (or *pvmd)* that resides on all the computers making up the virtual machine and a library consisting of PVM interface routines. The PVM interface routines are needed for cooperation between tasks of an application. This library is used for calling routines for message passing, generating processes, coordinating tasks, and modifying the virtual machine.

The method that has been used for parallelizing matrix multiplication is data parallelism. In this method all the tasks are the same, but each one only knows and solves a small part of the data. This is also referred to as the SPMD (single-program multiple-data) model of computing. All PVM tasks are identified by an integer *task identifier* (TID).

The Kalman filtering program written in C invokes calls to the PVM library. The program is compiled for all architectures in the host pool (machines used to build the parallel virtual machine), and the resulting object files are placed at a location, which is accessible from all machines in the host pool. To execute an application, a copy of one task (called as "master") is started by hand from a machine within the host pool. This process subsequently starts PVM tasks (called as "slaves"), creating a collection of active tasks that then compute locally and exchange messages with each other to solve the large problem.

6.2.3 Working Methodology

Figure 6.1 gives a pictorial description of the working methodology of the parallel implementation of KF. A network ("host pool") is built by adding on any number of accessible processors. These processors can be single processors of different architectures, or can be part of a multiprocessor system. The working of the developed system can be explained briefly as follows:

- PVM is started and a parallel virtual machine is built by adding on multiple processors

- The main program for Parallel implementation of KF is compiled and run on any one of the processors (acts as the "Master") of the host pool
- The main program initiates PVM thereby spawning processes on all the processors of the host pool
- The program starts the Kalman loop which calls the subroutine for performing matrix multiplications
- The Master processor then sends the matrix data to the slaves. Each Slave is sent one row number of the 1st matrix in the multiplication, in sequential order
- Each Slave works on the row that is assigned to it and computes the corresponding row of the resultant matrix after multiplication. After completion the computed row is sent back to the Master
- The Master doesn't wait for all slaves to complete their computations to send the next row number. It sends the next row number to the first Slave sending the next row information. In this way dynamic load balancing (effective resource utilization) is introduced in the system, and hence the work done by each slave depends on its processing speed. So faster slaves need not wait for slower slaves to finish their work and so no processor stays idle
- Finally after calculating all the rows of the resultant matrix, the Master stops the slave processors and sends the output matrix to the main program
- The main program then continues on with the other functions and the whole process continues for the next matrix multiplication

- Finally after the processing is done on all the measurement data, the main program finalizes PVM

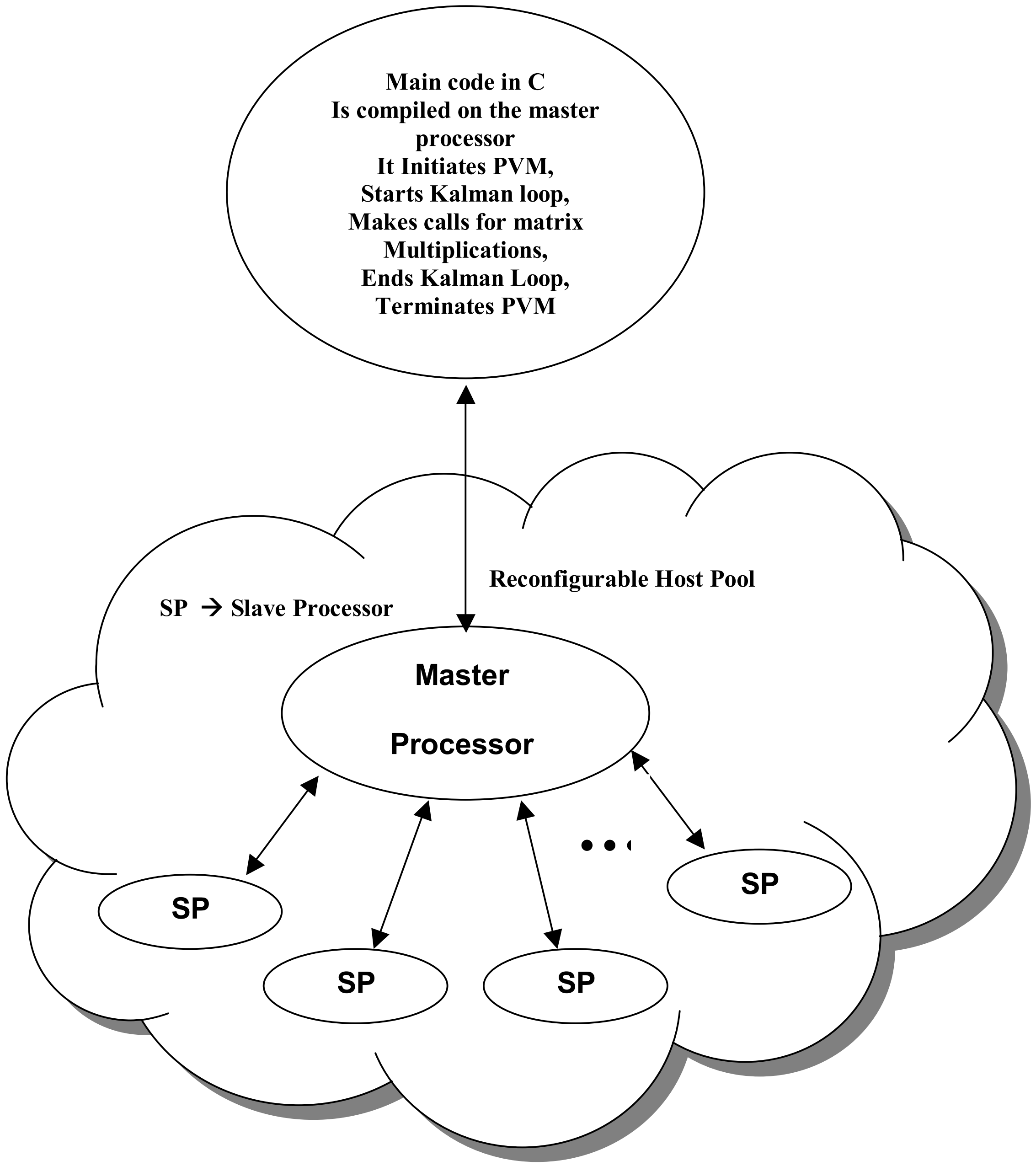

Figure 6.1 Diagram depicting the Parallel processing structure implemented in KF

6.2.4 Provisions made for huge matrix sizes

Changes could be easily made to vary the information transfer between the Master and the Slaves. For example in the earlier case, the Master sends both the matrices to the slaves and sends the row numbers of the first matrix in sequential order. Instead of that it could be made to send only the smaller matrix to the Slaves and send row data (single or multiple rows) of the larger matrix in sequential order. In this way processors having less main memory could also be added to the virtual machine. If both the matrices to be multiplied are large then change could be made so that the Master sends the row data of the first matrix and column data of the second matrix to each slave. Care should be taken to combine the information obtained from each slave to form the resultant matrix.

6.2.5 Testing and Results

A multiprocessor system consisting of 4 Intel Pentium III processors with processing speed of 697 MHz each has been used for testing the developed parallel KF software (testing for Matlab implementation has also been done on a similar processor). Testing is also done by adding another dual processor (Intel Xeon) system offering a processing speed of 2.2GHz per processor to the above system. The results obtained are presented in Figure 6.2.

It could be seen that the processing time for the C implementation of KF is higher than that taken for implementation in Matlab. This is due to the incorporation of LAPACK (Linear Algebra Package, is a large multi-author, which Fortran library for numerical linear algebra), and machine-dependent optimized BLAS (Basic Linear Algebra Subprograms) for matrix computations in Matlab. LAPACK uses block algorithms, which operate on several columns of a matrix at a time. On machines with high-speed cache memory, these block operations can provide a significant speed advantage. Hence further work can be done to incorporate BLAS in the developed KF code.

But as we go for finer resolutions and as the data and matrix sizes increase further, the developed parallel C code performs better than the Matlab code. This is because for coarse resolutions, the potential of parallel processors is not fully realized as a significant amount of time is consumed for inter-processor communication and

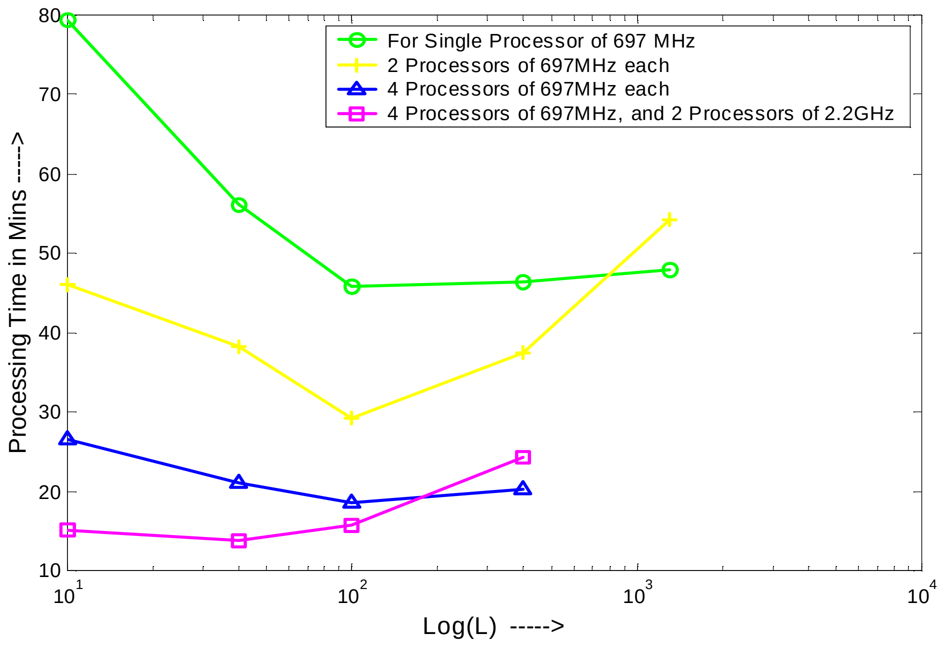

Figure 6.2 Processing time versus L for different number of parallel processors used

for spawning multiple tasks and actually starting them. But as the resolution becomes finer the processing time consumed by these latencies is less significant when compared to that taken for the required computations, and the worth of the developed parallel code becomes more discernable. The most important advantage offered by the developed parallel code is that it could be run on any processor and could be easily modified to adjust to the available RAM in the machines. In other words the developed code could be run on any machine independent of its hardware characteristics, and the speed of processing could be increased by just adding more machines.

The improvement in speed obtained by adding multiple processors is evident from Figure 6.2. For example for L=10 there is a speed improvement of almost two fold when the number of processors are doubled. While for L=1300 it could be seen that the processing speed actually decreases with addition of a processor. This is because more time is consumed for communication between processors than the actual processing time. This could also be seen for L=400, the processing time gets reduced to half in this case when the number of processors are increased from 2 to 4, but when they are increased to 6, the processing time taken increases. Moreover since the two newly added processors are of a different multiprocessor system, so there is a significant increase in the communication time. This shows the importance of the connection network of the multiple processors. It is always advisable that the slave processors used be connected directly to the master processor.

It could also be seen from the figure that though for 1,2 and 4 processors the processing is fastest for L=100, the speed improvement between L=100 and L=40 goes on decreasing. The figure also shows the increase in communication time as the number of iterations is increased. All these factors have to be taken into account while using the developed parallel KF.

7. Conclusions and Future Work

7.1 Conclusions

Matched, ML and MMSE filters have been designed to process data obtained from sparsely populated multiple aperture spaceborne radar. But these filters have their own disadvantages. Matched filter is unable to eliminate measurement error due to clutter, while the ML filter is unable to minimize error due to noise. The MMSE filter though being able to give a good estimate, has inherent computational complexities. A new filter is developed to give the same accuracy offered by MMSE while being able to reduce the computational load inherent in MMSE.

The developed Kalman Filter (KF) is an iterative implementation of the MMSE, and is tested for different real-time SAR applications and for different target characteristics. It is highly optimized by considering all the matrix operations involved. It is tested for dependencies on the initial parameters given to it, and optimal values are chosen for them.

Many tradeoff scenarios between the accuracy of the estimate and processing load have been proposed. Some of these proposed methods decrease the processing load many fold for a small loss in the accuracy of the estimate.

Finally a parallel version of KF is developed so as to distribute the processing load on multiple processors. This parallel code neutralizes the dependency of KF processing on super-computers or high-speed processors, and could be implemented on any kind of a processor environment.

7.2 Future Work

The first enhancement that could be made is by testing the system on real data obtained from spaceborne radar instead of a simulated version. This can determine the performance benefit of the proposed filter more effectively. Further the different optimization scenarios can be tested on the real data and quantitative results could be obtained. LAPACK (Linear Algebra Package) and BLAS (Basic Linear Algebra Subprograms) routines should be incorporated in the developed parallel code to further improve its processing speed. Further testing should be done for different resolution scenarios and different parallel network configurations.

8. REFERENCES

[1] N. Goodman, D. Rajakrishna, and J.M. Stiles, “Wide swath, high resolution SAR using multiple receive apertures,” in *Proc. of the IEEE International Geoscience and Remote Sensing Symposium*, Hamburg, Germany, pp. 1767-1769, June 1999

[2] J.M. Stiles, N. Goodman, and S. Lin, “Performance and processing of SAR satellite clusters,” in *Proc. of the IEEE International Geoscience and Remote Sensing Symposium*, Honolulu, Hawaii, pp. 883-885, July 2000

[3] J.M. Stiles and N. Goodman, “Processing of multi-aperture SAR to produce fine-resolution images of arbitrarily large extent,” in *Proc. of the 2001 IEEE Radar Conference*, Atlanta, Georgia, pp. 451-456, May 2001

[4] N. Goodman and J.M. Stiles, “The information content of multiple receive aperture SAR systems,” in *Proc. of the IEEE International Geoscience and Remote Sensing Symposium*, Sydney, Australia, July, 2001

[5] N. Goodman and J. Stiles, “Resolution and Synthetic Aperture Characterization of Sparse Radar Arrays”, submitted to the *IEEE Transactions on Aerospace and Electronics Systems*, Dec. 2001

[6] N. Goodman, S. Lin, D. Rajakrishna, and J. Stiles, “Processing of multiple-receiver, spaceborne arrays for wide-area SAR,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, no. 4, pp. 841-852, April, 2002

[7] Anand Sundaram, “A Software Simulator for Multi-Aperture Spaceborne Radar”, Master’s Thesis, The University of Kansas, Nov. 2000

[8] Nathan Goodman, "SAR and MTI Processing of Sparse Satellite clusters", Doctoral thesis, The University of Kansas, July 2002

[9] Guruvayurappan, "Spatial and Temporal Processing for a Compact Landmine Detection Radar", Master's Thesis, The University of Kansas, Oct. 2002

[10] K. Sam Shanmugan and A.M. Breipohl, " Random Signals Detection, Estimation and Data Analysis, John Wiley & Sons, Inc 1988

[11] Simon Haykins, "Adaptive Filter Theory", Pearson Education, Inc. 2002

[12] Merill I. Skolnik, "Introduction to Radar Systems", McGraw Hill Publications, 3rd edition, 2001

[13] Merill I. Skolnik, "Radar Handbook", McGraw Hill Publications, 1970

[14] Al Geist, Adam Beguelin, Jack Dongarra, Weicheng Jiang, Robert Manchek, Vaidy Sunderam, "PVM: Parallel Virtual Machine, A User's Guide and Tutorial for Networked Parallel Computing", MIT Press, 1994

[15] Steven M. Kay, "Fundamentals of Statistical Signal Processing", Prentice Hall, Inc. 1998

[16] H. W. Sorenson, "Least-Squares Estimation: from Guass to Kalman", IEEE Spectrum, vol.7, pp. 63-78, July 1970

[17] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", Trans. ASME, J. Basic Eng., Series 82D, pp. 35-45, Mar.1960

[18] J. Patrick Fitch, "Synthetic Aperture Radar", Springer-Verlag, 1988