

A NSGA-II BASED SENSOR SELECTION SCHEME FOR TARGET TRACKING IN
WIRELESS SENSOR NETWORKS



by
Mert Lale

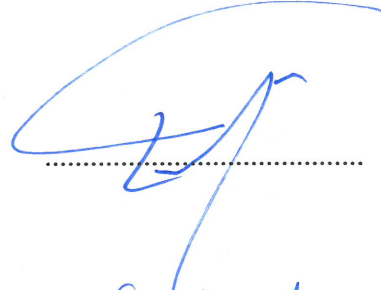
Submitted to Graduate School of Natural and Applied Sciences
in Partial Fulfillment of the Requirements
for the Degree of Master of Science in
Electrical and Electronics Engineering

Yeditepe University
2019

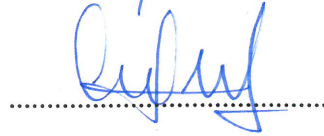
A NSGA-II BASED SENSOR SELECTION SCHEME FOR TARGET TRACKING IN
WIRELESS SENSOR NETWORKS

APPROVED BY:

Assoc. Prof. Dr. Engin Maşazade
(Thesis Supervisor)
(Marmara University)



Prof. Dr. Duygun Erol Barkana
(Yeditepe University)



Prof. Dr. Haluk Küçük
(Marmara University)



DATE OF APPROVAL: /.... /2019

ACKNOWLEDGEMENTS

I would like to thank my advisor Engin Maşazade for the continuous support of my M.Sc thesis. I wrote this thesis with the help of his guidance through the research.

In addition, my special thanks to my friends Cihan Yüksel and Veysel Yaman Akgün.

I thank to my family for their faith in me and their support.



ABSTRACT

A NSGA-II BASED SENSOR SELECTION SCHEME FOR TARGET TRACKING IN WIRELESS SENSOR NETWORKS

In this thesis, we study the sensor selection problem in target tracking for a wireless sensor network (WSN). The target emits energy and the sensors transmit their measurements from the target to the Fusion Center (FC). FC estimates the location of the target by using these measurements. Since a WSN may have limited resources, it is critical to gather measurements only from the most informative sensors rather than all the sensors in the WSN. Our aim is to find the sensor selection strategy at each time step of tracking by the joint minimization of objective functions representing the estimation error and total number of sensors transmitting to the FC, where we use a Non-dominated Sorting Genetic Algorithm - II (NSGA-II) to determine the solutions between the two conflicting objectives. Different from the existing results in the literature, our aim is to get the solutions of NSGA-II accurate and fast by setting right parameters. Firstly, rather than randomly initializing the initial population of NSGA-II at each time step of tracking, we use the solutions of the previous time step in the initial population of the current time step. Secondly, rather than executing NSGA-II for excessive generations to observe the near Pareto-optimal front, we define a stopping rule by using the Generational Distance metric. We further compare the solutions proposed Multi-objective optimization problem under different population sizes and crossover operators as well as under target trajectories with different process noise parameters, and different total number of sensors in the WSN.

ÖZET

KABLOSUZ ALGILAYICI AĞLARINDAKİ HEDEF TAKİBİ İÇİN BSGA-II TABANLI SENSÖR SEÇİM YÖNTEMİ

Bu tezde, kablosuz algılayıcı ağında (KAA) hedef takip problemi üzerinde çalışıyoruz. Hedef enerji yayar ve algılayıcılar bu enerjiyi ölçüp, ölçümlerini Tümeleştirme Merkezi (TM)'ne gönderir. TM bu ölçümlere göre hedefin yerini tahmin eder. Bir KAA'nın kaynakları sınırlı olduğundan, bütün algılayıcıları kullanmak yerine, yalnızca en bilgilendirici algılayıcılardan ölçümler almak önemlidir. Amacımız, Baskınlanmamış Sıralayan Genetik Algoritma-II (BSGA-II)'yi kullanarak, iki ayrı ve birbirileriyle ters orantılı görev fonksiyonları olan tahmin hatasını ve gönderim yapan algılayıcıların toplam sayısını her zaman adımında minimize etmek için bir algılayıcı seçme stratejisi bulmaktır. Literatürde bulunmuş sonuçlardan farklı olarak, hedefimiz doğru değişkenleri kullanarak BSGA-II'den sonuçları daha hızlı ve hatasız alabilmektir. İlk olarak, giriş popülasyonunu her zaman adımında rastgele oluşturmak yerine, önceki zaman adımının sonucundaki çözümleri kullanarak şimdiki zaman adımındaki giriş popülasyonunu oluşturuyoruz. İkinci olarak, BSGA-II'yi en iyi sonuçları içeren listeyi (Pareto- Optimal Front) elde etmek için gereğinden fazla nesille çalıştırmak yerine, Nesilsel Uzaklık ölçüsünü kullanarak bir durdurma kuralı tanımlıyoruz. Buna ek olarak, KAA'da farklı popülasyon miktarları, geçiş operatörleri, işlem gürültüsü katsayıları ve toplam algılayıcı sayıları kullanarak elde ettiğimiz çözümleri karşılaştırıyoruz.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	iii
ABSTRACT	iv
ÖZET	v
LIST OF FIGURES	viii
LIST OF TABLES	x
LIST OF SYMBOLS/ABBREVIATIONS	xi
1. INTRODUCTION	1
1.1. LITERATURE SURVEY.....	3
1.2. LIST OF CONTRIBUTIONS.....	6
1.3. THESIS ORGANISATION.....	8
2. PRELIMINARIES	10
2.1. EXPECTED VALUES, CORRELATION & COVARIANCE MATRICES OF RANDOM VARIABLES.....	10
2.1.1. Minimum Mean Square Error.....	12
2.2. OPTIMUM LINEAR SYSTEMS & ORTHOGONALITY CONDITION	13
2.3. 1-D KALMAN FILTER	16
2.4. VECTOR KALMAN FILTER	20
2.5. EXTENDED KALMAN FILTER	23
3. SYSTEM MODEL.....	25
3.1. SENSOR SELECTION	27
4. NONDOMINATED SORTING GENETIC ALGORITHM-II	29
4.1. DECISION MAKING WITH PSEUDO-WEIGHT CALCULATION.....	31
4.2. DECISION MAKING WITH MINIMUM DISTANCE TO ORIGIN	31
4.3. DECISION MAKING WITH KNEE POINT SOLUTION	32
4.4. Crossover OPERATORS & MUTATION.....	32
4.5. INITIAL POPULATION TYPES.....	34
4.6. STOPPING RULE.....	35
5. SIMULATIONS RESULTS AND DISCUSSION.....	37
5.1. SIMULATION PARAMETERS.....	37
5.2. NUMERICAL RESULTS.....	38

5.2.1. Effect of Fixed Number of Selected Sensors	38
5.2.2. Effect of Solution Selection on the Pareto-optimal Front	39
5.2.3. Effect of Population Size on Stopping Rule	40
5.2.4. Effect of Different NSGA-II Implementations	42
6. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS	50
REFERENCES	52



LIST OF FIGURES

Figure 1.1. Diagram of the target tracking process.....	2
Figure 2.1. Optimal linear filter structure [54]	14
Figure 2.2. 1-D Kalman Filter signal structure [54]	17
Figure 4.1. A basic flowchart of NSGA-II algorithm [50]	30
Figure 4.2. Pareto optimal front of NSGA-II after (a) G=1 , (b) G=3, (c) G=6, (d) G=10 generations, N=16 sensors in the WSN.	36
Figure 5.1. An example WSN model for target tracking where (a) N = 16 sensors are deployed randomly in the ROI., (b) Target tracking performance while all sensors are active at all time steps	39
Figure 5.2. Comparison while all and only the most informative sensors are selected by (a) MSE with all implementations, (b) MSE without worst implementations	40
Figure 5.3. Pareto-optimal front between the trace of error covariance matrix and total number of sensors selected observed at time steps (a) t = 1, (b) t = 3, (c) t = 6, (d) t = 9 ...	41
Figure 5.4. Comparison when using the knee point sol., pseudo-weight sol. And minimum distance sol. for $\tau = 10^{-2}$ by (a) MSE, (b) Total number of selected sensors	42
Figure 5.5. Comparison by multi-objective optimization methods by (a) MSE, (b) Selected sensor number	43
Figure 5.6. Generational distance metric after each generation for $N_{pop} = 100$ and $N_{pop} = 250$	44

Figure 5.7. Performances of 6 methods by (a) MSE for crossover 1 and 250 populations, (b) Sensor selections for crossover 1 and 250 populations, (c) MSE for crossover 2 and 250 populations, (d) Sensor selection for crossover 2 and 250 populations.....46

Figure 5.8. Performances of 6th Implementations by (a) MSE and (b) Total number of selected sensors.....47

Figure 5.9. MSE comparison for 6th Implementations versus while all and only the most informative sensors are active48

Figure 5.10. An example WSN model for target tracking where $N = 25$ sensors are deployed randomly in the ROI48

Figure 5.11. Performance comparison between implementation 1 and implementation 6 with CX2, $N_{pop} = 250$ and $N_{pop} = 25$ by (a) MSE and (b) Total number of selected sensors49

LIST OF TABLES

Table 5.1. Simulation parameters 37

Table 5.2. Average number of sensors selected at each time step of tracking, the total number of NSGA-II generations used to get the Pareto-optimal front, and different total number of sensors in the WSN 45



LIST OF SYMBOLS/ABBREVIATIONS

B	Control-input model
COV(X)	Covariance matrix of vector random variable X
$E[]$	Expected value
F	Front list in NSGA-II
F	State transition model
H_t	Model of observation
I	Time interval
I	Identity matrix
K_t	Kalman gain matrix
N	Total number of sensors
N_t	Observation noise in 1-D Kalman Filter
N_{pop}	Population number
P	Covariance matrix in Kalman Filters
P_t	Initial population in NSGA-II
P_0	Signal power of the target
R	Covariance matrix of observation noise
R_t	Newly created list in NSGA-II
R_x	Autocorrelation
R_x	Correlation matrix
S_t	Innovation covariance
S_p	Set of solutions
S_x	Set of random variables
Q_t	Offspring solution
Q	Covariance matrix
X	Random variable
X_n	Observations of a signal
Y	Estimate of observations
Z	Function of random variable

c	Offspring solution by crossover operators
Δ	Target sampling interval
d	Distance of a sensor to the target
e	Mean square error
g_i	Euclidean distance
k_n	Kalman gain
m_x	Mean of random variable x
\mathbf{m}_x	Mean of vector random variable
n	Measurement noise
n_p	Domination count
p	Solutions in NSGA-II
$p_x(x)$	Probability of random variable x
τ	Process noise parameter
σ^2	Noise variance
\mathbf{v}_t	Process noise in system model
\mathbf{u}	Control vector
\mathbf{v}_t	Observation noise in Kalman Filter
w	Pseudo-weight
\mathbf{w}_t	Process noise in Kalman Filter
\mathbf{w}_t	Measurement noise in system model
$w_{i,t}$	Element of measurement noise in system model
x_t, y_t	Target locations
\mathbf{x}	True signal in Kalman Filter
\dot{x}_t, \dot{y}_t	Target velocities
\hat{x}_t	Estimation of random variable
$\hat{\mathbf{x}}_{t t-1}$	Priori state in Kalman Filter
$\hat{\mathbf{x}}_{t t}$	Posteriori state in Kalman Filter
$\hat{\mathbf{y}}_t$	Innovation
z	Received signal at sensor
$z_{i,t}$	Element of received signal at sensor

EKF	Extended Kalman Filter
EMO	Evolutionary multi-objective optimization
FC	Fusion center
FI	Fisher information
FI-SS	Fisher information selection scheme
GD	Generational distance metric
MI	Mutual information
MI-SS	Mutual information selection scheme
MIUB	Mutual information upper bound
MIUB-SS	Mutual information upper bound selection scheme
MMSE	Minimum mean square error
MOO	Multi-objective optimization
MOP	Multi-objective optimization problem
MSE	Mean square error
NSGA-II	Non-dominated sorting genetic algorithm-II
ROI	Region of interest
WSN	Wireless sensor network

1. INTRODUCTION

A typical wireless sensor network (WSN) consists of a large number of sensors where each sensor is assumed to be a simple battery-powered device with some limited signal processing capabilities. If sensors are well programmable and have sufficient connections, a WSN is useful for variant applications such as battlefield surveillance , target tracking and environmental monitoring , industrial applications and health monitoring [1–5] .

Limited signal processing and energy capacity constraints in a WSN lead to the development of adaptive management of sensor strategies. Terms such as sensor selection and bit allocation are examples in adaptive sensor management, and these methods provide the most effective ways of obtaining results by retrieving data from the most informative sensors rather than retrieving data from all sensors in the WSN. For sensor selection, only the most informative sensors transmit their data and the others stay silent not only keeps the estimation error small, but also decreases the number of transmissions and energy spent on each transmission. Transmission bandwidth can be useful for providing better estimation performance for the selection of sensors [6]. In this thesis, we assume that all the WSN's sensors receive measurements from an energy emitting target and send their data to the Fusion Center (FC). FC is not only responsible for statistical inference based on received sensor measurements, but also responsible for adaptive sensor management, i.e., it decides the informative sensors and gathers their data in the next query. In the bit allocation problem, total number of bits in a WSN is limited and FC has another object that addition to the selecting sensors, it distributes the available bits dynamically and optimally among the selected sensors. An example of target tracking application in a WSN is given in Fig. 5.1, where sensors are deployed randomly in a given region of interest (ROI).

Target tracking can be performed by using various algorithms under different situations. The non-linear version of KF, Extended Kalman Filter (EKF), is an algorithm that estimates the unknown states by set of measurements observed over time with considering additive Gaussian noise [7]. In the original algorithm, EKF predicts the location of the target when all sensors transmit their complete measurements. On the other hand, if we force the columns

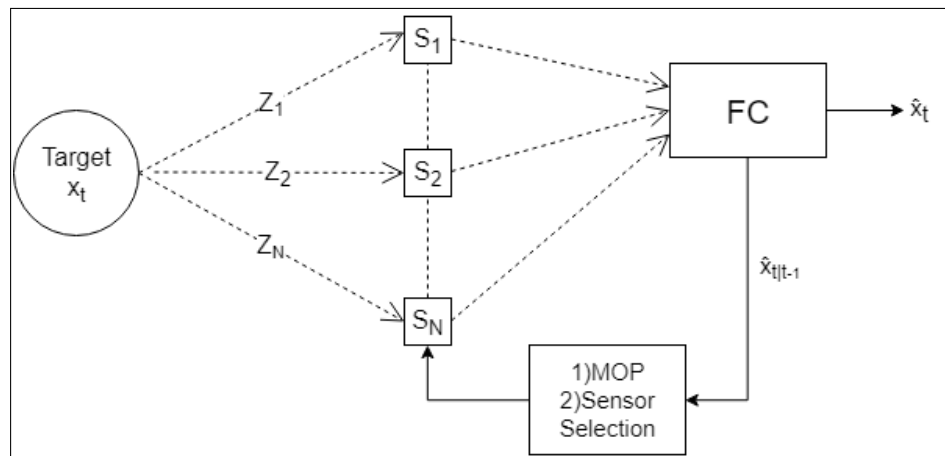


Figure 1.1. Diagram of the target tracking process

of the "Kalman Gain Matrix" to be all zero vectors, then the most informative sensors can be further determined [8]. In order to better handle the non-linear process and measurement models of target tracking, particle filters can be also used instead of using an EKF [9]. Using a particle filter, the performance of target tracking with a pre-specified number of selected sensors was demonstrated for both analog and quantized sensor measurements [10, 11].

Rather than setting a fixed number of sensors selected at each time step of tracking, we can let the system to determine the number of informative sensors automatically. To do so, we can use Multi-objective Optimization (MOO), where the objectives are typically the minimization of error in estimation and minimization of resources used for transmissions [12–20]. The objectives are typically conflicting to each other and the result of the MOO yields us a trade-off curve between the objective functions.

Non-dominated Sorting Genetic Algorithm-II (NSGA-II) is a multi-objective evolutionary algorithm and typically used when the analytical result of the MOO is hard to obtain [21]. NSGA-II is an elitist method, which promotes the solutions dominating the others and yields the pareto-optimal front. NSGA-II was used for sensor selection in target tracking with a specified number of population and fixed number of generations [19, 20]. On the other hand, NSGA-II parameters may affect the results. As an example, a large number of population size may cause the process to slow down, but reduce the error in estimation. In this study,

we demonstrate several methods that form the population that is used at the beginning of the algorithm. Using such adaptations, we can both reduce estimation error and save processing time. Once we have the trade-off curve between the conflicting objectives, we have to select a solution that is employed for the next step of tracking.

1.1. LITERATURE SURVEY

Sensors are typically located in a wide region of interest to observe a phenomenon of interest. Since the WSN resources such as their energy and transmission bandwidth are assumed to be limited, it may bring challenges such as node failure, limited lifetime, or limited node coverage. In a given ROI, it is not desired to gather all sensor measurements, specifically from the uninformative nodes which are far away from the phenomenon of interest. In other words, if uninformative sensors are used for transmission, they consume energy, even if they do not have a significant contribution on the error in estimation. In order to handle such an issue, sensor selection methods have been developed, where the aim is to find the most informative sensor set [22]. The selection of most informative sensors reducing the estimation error and provide the necessary information were discussed [10, 11, 23–31]. The methods considering the mutual information (MI) or entropy were addressed where the aim is to determine the sensor set maximizing the mutual information between the target of interest and sensor measurements [24–27]. In Fisher Information (FI) based methods, the aim is to minimize the error in estimation by maximizing the FI between the target of interest and sensor measurements [10, 11]. Maximizing the FI also minimizes the posterior Cramer-Rao lower bound (PCRLB) on estimation, where PCRLB is basically the trace of the inverse Fisher Information (FI) matrix. Based on quantized sensor measurements, MI and PCRLB are two main methods to be used for sensor selection and were compared [28]. It was shown that, under perfect transmission channels, with using PCRLB, the results have similar mean square error (MSE) with the ones observed under MI and more importantly PCRLB had less mathematical complexity than that of MI [28].

The overall Fisher Information matrix at each time step of tracking can be decomposed into sums of Fisher Information corresponding to each individual sensors and the Fisher Information of the prior time step [6]. Then, if the number of sensors that need to be

selected is known in advance, convex optimization can be used to obtain the best sensor selection strategy [23]. The sensor selection problem was first formulated as an integer programming problem and then relaxed by replacing the binary variables with their continuous counterparts [23]. In such approach the number of selected sensors were given in advance. In previous studies, it is necessary to know in advance how many sensors that need to be selected in each time step of tracking [10, 11, 23, 28]. However, in practice, it is usually not known to the end user how many sensors that need to be selected at each time step of tracking, since the number of informative sensors may vary in time. As an example, when the target passes through a densely deployed area, more sensors become informative, and when the target passes through an area with few sensors, a limited number of sensors may become informative [19].

In previous studies on sensor management, it was assumed that the sensors were not subjected to interference. However, sensor measurements may be uncertain in some situations [32–36]. In some cases, some sensors may not work effectively temporarily, sudden changes in environmental influences may affect the measurements or interruptions, such as the passage of living beings or objects, may affect the information received by the sensors [34, 35]. In addition, random interference may enter the communication channel, or a jammer may be used to disrupt the waves by the enemy [36]. Due to such uncertainties, there might be errors on sensor measurements, which may further adversely affect the statistical inference in the FC. In other words, in such uncertain conditions, sensor readings may only include actual measurement about the target with a certain probability. For these reasons, it is also crucial to study the selection of sensors in an uncertain environment as well. There are some studies on such a situation where the sensors cannot correctly locate the target due to a barrier. Considering this type of uncertainty, a stochastic model for sensor measurements was shown [32, 33]. In addition, it was assumed that the sensors have different uncertainties at different time steps and the same model was generalized [34, 35]. There are studies in the literature using Kalman filter for uncertain WSN cases [33, 37–39]. In addition, studies were done for target localization in communication channels which are not ideal [40, 41]. For target tracking in a WSN, the study of sensor selection in an uncertain measurement environment using multi-objective optimization is further studied [20].

WSN design involves simultaneous consideration of multiple and usually conflicting objectives, such as minimizing the error in estimation, minimization of transmission power, or lifetime maximization of the WSN [13–20, 42]. Such circumstances have been formulated as a Multiobjective Optimization Problem (MOPs) where the solutions of the MOP reflect different trade-offs between several objectives. Mobile agent routing problem was solved by formulating a MOP where the objectives are selected as minimization of energy consumption, minimization of path loss, and maximization of total detected signal energy [15]. Distributed detection problem was formulated as a MOP that the sensor decision thresholds were used as decision variables which jointly determine the probability of decision error and the overall energy consumption of the WSN [17]. The binary quantizer designer problem for detection in this work was then extended to target tracking [18]. Assuming that sensors transmit quantized measurements over ideal channels, the MOP found the sensor selection strategy which jointly minimizes the error in estimation by minimizing the FI gap and minimize the number of selected sensors [19]. The authors extended their previous results in addition to the above situation, the uncertainty in WSN is added and the selection problem is tried to be solved under uncertainty on sensor measurements [20]. The error in estimation was sought to be minimized by using two alternative metrics namely FI selection scheme (FI-SS) and MI selection scheme (MI-SS). Under measurement uncertainties, FI-SS selected sensors close to the target, while MI-SS selected sensors with high sensing probability. Although MI-SS was more successful in performance, it introduced higher computational complexity. A simplified version of MI-SS, that is MI upper bound (MIUB)'s complexity is the same as FI but yielded near optimal results as in MI-SS [20]. Moreover, the MOP problem was solved using Non-dominating Sorting Genetic Algorithm-II (NSGA-II), which is a multiobjective evolutionary algorithm [20, 21]. Other selection methods, e.g., convex optimization and weighted sum method were also used in conjunction with NSGA-II, and the results obtained from all methods were compared.

Evolutionary multi-objective optimization (EMO) is the general name of the methods that maximize or minimize the objective functions with respect to users need. NSGA-II is one of the EMO algorithm. EMO methods have been used in many applications since 1993, when it was developed. Examples can be found in the related books and conferences [43–49]. Many different crossover operators are used to implement the NSGA-II algorithm [50]. There are metrics that can be used to set the number of generations created in the NSGA-II algorithm

such as generational distance, spread and domination metrics [51].

Upon having the all the solutions on the Pareto-optimal front, there are two main methods to determine the solution best reflecting the trade-offs between the objective functions. The first one is the calculation of the pseudo-weight of the related solution [52]. The second one is to select the point closest to the utopia point (the point corresponding to the minimum values of all objectives) of the problem [20]. The performance of different multiobjective evolutionary algorithms can be defined based on metrics mentioned.

1.2. LIST OF CONTRIBUTIONS

In this thesis, we study sensor selection for target tracking in a WSN based on an Extended Kalman Filter similar to the framework presented before [8]. The sensor selection problem is a combinatorial problem and it is hard to solve by using brute force search [23]. In literature, the MOP problem was solved by turning the bi-objective problem into a single objective problem by defining appropriate weight coefficients to each objective functions [8]. The single objective problem was first relaxed and then solved by Alternating Directions Method of Multipliers Method (ADMM). On the other hand, in a practical scenario such weights per each objective might not be clear to the user in advance.

In this study, we formulate a MOP problem for sensor selection similar to the ones presented in related works [19,20]. In previous works, the authors employed particle filters and used Fisher Information at each time step of tracking [19,20]. However in our work, we employ Extended Kalman Filter and used trace of posterior error covariance matrix to quantify the estimation error in the first objective. The second objective is to minimize the total number of sensors selected at each time step of tracking. Our results can be directly extended to the particle filtering scenario given in previous studies [19,20]. In these, at each time step of tracking, the initial population was determined randomly and independent from the final population of the previous time step. Furthermore, at each time step of tracking the NSGA-II generations were executed $G = 100$ times, where NSGA-II might have converged earlier. Among the solutions on the Pareto-Optimal front, the solution for sensor selection was determined based on the knee-point solution and the compromise solution.

The list of contributions of this thesis as compared to the previous literature can be listed as follows:

- Using an Extended Kalman Filter framework, we force the columns of the "Kalman Gain Matrix" to be all zero vectors, then the non-zero columns of the Kalman Gain Matrix represents the sensors to be selected at each time step of tracking. We define a binary vector of sensor selections whose size is equal to the total number of sensors in the WSN. If the vector element is one then the corresponding column of Kalman Gain Matrix becomes non-zero, and the sensor transmits its measurement; otherwise the corresponding column of Kalman Gain Matrix becomes zero and the sensor stays silent. The binary vector of sensor selections is then obtained using NSGA-II algorithm where the objectives are minimization the error in estimation by minimizing the trace of the posterior error Covariance matrix and minimization the total number of sensors transmitting at each time step of tracking.
- Upon completing the NSGA-II generations, we select the sensor selection vector on the Pareto-optimal front based on knee-point solution and minimum distance solutions as well as the Pseudo-Weight solution [20, 52].
- Rather than initializing NSGA-II with totally randomly generated populations at each time step of tracking, we propose to use two alternative initialization populations as in Dynamic NSGA-II [52]. In the first case, we use the final population of the previous time step entirely as the initial population of the current time step. In the second case, we first fill the initial population of the current time step with the unique final solutions of the previous time step, rest of the solutions in the population are then determined randomly.
- In previous works, for each execution of NSGA-II, the algorithm was executed for an excessive predetermined number of generations, where it was quite possible that NSGA-II was converged in an earlier generation [19, 20]. In order to terminate the algorithm in an earlier generation, we utilize the generational distance as a stopping metric. We present the performance of the stopping metric under different population sizes.
- We also compare the performance of NSGA-II algorithm under two different crossover operators, namely uniform crossover and simple crossover.

We tested the performance of sensor selection for single target tracking under target trajectories with different process noise parameters, and different number of sensors in the WSN. Our numerical results show that rather than achieving the Pareto-optimal front after excessive generations generational distance metric can be used as a good metric for stopping as long as the population size is relatively large. Furthermore rather than initializing the NSGA-II iterations with random solutions, the final unique solutions of the previous time step can be added to the initial population of the current time step to help NSGA-II terminate at an earlier generation.

1.3. THESIS ORGANISATION

The organisation of this thesis is summarized as given below,

In Chapter 2, we present the theoretical foundations of this work in detail. Firstly, we present the definitions of expected value, correlation and covariance matrices of random variables. In addition, we clarify the definition of minimum square error (MSE) that determines the target tracking performance in the Kalman filter. Using the previous definitions, we next give information about how optimal filters are formed, and explain the orthogonality condition. Such definitions form the basis of Kalman filters. After explaining the basic equations and terms, we introduce 1-D Kalman filter. In addition, we also explain the Vector Kalman filter and the Extended Kalman filter used in our target tracking application.

In Chapter 3, we describe our WSN system model and the target tracking problem that we are trying to solve. The description of the variables and matrices used in EKF and how the sensors make the measurements are explained in detail. In this application, we show that the Kalman gain matrix is modified by considering the selected sensors.

In Chapter 4, we explain that optimum sensor selection is performed by using NSGA-II. We clarify all the steps of the NSGA-II algorithm in detail. We describe the types of populations and how to create initial and offspring populations accordingly. When producing populations, we describe tournament selection, non-domination sorting, crowding distance methods. We define the two methods used in decision making, pseudo-weight calculation, minimum

distance to origin. We further introduce the crossover and mutation operators, as well as the generational distance metric which is used as a stopping rule.

In Chapter 5, we present our numerical results. The algorithms developed throughout the thesis describe what kind of changes depend on which variables. In the target tracking application, how the NSGA-II algorithm is made more effective and faster, what criteria it depends and the comparisons between them are explained.

In Chapter 6, we summarize the applications made in the thesis, their results and benefits. In addition, we give advice on what can be done in the future to develop more advanced algorithms for this application.

2. PRELIMINARIES

This chapter describes the applications of Kalman filter and how it works. Some equations and values need to be defined before starting to describe the Kalman filter. Characterization of a vector random variable can be done by using Expected values. It is done by mean vector and covariance matrix.

2.1. EXPECTED VALUES, CORRELATION & COVARIANCE MATRICES OF RANDOM VARIABLES

Expected value refers to mean of a random variables x that occur from an event X and its formula is given below,

$$m_x = E[X] = \sum_{x \in S_x} xp_x(x) = \sum_k x_k p_x(x_k) \quad (2.1)$$

where m_x is mean of x , the set S_x of random variables that x can be, $p_x(x_k)$ is the probability of a random event X equals to random variable x_k . If the random variable is a function that $Z = g(x)$, then it's expected value or mean becomes,

$$E[Z] = E[g(X)] = \sum_k g(x_k)p_x(x_k) \quad (2.2)$$

where Z or $g(X)$ is a function of random variable. Variance value can be found as follows,

$$\begin{aligned} \sigma_x^2 = VAR[X] &= E[(X - m_x)^2] \\ &= E[X^2] - m_x^2 \end{aligned} \quad (2.3)$$

These values give us information about how the random variable oscillates around which point. If there are more than one random variables the formula changes. Expected values of a

function of two discrete random variables are found by,

$$E[X^j Y^k] = \sum_i \sum_n x_i^j y_n^k p_{X,Y}(x_i, y_n) \quad (2.4)$$

where $p_{X,Y}(x_i, y_n)$ is the joint probability of x and y . The relationship of two random variables is determined by covariance. If they are independent, then their covariance will be zero but the opposite is not always true. According to this, we cannot say they are independent if their covariance is zero for every situation. Covariance formula is given below,

$$\begin{aligned} COV(X, Y) &= E[(X - E[X])(Y - E[Y])] \\ &= E[XY] - E[X]E[Y] \end{aligned} \quad (2.5)$$

If random variables are vector, their expected values are shown by,

$$\mathbf{m}_x = E[\mathbf{X}] = E \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} E[X_1] \\ E[X_2] \\ \vdots \\ E[X_n] \end{bmatrix} \quad (2.6)$$

The relationship between the elements of the vector is shown by the correlation matrix which is given below,

$$\mathbf{R}_x = \begin{bmatrix} E[X_1^2] & E[X_1 X_2] & \cdots & E[X_1 X_n] \\ E[X_2 X_1] & E[X_2^2] & \cdots & E[X_2 X_n] \\ \vdots & \vdots & \ddots & \vdots \\ E[X_n X_1] & E[X_n X_2] & \cdots & E[X_n^2] \end{bmatrix}$$

$$\mathbf{R}_x = E[\mathbf{X}\mathbf{X}^T] \quad (2.7)$$

If the elements are independent to each other, \mathbf{R}_x 's diagonal elements will be zero. There is also one more matrix definition to understand the Kalman filter. It is called Covariance matrix which includes expected values of units in vectors minus their means. The matrix is written by,

$$\mathbf{COV}(\mathbf{X}) = \begin{bmatrix} E[(X_1 - m_1)^2] & E[(X_1 - m_1)(X_2 - m_2)] & \cdots & E[(X_1 - m_1)(X_n - m_n)] \\ E[(X_2 - m_2)(X_1 - m_1)] & E[(X_2 - m_2)^2] & \cdots & E[(X_2 - m_2)(X_n - m_n)] \\ \vdots & \vdots & \ddots & \vdots \\ E[(X_n - m_n)(X_1 - m_1)] & E[(X_n - m_n)(X_2 - m_2)] & \cdots & E[(X_n - m_n)^2] \end{bmatrix}$$

$$\mathbf{COV}(\mathbf{X}) = \mathbf{R}_x - \mathbf{m}_x \mathbf{m}_x^T \quad (2.8)$$

As seen in (2.8), covariance matrix's diagonal elements are variances of the elements that related with the row.

2.1.1. Minimum Mean Square Error

Minimum mean square error is used for developing Kalman Filter. Estimator of a random variable X is $\hat{x} = g(Y)$ where Y is the observations, then mean square error (MSE) becomes,

$$e = E[(X - g(Y))^2] \quad (2.9)$$

If Y is a vector of observations that $Y = (Y_1, Y_2, \dots, Y_n)^T$ and $g(Y)$ is estimation vector of observations X , then it can be written as

$$\hat{x} = g(Y) = \sum_{k=1}^n w_k Y_k \quad (2.10)$$

where w_k set of constants. For this situation, finding $g(Y)$ value that minimizes error gives us the value of the best estimation of X .

2.2. OPTIMUM LINEAR SYSTEMS & ORTHOGONALITY CONDITION

A discrete time random process $X(t)$ must be wide-sense stationary(WSS) to be applied to the optimum linear filter. The rule of that is random process' mean is constant for all t and autocorrelation function depends only on $t_1 - t_2$ which equals τ . Mean and autocorrelation of $X(t)$ formulas given below,

$$m_x(t) = E[X(t)] = c \quad (2.11)$$

$$R_X(t_1 - t_2) = E[X(t_1)X(t_2)] \quad (2.12)$$

$$R_X(\tau) = E[X(t)X(t + \tau)] \quad (2.13)$$

$$R_X(t_1, t_2) = E[X(t_1)X(t_2)] = R_X(\tau) \quad (2.14)$$

where c is a constant. The structure for an optimal linear filter is shown in Figure 2.1. A zero mean, discrete time random process X_α is observed in time interval $I = t - a, \dots, t + b$. $a + b + 1$ resulting every element of the observation $X_{t-a}, X_{t-a+1}, \dots, X_{t+b}$ is multiplied by appropriate constants h_a, h_{a-1}, \dots, h_b and their summation Y_n will be our estimation for zero mean process Z_t which is shown by,

$$Y_t = \sum_{\beta=t-a}^{t+b} h_{t-\beta} X_\beta = \sum_{\beta=-b}^a h_\beta X_{t-\beta} \quad (2.15)$$

Mean square error equals to,

$$E[e_t^2] = E[(Z_t - Y_t)^2] \quad (2.16)$$

To minimize the mean square error, optimum h_β have to be found. The error e_t is orthogonal to all X_α if the filter is optimum filter. The rule is named as orthogonality condition.

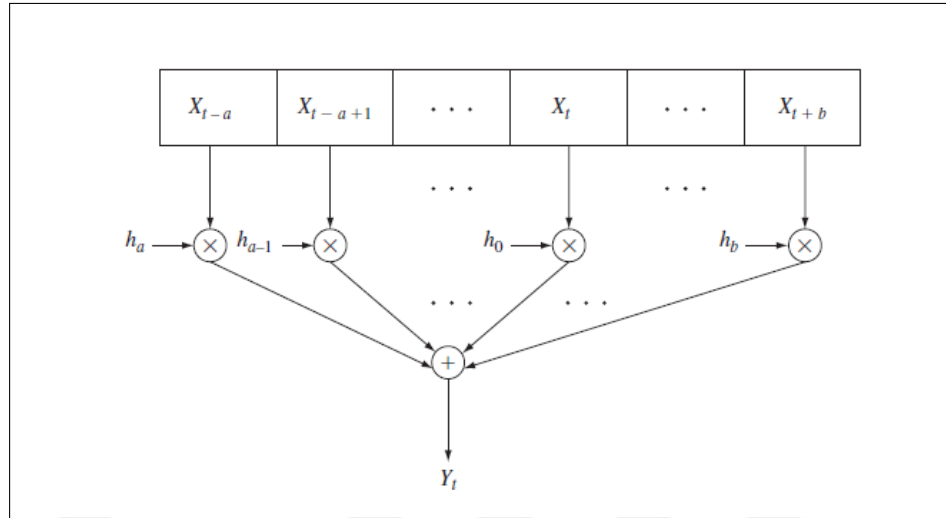


Figure 2.1. Optimal linear filter structure [53]

Orthogonality condition where $\alpha \in I$ and new equation is given below,

$$E[e_t X_\alpha] = 0$$

$$E[(Z_t - Y_t) X_\alpha] = 0$$

$$E[Z_t X_\alpha] = E[Y_t X_\alpha] \quad (2.17)$$

Substituting (2.15) and (2.2) we get,

$$E[Z_t X_\alpha] = E \left[\sum_{\beta=-b}^a h_\beta X_{t-\beta} X_\alpha \right]$$

$$= \sum_{\beta=-b}^a h_\beta E[X_{t-\beta} X_\alpha]$$

$$E[Z_t X_\alpha] = \sum_{\beta=-b}^a h_\beta R_x(t - \alpha - \beta) \quad (2.18)$$

As seen on (2.2), $E[Z_t X_\alpha]$ depends only on $t - \alpha$ that means these two random processes are jointly WSS processes [53]. Thus the equation can be written,

$$R_{Z,X}(t - \alpha) = \sum_{\beta=-b}^a h_\beta R_x(t - \alpha - \beta) \quad t - a \leq \alpha \leq t + b$$

If we let $m = t - \alpha$, then we get,

$$R_{Z,X}(m) = \sum_{\beta=-b}^a h_\beta R_x(m - \beta) \quad -b \leq m \leq a \quad (2.19)$$

Thus, the filter which minimizes the mean square error $E[(Z_t - Y_t)^2]$ must satisfy (2.19). Additionally, to determine MSE, we will benefit from one more orthogonal condition which includes the error e_t and the estimate Y_t by,

$$E[e_t Y_t] = 0 \quad (2.20)$$

Then MSE becomes,

$$E[e_t^2] = E[e_t(Z_t - Y_t)] = E[e_t Z_t] - E[e_t Y_t]$$

$$E[e_t^2] = E[e_t Z_t] \quad (2.21)$$

While we continue substituting e_t , we get,

$$\begin{aligned}
 E[e_t^2] &= E[e_t Z_t] = E[(Z_t - Y_t)Z_t] = E[Z_t Z_t] - E[Y_t Z_t] \\
 &= R_z(0) - E[Z_t Y_t] \\
 &= R_z(0) - E \left[Z_t \sum_{\beta=-b}^a h_\beta X_{t-\beta} \right] \\
 E[e_t^2] &= R_z(0) - \sum_{\beta=-b}^a h_\beta R_{Z,X}(\beta) \tag{2.22}
 \end{aligned}$$

2.3. 1-D KALMAN FILTER

Kalman filter is an algorithm that estimates unknown variables by set of measurements observed over time with considering noise. The objective is finding MSE of the true signal Z_t for every time step, with considering observations X_0, X_1, \dots, X_{t-1} . t is the time step here. The linear estimator is given below,

$$Y_t = \sum_{j=1}^t h_j^{(t-1)} X_{t-j} \tag{2.23}$$

Z_t is assumed to be unknown and it's structure is shown in Figure 2.2. X_t is the observation signal and is generated by adding observation noise N_t to the actual signal. N_t has zero mean and time-varying variances $E[N_t^2]$. There is also another noise type called process noise W_t which also has zero mean and time-varying variances $E[W_t^2]$. Finally, a_t is a known set of constants.

As seen in the structure signal model as obtained as,

$$Z_t = a_{t-1} Z_{t-1} + W_{t-1} \tag{2.24}$$

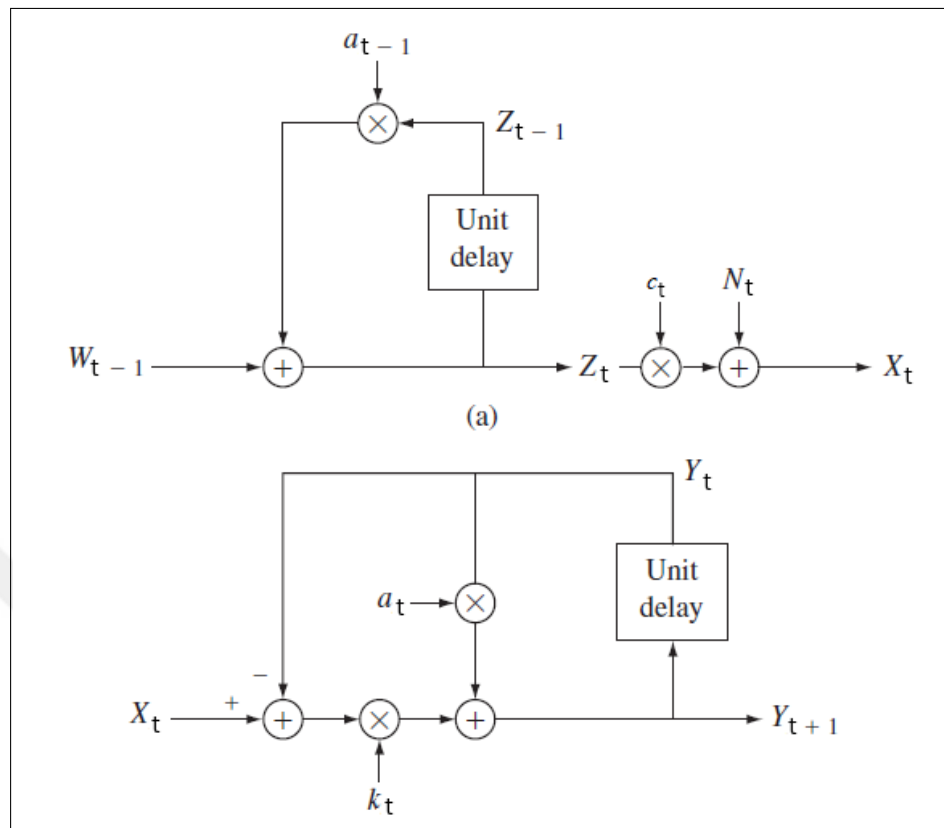


Figure 2.2. 1-D Kalman Filter signal structure [53]

$$X_t = c_t Z_t + N_t \quad (2.25)$$

where c_t is a known set of constants. We apply orthogonality rule to the this filter,

$$E[(Z_t - Y_t)X_l] = 0 \quad l = 0, 1, \dots, t - 1$$

$$E[Z_t X_l] = E[Y_t X_l]$$

$$R_{Z,X}(t, l) = E \left[\sum_{j=1}^t h_j^{(t-1)} R_X(t-j, l) \right] \quad (2.26)$$

If we substitute Z_t , $R_{Z,X}(t, l)$ also equals to,

$$Z_t = \frac{X_t - N_t}{c_t}$$

$$\begin{aligned} R_{Z,X}(t, l) &= E \left[\left(\frac{X_t - N_t}{c_t} \right) X_l \right] \\ &= \frac{E[X_t X_l] + E[X_t N_l]}{c_t} \end{aligned} \quad (2.27)$$

X_t and N_l are independent from each other, thus $E[X_t N_l]$ equals to zero and we get,

$$\begin{aligned} &= \frac{E[X_t X_l]}{c_t} = \frac{R_X(t, l)}{c_t} \\ R_{Z,X}(t, l) &= \frac{R_X(t, l)}{c_t} \end{aligned} \quad (2.28)$$

Y_{t+1} , the estimation of Z_{t+1} becomes,

$$Y_{t+1} = \sum_{j=1}^{t+1} h_j^{(t)} X_{t+1-j} \quad (2.29)$$

Now, we apply orthogonality condition for the next time step $t + 1$ with using (2.24),

$$E[(Z_{t+1} - Y_{t+1})X_l] = 0 \quad l = 0, 1, \dots, t - 1$$

$$R_{Z,X}(t + 1, l) = E \left[\sum_{j=1}^{t+1} h_j^{(t)} R_X(t + 1 - j, l) \right]$$

$$R_{Z,X}(t + 1, l) = E[Z_{t+1}X_l] = E[(a_t Z_t + W_t)X_l]$$

$$= a_t E[Z_t X_l] + E[W_t X_l] = a_t E[Z_t X_l]$$

$$R_{Z,X}(t + 1, l) = a_t E[Z_t X_l]$$

$$R_{Z,X}(t + 1, l) = a_t R_{Z,X}(t, l) \quad (2.30)$$

If we substitute (2.28) with (2.3),

$$a_t R_{Z,X}(t, l) = h_1^{(t)} c_t R_{Z,X}(t, l) + \sum_{j=2}^{t+1} h_j^{(t)} R_X(t + 1 - j, l) \quad (2.31)$$

$$R_{Z,X}(t, l) = \sum_{j=1}^t \left(\frac{h_{j+1}^{(t)}}{a_t - c_t h_1^{(t)}} \right) R_X(t - j, l) \quad (2.32)$$

Equating (2.31) with (2.3), we get,

$$h_{j+1}^{(t)} = (a_t - c_t h_1^{(t)}) h_j^{(t-1)} \quad (2.33)$$

After mathematical substitutions are done to (2.29) and (2.33), we get another equation where Y_{t+1} is equal. Y_{t+1} equation also equals to,

$$Y_{t+1} = a_t Y_t + h_1^{(t)} (X_t - c_t Y_t) \quad (2.34)$$

According to these, optimum $h_1^{(t)}$, later called k_t which refers to "Kalman gain" has to be found for every time step. With using the prediction error $\varepsilon_t = (Z_t - Y_t)$, next time step's prediction error becomes with substituting (2.34) with (2.24),

$$\varepsilon_{t+1} = (a_t - k_t c_t) \varepsilon_t + W_t - k_t N_t \quad (2.35)$$

Thus, mean square prediction error $E[\varepsilon_{t+1}^2]$ is equal to,

$$E[\varepsilon_{t+1}^2] = a_t(a_t - c_t k_t) E[\varepsilon_t^2] + k_t^2 E[N_t^2] \quad (2.36)$$

We have to find optimum k_t that minimizes the MSE, this method is provided by taking the derivative of the equation and then equalizing to 0. Finally k_t equation is found as,

$$k_t = \frac{c_t a_t E[\varepsilon_t^2]}{c_t^2 E[\varepsilon_t^2] + E[N_t^2]} \quad (2.37)$$

2.4. VECTOR KALMAN FILTER

A number of changes in the use of kalman filter with vectors are based on the current prior situation. If we show time step as t , we will accept \mathbf{x}_t as true signal and \mathbf{z}_t as observation signal in current equations. In the Kalman filter, it is supposed that true state signal \mathbf{x}_t is developed by the previous state signal \mathbf{x}_{t-1} by the formula below,

$$\mathbf{x}_t = \mathbf{F}_t \mathbf{x}_{t-1} + \mathbf{B}_t \mathbf{u}_t + \mathbf{w}_t \quad (2.38)$$

where \mathbf{F}_t refers to state transition model, \mathbf{B}_t is the control-input model which is multiplied by control vector \mathbf{u}_t , \mathbf{w}_t is process noise like before and assumed to has zero mean gaussian with the covariance matrix \mathbf{Q}_t . Observation signal's model is given as,

$$\mathbf{z}_t = \mathbf{H}_t \mathbf{x}_t + \mathbf{v}_t \quad (2.39)$$

where \mathbf{H}_t is the model of observation that transfers true state space to observed space and \mathbf{v}_t is the observation noise has zero mean, gaussian with covariance \mathbf{R}_t . The new state is got with using the previous estimation and the current measurements. Kalman filter has 2 steps which are "priori" or "predict" and "posteriori" or "update". Priori state estimations are notated as $\hat{\mathbf{x}}_{t|t-1}$ and update states as $\hat{\mathbf{x}}_{t|t}$. $\mathbf{P}_{t|t-1}$ and $\mathbf{P}_{t|t}$ are covariance matrices that reflect the covariance of estimates in priori and posteriori state, respectively as seen below,

$$\mathbf{P}_{t|t} = \text{cov}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t}) \quad (2.40)$$

$$\mathbf{P}_{t|t-1} = \text{cov}(\mathbf{x}_t - \hat{\mathbf{x}}_{t|t-1}) \quad (2.41)$$

Prediction state consists of 2 equations given by,

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F}_t \hat{\mathbf{x}}_{t-1|t-1} + \mathbf{B}_t \mathbf{u}_{t-1} \quad (2.42)$$

$$\mathbf{P}_{t|t-1} = \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}_t^T + \mathbf{Q}_t \quad (2.43)$$

$\hat{\mathbf{y}}_t$ is defined as innovation or measurement pre-fit residual that obtained by,

$$\hat{\mathbf{y}}_t = \mathbf{z}_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t-1} \quad (2.44)$$

There is also one more covariance matrix \mathbf{S}_t which is innovation covariance that equals to $\text{cov}(\hat{\mathbf{y}}_t)$. Then the update state without innovation becomes,

$$\mathbf{S}_t = \mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t \quad (2.45)$$

$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{S}_t^{-1} \quad (2.46)$$

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \hat{\mathbf{y}}_t \quad (2.47)$$

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbf{P}_{t|t-1} \quad (2.48)$$

$$\hat{\mathbf{y}}_{t|t} = \mathbf{z}_t - \mathbf{H}_t \hat{\mathbf{x}}_{t|t} \quad (2.49)$$

where \mathbf{K}_t is the Optimal Kalman gain that minimizes the residual error and $\hat{\mathbf{y}}_{t|t}$ is the measurement post-fit residual. Substitutions on (2.40) gives another equation of $\mathbf{P}_{t|t}$. Derivation of \mathbf{K}_t is gotten from minimizing MSE that equals to $E[\mathbf{x}_t - \hat{\mathbf{x}}_{t|t}]$. This is also equals to minimize the trace (a square matrix is defined to be the sum of diagonal elements) of $\mathbf{P}_{t|t}$. The related equations are given below,

$$\begin{aligned} \mathbf{P}_{t|t} &= \mathbf{P}_{t|t-1} - \mathbf{K}_t \mathbf{H}_t \mathbf{P}_{t|t-1} - \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{K}_t^T + \mathbf{K}_t (\mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t) \mathbf{K}_t^T \\ &= \mathbf{P}_{t|t-1} - \mathbf{K}_t \mathbf{H}_t \mathbf{P}_{t|t-1} - \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{K}_t^T + \mathbf{K}_t \mathbf{S}_t \mathbf{K}_t^T \end{aligned} \quad (2.50)$$

$$\frac{\partial \text{tr}(\mathbf{P}_{t|t})}{\partial \mathbf{K}_t} = -2(\mathbf{H}_t \mathbf{P}_{t|t-1})^T + 2\mathbf{K}_t \mathbf{S}_t = 0 \quad (2.51)$$

$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{S}_t^{-1}$$

After finding the current states, the filter moves to the next time step and continues iterative operations, so that the next states are found using the previous ones. Finally, this filter applies to linear systems, and for non-linear ones we will move to the next section, EKF.

2.5. EXTENDED KALMAN FILTER

The extended kalman filter is a more advanced version of the kalman filter, designed for nonlinear situations. \mathbf{x}_t and \mathbf{z}_t models become differentiable functions. The models are given by,

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{u}_t) + \mathbf{w}_t \quad (2.52)$$

$$\mathbf{z}_t = h(\mathbf{x}_t) + \mathbf{v}_t \quad (2.53)$$

The function can vary according to the system, the model we use will be mentioned later. The functions h and f are used to find predict states as before, but the difference here they don't directly affect covariance, the partial derivatives of them need to be calculated. Prediction state equations are given below,

$$\hat{\mathbf{x}}_{t|t-1} = f(\hat{\mathbf{x}}_{t-1|t-1}, \mathbf{u}_t) \quad (2.54)$$

$$\mathbf{P}_{t|t-1} = \mathbf{F}_t \mathbf{P}_{t-1|t-1} \mathbf{F}_t^T + \mathbf{Q}_t \quad (2.55)$$

And the update state formulas are given below,

$$\hat{\mathbf{y}}_t = \mathbf{z}_t - h(\hat{\mathbf{x}}_{t|t-1}) \quad (2.56)$$

$$\mathbf{S}_t = \mathbf{H}_t \mathbf{P}_{t|t-1} \mathbf{H}_t^T + \mathbf{R}_t \quad (2.57)$$

$$\mathbf{K}_t = \mathbf{P}_{t|t-1} \mathbf{H}_t^T \mathbf{S}_t^{-1} \quad (2.58)$$

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t \hat{\mathbf{y}}_t \quad (2.59)$$

$$\mathbf{P}_{t|t} = (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t) \mathbf{P}_{t|t-1} \quad (2.60)$$

where \mathbf{F}_t and \mathbf{H}_t are the Jacobian matrices that defined as,

$$\mathbf{F}_t = \frac{\partial f}{\partial x} \Big|_{\hat{\mathbf{x}}_{t-1|t-1}, \mathbf{u}_t} \quad (2.61)$$

$$\mathbf{H}_t = \frac{\partial h}{\partial x} \Big|_{\hat{\mathbf{x}}_{t-1|t-1}, \mathbf{u}_t} \quad (2.62)$$

The values of all these variables and functions will be given in the next section when describing our system model. Thus, it will be clearly stated how extended kalman filter is applied to our system.



3. SYSTEM MODEL

We assume that a moving target emits a signal at each time step t at its location (x_t, y_t) and randomly distributed N sensors measure this signal and attempt to estimate the target's location and velocity (\dot{x}, \dot{y}) [8]. Where the sensors are located does not affect the operation of the system, but only if their locations are known. There are two objectives in this application for us, minimize selected sensors number and MSE which states target tracking error.

The status of the target is indicated by a matrix of 4×1 which is,

$$\mathbf{x}_t = \begin{bmatrix} x_t & y_t & \dot{x}_t & \dot{y}_t \end{bmatrix}^T \quad (3.1)$$

Next time step's state has a formula given below,

$$\mathbf{x}_{t+1} = \mathbf{F}\mathbf{x}_t + \mathbf{v}_t \quad (3.2)$$

where \mathbf{v}_t is the white gaussian process noise signal with zero mean and covariance matrix \mathbf{Q} . \mathbf{Q} and state transition model \mathbf{F} are both matrices with 4×4 sizes. Their details are given below,

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & \Delta & 0 \\ 0 & 1 & 0 & \Delta \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{Q} = \tau \begin{bmatrix} \frac{\Delta^3}{3} & 0 & \frac{\Delta^2}{2} & 0 \\ 0 & \frac{\Delta^3}{3} & 0 & \frac{\Delta^2}{2} \\ \frac{\Delta^2}{2} & 0 & \Delta & 0 \\ 0 & \frac{\Delta^2}{2} & 0 & \Delta \end{bmatrix} \quad (3.3)$$

where τ is the process noise parameter. We assume observation vector as $\mathbf{z}_t \triangleq [z_{1,t}, \dots, z_{N,t}]^T$, which has the equation,

$$\mathbf{z}_t = h(\mathbf{x}_t) + \mathbf{w}_t \quad (3.4)$$

\mathbf{w}_t is the measurement noise simulates the errors in signal parameters' models and background

noise [8]. It is like process noise with white gaussian zero-mean, in contrast, its covariance matrix is $\mathbf{R} = \sigma^2 \mathbf{I}_{N \times N}$. \mathbf{I} represents the identity matrix. The noisy signal amplitudes which are units of z_t detected by the sensors are shown below by the equation of $z_{i,t}$,

$$z_{i,t} = \sqrt{\frac{P_0}{1 + (d_{i,t})^n}} + w_{i,t} \quad (3.5)$$

where P_0 is the power of the signal source, n is the decay exponent and $d_{i,t}$ represent the distance of i^{th} sensor to the target [8]. It is determined that the location of i^{th} sensor is (x_i, y_i) and n is assumed to be 2. Then $d_{i,t}$ becomes,

$$d_{i,t} = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2} \quad (3.6)$$

In the predicted state, priori estimation $\hat{\mathbf{x}}_{t|t-1}$ is found by,

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{F} \hat{\mathbf{x}}_{t-1|t-1} \quad (3.7)$$

For the update step, \mathbf{H}_t equals to derivative of $h(\mathbf{x}_t)$ with respect to $\hat{\mathbf{x}}_{t|t-1}$. Then, $h(\mathbf{x}_t)$ and its derivative \mathbf{H}_t becomes like below,

$$h_i(\mathbf{x}_t) = \sqrt{\frac{P_0}{1 + (d_{i,t})^2}} \quad (3.8)$$

$$\mathbf{H}_t = \begin{bmatrix} \frac{\partial h_1}{\partial x_t} |_{\hat{\mathbf{x}}_{t|t-1}} & \frac{\partial h_1}{\partial y_t} |_{\hat{\mathbf{x}}_{t|t-1}} & 0 & 0 \\ \frac{\partial h_2}{\partial x_t} |_{\hat{\mathbf{x}}_{t|t-1}} & \frac{\partial h_2}{\partial y_t} |_{\hat{\mathbf{x}}_{t|t-1}} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial h_N}{\partial x_t} |_{\hat{\mathbf{x}}_{t|t-1}} & \frac{\partial h_N}{\partial y_t} |_{\hat{\mathbf{x}}_{t|t-1}} & 0 & 0 \end{bmatrix} \quad (3.9)$$

Other equations same with the Extended Kalman Filter which are obtained by (2.56) - (2.60).

3.1. SENSOR SELECTION

Kalman gain \mathbf{K}_t matrix contains 4 rows and N columns which is the total sensor number. Each column contains the data of the related sensor, and if a column is all zero, it means that the sensor associated with the column is in off mode at that time step. Using this, we are able to ensure that the sensors operate as optional.

We will compare the results obtained when all the sensors work together with the results obtained when best 1,2,3,4 and 5 sensors are selected which ensures the minimum error rate. When making these sensor selections, we make a decision by examining the calculated $\mathbf{P}_{t|t}$ at the end of the each time step we will compare. The lower sum of the diagonal elements of the matrix $\mathbf{P}_{t|t}$ means that the error is smaller. For example, let's say we select the best 3 sensors that minimize the error and let these sensors be 1st, 2nd and 3rd sensors. In this case we create a binary matrix of the same size as the \mathbf{K}_t matrix and write 1 to the first, second and third columns which are related selected sensors and write 0 to the others. When we calculate the entrywise product or Hadward product of this matrix with \mathbf{K}_t , the process is continued with the newly formed \mathbf{K}_t^* , thus eliminating unnecessary sensors.

The example of matrix operations are given below,

$$\mathbf{K}_t^* = \mathbf{K}_t \circ \text{selection matrix} \quad (3.10)$$

$$= \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & a_{2,N} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N,1} & a_{N,2} & a_{N,3} & \cdots & a_{N,N} \end{bmatrix} \circ \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

$$= \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & 0 \\ a_{2,1} & a_{2,2} & a_{2,3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{N,1} & a_{N,2} & a_{N,3} & \cdots & 0 \end{bmatrix}$$

After performing all time step estimations, we will use the MSE which is mentioned before to see how is the system's performance. To do that, we will have the help of this equation,

$$MSE(t) \triangleq \frac{1}{T_{trials}} \sum_{c=1}^{T_{trials}} (x_c(t) - \hat{x}_c(t))^2 + (y_c(t) - \hat{y}_c(t))^2 \quad (3.11)$$

The closer the result of MSE to zero, the more successful the system is. In the next chapter we will see details of the NSGA-II algorithm and how it is adapted to EKF(Extended Kalman Filter).

4. NONDOMINATED SORTING GENETIC ALGORITHM-II

Creating an optimal solution list at all time steps is provided by an algorithm. We use a MOO algorithm which is named as NSGA-II. The best sensor selections are saved from all the selection combinations by this algorithm. After that, one of the best results is chosen and how the process is going will be explained in the next sections. In this method, at first, a graph is drawn in which two objective forms the x and y axes. There are two objectives we use and try to minimize, these are the Minimum Square Error(MSE) by minimizing the trace of estimation error covariance matrix and the total number of sensors selected for all combinations.

Let us consider two separate points in this graph, and let them be called A and B. In order for point A to dominate B, at least one of the two points in the x and y axis for A must be lower than B, and the other value must be equal to or lower than the other value of B. The x-axis values will be compared to the x-axis values of the other point, and the y-axis values will be compared to the y-axis values of the other point. For each point or solution p there are two values have to be calculated which are: 1) Domination count(n_p) which is the number of solutions that dominate p solution, 2) A set of solutions(S_p) which includes points that p dominates [21]. Non-dominated solutions(a solution p has $n_p=0$) are selected first and written to first front list (F_1). Then the domination count of the solutions in the S_p list of any p solution is reduced by one. By doing so, we temporarily remove the first front from the population to find the second front(F_2) with using the same method we did on the first front. This operation goes on until all fronts are found. Before making these applications, we need to create the solution population we will use. Let's call P_t to the initial population containing some solutions in one generation.

Population number (N_{pop}) may vary according to our preference. According to this number we stated, N_{pop} solutions are chosen randomly from all of the sensor selection combinations in a time step which include all possibilities about which sensors are open and which are closed. An offspring population Q_t with the same number N_{pop} of individuals is added to this population, and the new list containing these two is called R_t . Additionally to Nondomination

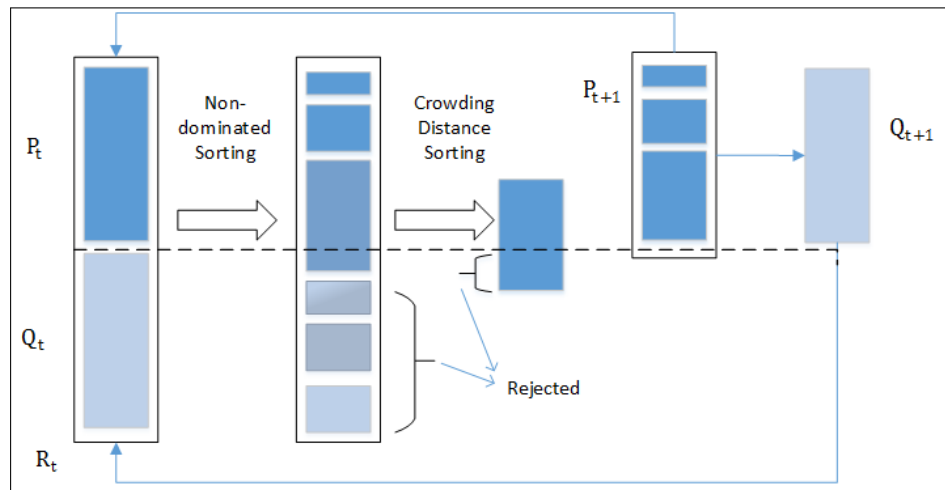


Figure 4.1. A basic flowchart of NSGA-II algorithm [54]

sorting, there is an operator which is called Crowded-Comparison Operator used when creating this Q_t .

Initially, random two individuals are selected from P_t , then this two are compared for their rank. The rank is the same number of the individual's front and lower rank is the first factor affecting the selection. If the ranks are equal, the second operator mentioned will be useful which uses the crowding distance of an individual that can be found by the cuboid formed using the distance of the two closest individuals in the same front to each other on the x and y axis. The side lengths of cuboid should be normalized before the calculations. The crowding distance of a point which has maximum or minimum value corresponding to each objective is considered infinite. For the other individuals, these lengths for each objective, gives us the crowding distance and the sum of individual distance values by each objective becomes the overall crowding distance. An individual which has the overall crowding distance higher which means an individual in a less crowded area will be parent for the next step. A second parent is obtained using the same method. This parent selection method is called as Tournament Selection. Additionally to this, one of the crossover methods which will be explained later and a mutation are used to modify the parents selected and added to Q_t .

Finally, we create R_t which is the population we will use for non domination sorting by combining P_t and Q_t . After determining the fronts and ranks for all individuals, we need

to halve R_t , which has 2 times more individuals than P_t . To do that, those with a better rank (lower rank) will be kept at R_t . After passing the population number N_{pop} , if there are elements in the same front (with the same rank) with the last selected individuals, they will be retained or deleted according to their overall crowding distance. The remaining fronts will be completely removed from the list. The new R_t , which consists of the rest, is called P_{t+1} and is used in a new generation (one cycle of NSGA-II) as an initial population. If the last generation is reached, this last set is called Pareto optimal front and thus the NSGA-II algorithm is finalized. In which generation the application will end depends on the user's preference. In this thesis, how the algorithm ends automatically according to the results of the generations will be explained in the following sections.

4.1. DECISION MAKING WITH PSEUDO-WEIGHT CALCULATION

After finding the optimal solutions in Pareto - Optimal Front from NSGA-II algorithm, it is time to choose the best result among them. The first method to do this is to calculate the pseudo weight [52] for every solution x by,

$$w_1(x) = \frac{(f_1^{max} - f_1(x))/(f_1^{max} - f_1^{min})}{(f_1^{max} - f_1(x))/(f_1^{max} - f_1^{min}) + (f_2^{max} - f_2(x))/(f_2^{max} - f_2^{min})} \quad (4.1)$$

where f_1^{max} and f_1^{min} are maximum and minimum values related to the trace of estimation error covariance matrix from the optimal solutions, with respect to this, f_2^{max} and f_2^{min} are for total number of selected sensors. The pseudo-weight values calculated for each individual result will be compared according to the specified threshold and the result with the value closest to that threshold will be selected. Threshold is set to 0.5 like in [52]. Results are in the Simulation Results section.

4.2. DECISION MAKING WITH MINIMUM DISTANCE TO ORIGIN

Another method we use to select the best result is to select the point whose coordinate is closest to the origin (0,0). The distances are normalized while they are divided by the maximum value of the respective objective. The results will be compared to the previous

method using three separate process noise parameters. Each distance is found by,

$$distance(x) = \sqrt{(f_1(x)/f_1^{max})^2 + (f_2(x)/f_2^{max})^2} \quad (4.2)$$

We will continue with one of these two methods according to the results which can be found in Simulation Results section.

4.3. DECISION MAKING WITH KNEE POINT SOLUTION

Last decision making method used in this project is called Knee Point Solution. In this case, the knee of the trade-off curve is defined as the solution where a small decrease in one objective is associated with a large increase in the other [20]. In the Pareto-Optimal Front, let say α^a and α^b are two solutions which are nearest solutions to each other where $f_1(\alpha^a) > f_1(\alpha^b)$ and $f_2(\alpha^a) < f_2(\alpha^b)$. Then the slope of the curve between these two solutions can be found as,

$$slope\{\alpha^b\} = 180 - \left[\arctan \left(\frac{f_1(\alpha^a) - f_1(\alpha^b)}{f_2(\alpha^a) - f_2(\alpha^b)} \right) \frac{180}{\pi} \right] \quad (4.3)$$

We assume $f_1(\alpha^1) = f_1^{max}$ and $f_1(\alpha^P) = f_1^{min}$, accordingly $f_2(\alpha^1) = f_2^{min}$ and $f_2(\alpha^P) = f_2^{max}$. The point that maximizes the slope gives us the Knee Point Solution and that will be selected from the Pareto-Optimal Front.

4.4. CROSSOVER OPERATORS & MUTATION

As described previously, a crossover operator is used to find the Q_t , which is an offspring population. In this project, 2 separate crossover operators will be used. Accordingly, the latest results will be observed whether there is a change. The first of these is called uniform

crossover, where offspring solutions c_1 and c_2 are found from parents p_1 and p_2 by,

$$\begin{aligned} c_1 &= \xi p_1 + (1 - \xi) p_2 \\ c_2 &= (1 - \xi) p_1 + \xi p_2 \end{aligned} \quad (4.4)$$

where ξ is selected by a random number q between $[0,1]$ [20],

$$\begin{aligned} \xi &= 1 & q \leq 0.5 \\ \xi &= 0 & q > 0.5 \end{aligned} \quad (4.5)$$

After that there is also a mutation procedure called uniform mutation. Offspring solution c_l is determined using parent solution p_l by,

$$c_l = \delta(1 - p_l) + (1 - \delta)p_l \quad (4.6)$$

where δ is obtained by the same way with Equation 4.5. There is a second crossover operator used with the same mutation which is named Simple Crossover [50]. Let i refers to a random number between $\{1, \dots, N-1\}$. Suppose that p_1 and p_2 consist of binary sets (a_1^1, \dots, a_n^1) and (a_1^2, \dots, a_n^2) where n is the total number of sensors. Then, and the new binary set of chromosomes created as follows,

$$c_1 = (a_1^1, a_2^1, \dots, a_i^1, a_{i+1}^2, \dots, a_n^2) \quad (4.7)$$

$$c_2 = (a_1^2, a_2^2, \dots, a_i^2, a_{i+1}^1, \dots, a_n^1) \quad (4.8)$$

We are going to examine the effect of these two crossover operators on our results. 6 separate way of NSGA-II solution on with using these two operators, we will observe what differences an individual crossover makes. The data obtained will be shown in the Simulation Results section.

4.5. INITIAL POPULATION TYPES

In the previous sections it was said that the applications would be done with 6 different NSGA-II methods, now in this section we will clarify them in detail. First of all, let's mention their names which are: Constant generation(30 generations in our application, will be symbolized as "G30" thereafter) Random Initial Population, G30: Previous Population, G30: Previous Unique Solutions + Random Population, Stopping Rule(SR): Random Initial Population, SR: Previous Population, SR: Previous Unique Solutions + Random Population.

Names are given according to which type of initial population will be used in each time step. When we consider our system with 16 sensors in any time step, there are 2^{16} combinations of open or closed state of sensors which are independent to each other. We cannot put that much population into the NSGA-II algorithm because otherwise the processing time is too long. After various trials, 250 populations were found to be sufficient, but we wanted to see how we could achieve results by using 100 populations to push the system a little further. In the first mentioned method which is G30: Random Initial Population, the 250 or 100 initial populations to be used are randomly selected from 2^{16} possibilities at each time step. Accordingly, at each time step, algorithm independently creates its own population at the beginning. On the other hand, the "G30" says that the program stops at 30th generation and in the latest generation which has the best solution list (Pareto-Optimal Front) will be used.

In the second method, G30: Previous population, the initial populations to be used are not independent from the final populations(solutions in Pareto-Optimal front) in previous time steps. Thus, the solution set that appears at the end of the previous time step is used as the initial population in the next time step. The amount of elements contained in the list of solutions in latest Pareto-Optimal front is the same as the initial population. Therefore, if there are fewer optimal solutions than the stated amount of population, they may repeat and need to be separated as unique. In the third method which is "G30: Previous Unique Solutions + Random Population", we will use only unique ones for the next time step instead of the whole population, and the rest will be randomly selected from 2^{16} possible options. Hence, instead of using the same elements more than once, we plan to make the algorithm more effective by simply using the necessary elements once and selecting the rest randomly. The

remaining 3 methods have population selection variations like the first 3 methods. In contrast, the program will stop at a constant generation of the first 3 methods but the stopping rule methods stops at generations according to a metric which will be described later. These 6 methods will be tested using lower (100) and higher (250) initial populations and the results will be shown in the Simulation Results section. It is our goal that there will be no significant difference between the use of constant 30 generations and the last methods with the stopping rule, as well as to avoid to complete of more generations than necessary. This will give us that we can achieve the same performance in less time.

4.6. STOPPING RULE

After several iterations, the entire population contains only the solutions near or at the Pareto optimal front. We can then define the generational distance (GD) between two consecutive generations, generation j and generation $j + 1$ [51] as,

$$GD(j + 1, j) = \sqrt{\sum_{i=1}^M g_i^2} \quad (4.9)$$

measures the distance between the non-dominated solutions obtained at generations $j + 1$ and j where M represents the total number of solutions on the pareto-optimal front in generation $j + 1$. Then g_i represents the Euclidean distance between the solution in generation $j + 1$ to its the nearest solution in its previous generation j .

It will be seen that after some generations the generational distance metric will end its downward trend and then equals to be zero even if new generations are used. New generations to be used after that point are unnecessary and will result in a waste of time. Thus, if GD metric becomes 0, it is preferred that the algorithm stop.

Results will be in the Simulation Results section.

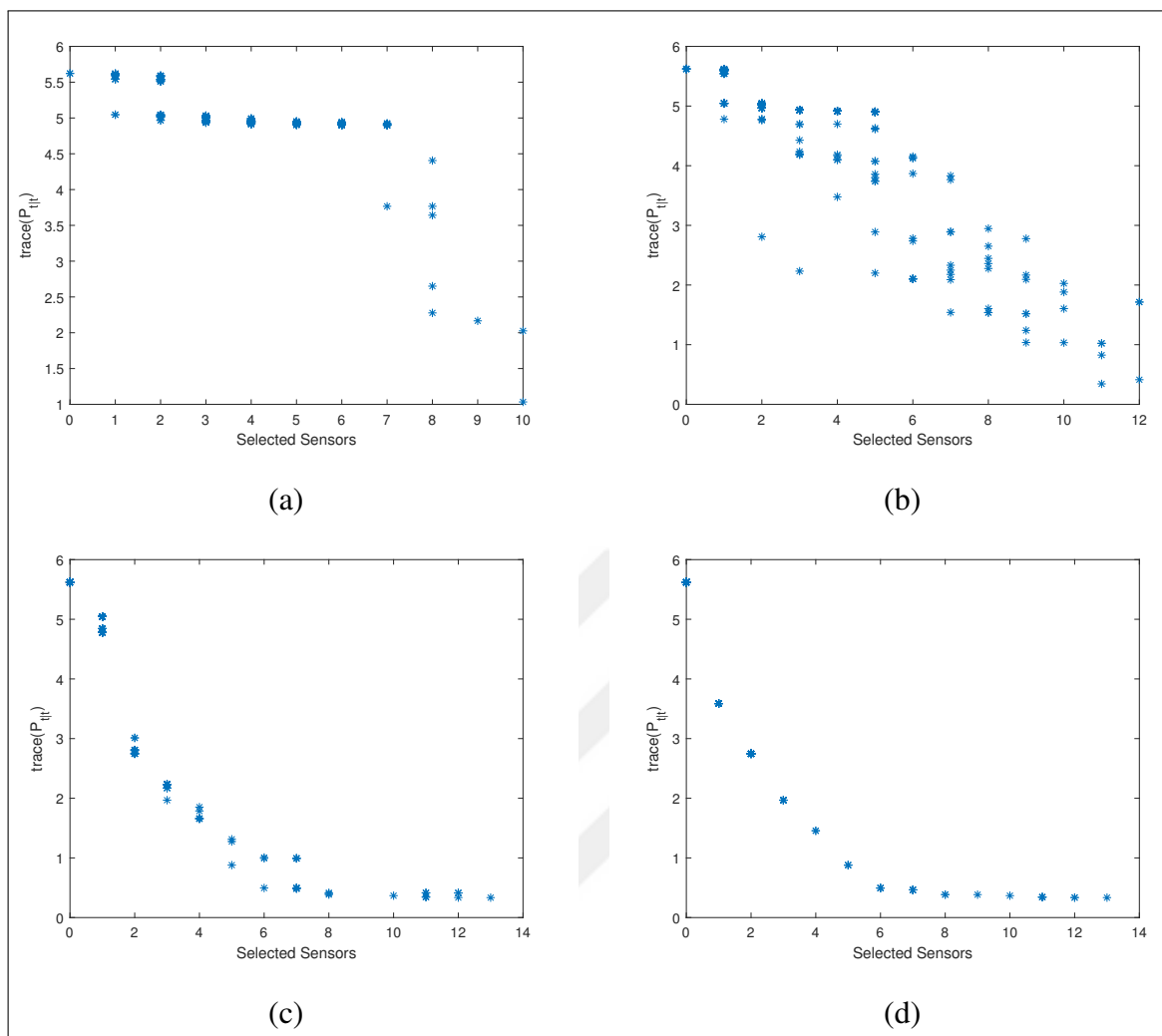


Figure 4.2. Pareto optimal front of NSGA-II after (a) $G = 1$, (b) $G = 3$, (c) $G = 6$, (d) $G = 10$ generations, $N = 16$ sensors in the WSN

5. SIMULATIONS RESULTS AND DISCUSSION

In this section, we first present the parameters used in our simulations. Then we present the performance of NSGA-II under different solution alternatives on the pareto-optimal front, different population sizes, different crossover operators and finally different number of sensors in the network.

5.1. SIMULATION PARAMETERS

We first set the number of sensors in the WSN as $N = 16$, where sensors deployed randomly in the ROI of size $50m. \times 50m.$ as shown in Fig. 5.1-(a). We generate the initial location of each target \mathbf{x}_0 using multivariate Gaussian probability density function $p(\mathbf{x}_0) \sim \mathcal{N}(\boldsymbol{\mu}_0, \mathbf{P}_0)$, with mean $\boldsymbol{\mu}_0 = [-20, -20, 2, 2]$ and covariance matrix $\mathbf{P}_0 = \text{diag}[\sigma_\theta^2, \sigma_\theta^2, 0.01, 0.01]$ where we select $3\sigma_\theta = 5$ so the initial point of the target remains in the ROI with high probability. Rest of the parameters used in our simulations are summarized in Table 5.1.

Table 5.1. Simulation parameters

Parameters	Values
Time interval(Δ)	1
Source Power (P_0)	10^4 W
Process noise parameter(τ)	10^{-2}
Measurement Noise (σ_n^2)	1
Number of Trials, T_{trials}	100
Total Time Step, T	10

At each time step of tracking $t \in \{1, 2, \dots, T\}$, we compute the MSE of Estimation between the real target state \mathbf{x}_t and estimated target state $\hat{\mathbf{x}}_t$ as,

$$MSE(t) = \frac{1}{T_{trials}} \sum_{j=1}^{T_{trials}} (\mathbf{x}_{t,j}(1) - \hat{\mathbf{x}}_{t,j}(1))^2 + (\mathbf{x}_{t,j}(2) - \hat{\mathbf{x}}_{t,j}(2))^2 \quad (5.1)$$

where $\mathbf{x}_{t,j}(1)$, $\mathbf{x}_{t,j}(2)$ and $\hat{\mathbf{x}}_{t,j}(1)$, $\hat{\mathbf{x}}_{t,j}(2)$ represents the first and second elements of the unknown state vector and estimated state vector in their j^{th} trials respectively. We select the total number of trials, i.e., different target trajectories to obtain the MSE as $T_{trials} = 100$.

We perform all the simulations on a personal computer in MATLAB with processor Intel Core i7 - 4712MQ CPU at 2.30 GHz with 12 GB RAM.

5.2. NUMERICAL RESULTS

In this section, we first examine the MSE performance when a predefined number of sensors are selected at each time step of tracking. Then, we compare the MSE performance of MOP under different solution selections, different population sizes, different implementations considering with and without stopping rules [8, 52].

5.2.1. Effect of Fixed Number of Selected Sensors

Initially, we use an Extended Kalman filter, which receives data from all sensors in the WSN [7]. Fig. 5.1-(b) shows the change of MSE over each time step over $T_{trials} = 100$ different trials. We observe that the tracking error decreases over time as compared to the initial steps of tracking and becomes stable to a constant after few time steps.

Next, at each time step of tracking, without any optimization, we choose the best A sensors which are located nearest to the estimated target location. In Fig. 5.2-(a), we vary the value of A from 1 to 5. Our numerical results show that when nearest 1 or 2 sensors are selected, MSE performance poorly diverges. Furthermore, in Fig. 5.2-(b), we present the MSE results by further excluding $A = 1$ and $A = 2$ cases. Our numerical results now show that after selecting $A = 4$ sensors, the MSE performance becomes very similar to the case where all sensors transmit their measurements.

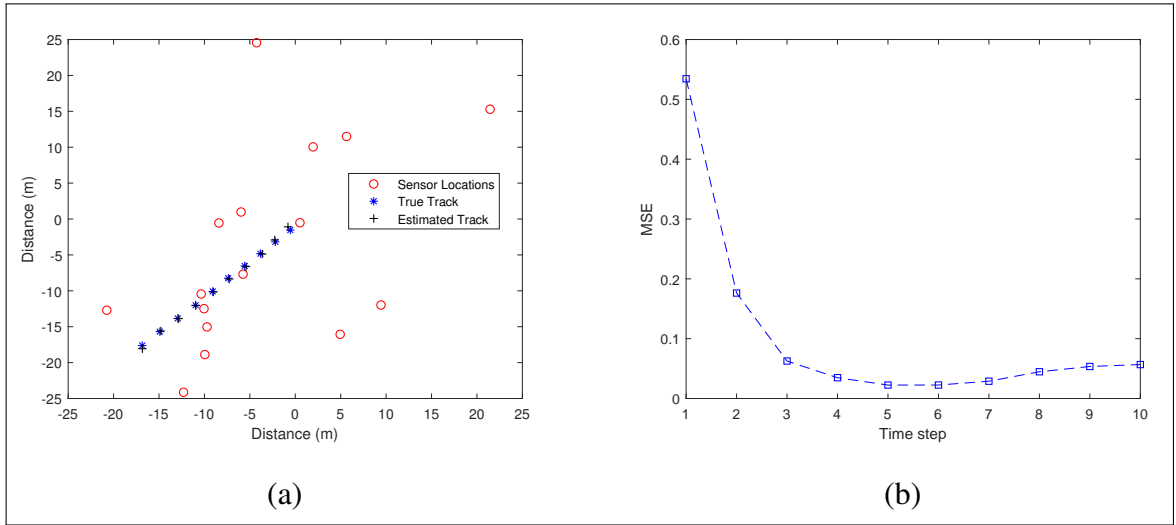


Figure 5.1. An example WSN model for target tracking where (a) $N = 16$ sensors are deployed randomly in the ROI., (b) Target tracking performance while all sensors are active at all time steps

5.2.2. Effect of Solution Selection on the Pareto-optimal Front

In this subsection, we execute NSGA-II algorithm, by setting the population size $N_{pop} = 100$, total number of NSGA-II generations without any stopping rule as $G = 30$, and we use the 1st crossover operator, uniform crossover. In Fig. 5.3, for an arbitrary target trajectory, we present the Pareto-optimal front between the trace of error covariance matrix and total number sensors selected observed at time steps (a) $t = 1$, (b) $t = 3$, (c) $t = 6$, and (d) $t = 9$ respectively. On each sub-figure, we also highlight the solution corresponding to knee-point solution, pseudo-weight solution, and the minimum distance solution. As seen from these figures, the knee-point solution typically chooses the solution with one sensor and minimum distance solution selects more sensors than that of both the knee-point solution and the pseudo-weight solution.

In Fig. 5.4, we further present the MSE and total number of selected sensors observed at each time step of tracking averaged over $T_{trials} = 100$ different target trajectories. Consistent with the above results, knee-point solution usually selects one sensor at a time and yields poor MSE performance. Pseudo-weight solution selects fewer sensors than the minimum distance solution and hence minimum distance solution yields better MSE performance as compared

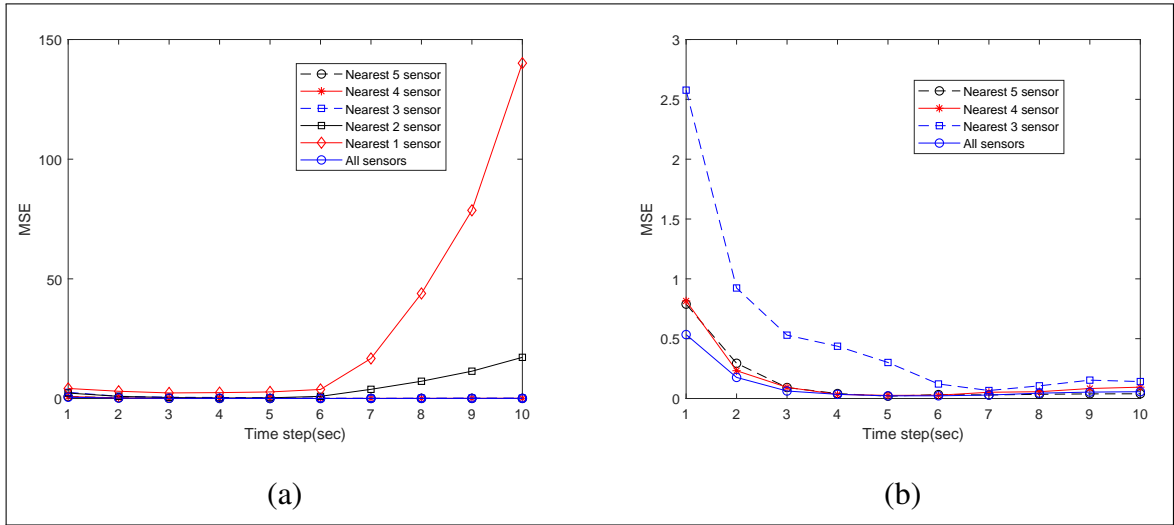


Figure 5.2. Comparison while all and only the most informative sensors are selected by (a) MSE with all implementations, (b) MSE without worst implementations

to the pseudo-weight solution.

Next in Fig.5.5-(a) and (b), for pseudo-weight solution and the minimum distance solution on the pareto-optimal front, we compare the MSE and total number selected sensors at each time step of tracking under three different process noise parameters $\tau = 10^{-1}$, $\tau = 10^{-2}$ and $\tau = 10^{-3}$. Note that $\tau = 10^{-1}$ corresponds the case with the largest uncertainty on the target and $\tau = 10^{-3}$ corresponds the case with the least uncertainty on the target. In other words, as τ increases, the target's angle of rotation tends to increase rapidly which is a more challenging tracking case and leads to a decrease in target tracking performance. The numerical results show that as the uncertainty of the target increases by increasing τ , the target tracking is successful as long as more sensors are selected. Since minimum distance solution selects more sensors than that of pseudo-weight solution, minimum distance solution has much better MSE performance than the pseudo-weight solution.

5.2.3. Effect of Population Size on Stopping Rule

Fig. 5.6 presents the Generational Distance (GD) Metric over $G = 30$ generations under two different NSGA-II population sizes $N_{pop} = 100$ and $N_{pop} = 250$. Note that a lower value of the metric indicates a better convergence, and after a number of generations, where GD

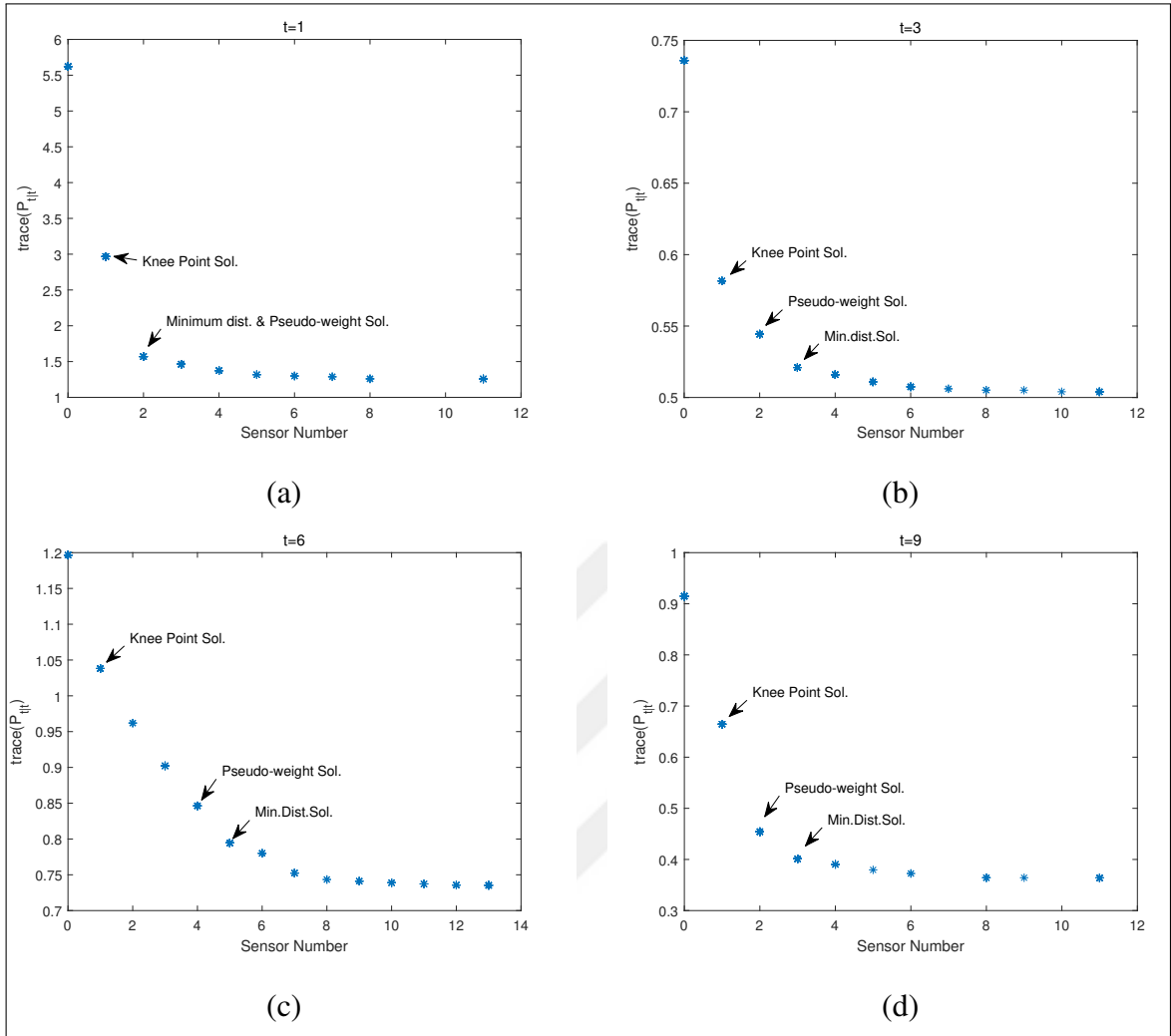


Figure 5.3. Pareto-optimal front between the trace of error covariance matrix and total number sensors selected observed at time steps (a) $t = 1$, (b) $t = 3$, (c) $t = 6$, (d) $t = 9$

tends to approach zero. Here initially the GD of $N_{pop} = 250$ is larger than the GD value of $N_{pop} = 100$ since $N_{pop} = 250$ has more solutions in the objective domain. On the other hand, after few generations GD value of $N_{pop} = 250$ becomes smaller than the GD value of $N_{pop} = 100$. This is due to the fact that increased population size tend to reproduce better solutions. In our proposed implementation, if the Pareto-optimal fronts at two consecutive generations is the same, that is $GD(j+1, j) = 0$, NSGA-II generations are terminated instead of moving to another generation since we assume that further generations may have no significant effect on the new pareto-optimal front.

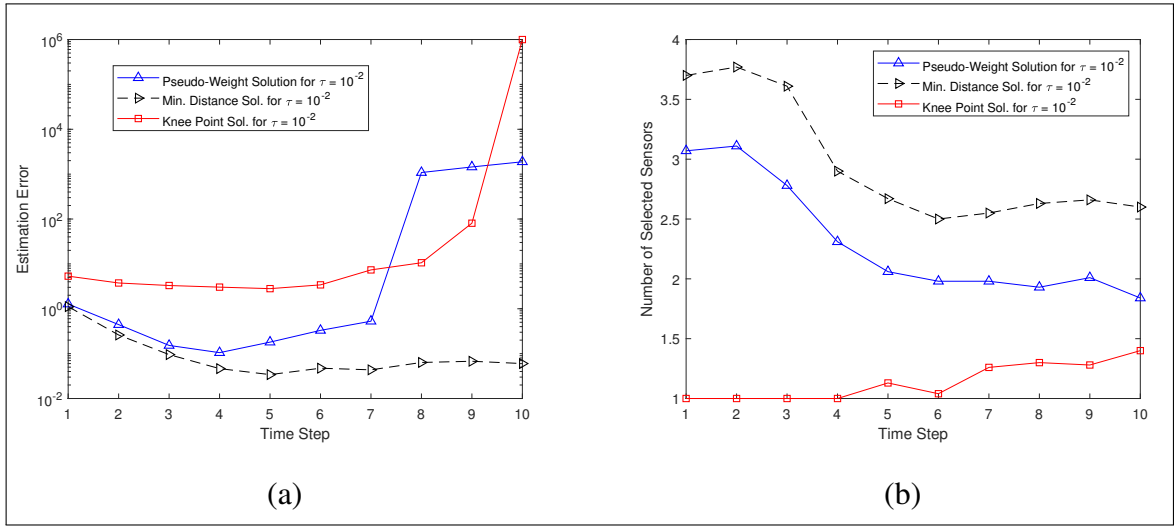


Figure 5.4. Comparison when using the knee point sol., pseudo-weight sol. and minimum distance sol. for $\tau = 10^{-2}$ by (a) MSE, (b) Total number of selected sensors

5.2.4. Effect of Different NSGA-II Implementations

In this thesis, we compare the MSE and total number of selected sensors under the following 6 different NSGA-II implementations:

1. **G30-Random Init.:** In the first approach, NSGA-II is executed for excessive $G = 30$ iterations and at the beginning of each NSGA-II executions N_{pop} solutions of the initial population are randomly selected.
2. **G30-Previous Population:** In the second approach, we execute NSGA-II for fixed $G = 30$ iterations and at the beginning of each NSGA-II execution, the final population of the previous time step is used as the initial population of the current time step.
3. **G30-Previous Unique and Random Init.:** As seen from, Fig. 5.3-(a) to 5.3-(d), the final pareto-optimal front consists of few solutions since, at the end of all generations, most of the solutions on the pareto optimal front converges to the same solution, i.e., the same sensor selection strategy. Therefore as a third approach, we keep executing NSGA-II for $G = 30$ iterations, but for the initial population of the current time step, we only accept the unique solutions in the final population of the previous time step and rest of the solutions are filled randomly.
4. **SR-Random Init.:** As a fourth approach, rather than executing NSGA-II excessive

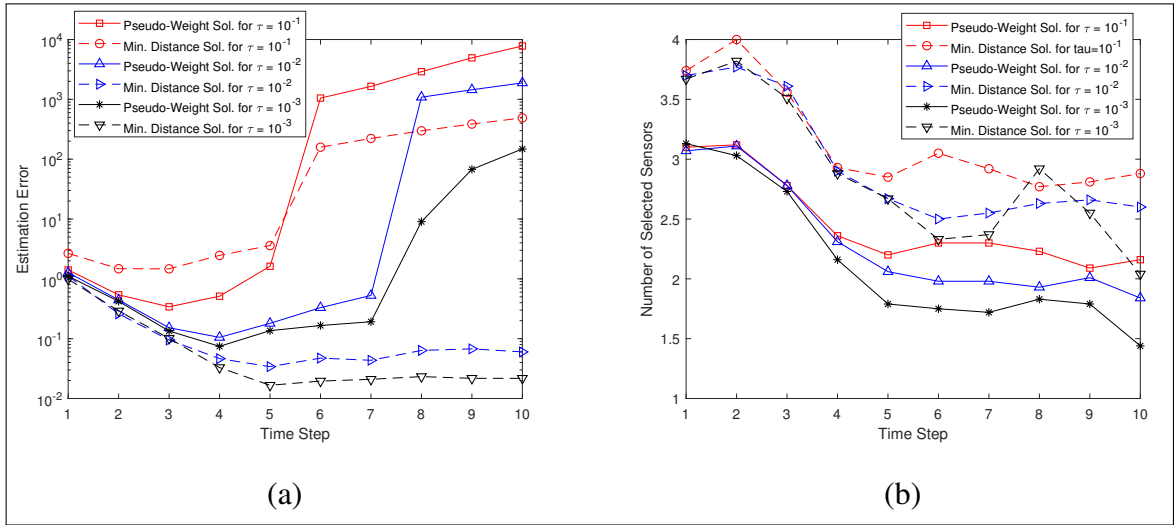


Figure 5.5. Comparison by multi-objective optimization methods by (a) MSE, (b) Selected sensor number

number of generations, we use the stopping rule (SR) defined in Section 4.6, aiming to achieve a similar performance with less number of generations. For the fourth approach, we fill the initial population randomly at the beginning of each NSGA-II executions.

5. **SR-Previous Population:** As a fifth approach, we terminate the NSGA-II generations when the SR is met and initial population at the beginning of each NSGA-II execution is selected as the final population of the previous time step.
6. **SR-Previous Unique and Random Init.:** As the last approach, we terminate the NSGA-II generations when the SR is met and same as the 3rd implementation, initial population at the beginning of each NSGA-II execution is formed by combining the unique solutions of the previous time step and randomly selected solutions.

Table 5.2 shows the number of sensors selected at each time step of tracking averaged over total tracking time steps T and total number of trials T_{Trials} , number of NSGA-II generations used to get the Pareto-optimal front with different population sizes N_{pop} , different cross-over operators (CX1: Uniform Crossover and CX2: Simple Crossover) and different number of sensors in the WSN, N . The numerical results in Table 5.2 shows that when there are $N = 16$ sensors in the WSN, the minimum distance solution selects on the average 3 sensors at each time step of tracking. When we introduce the stopping rule based on GD, with random initialization NSGA-II iterations typically terminate in about 10 iterations, where if we use

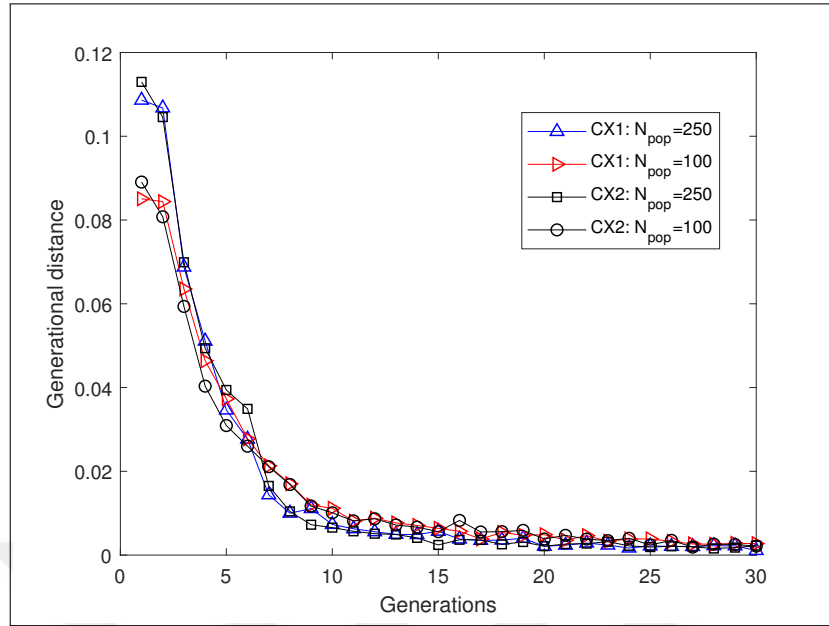


Figure 5.6. Generational distance metric after each generation for $N_{pop} = 100$ and $N_{pop} = 250$

the final population of the previous time step as the initial population choice, the generations tend to terminate around 9 generation.

Figure 5.7 shows the MSE performances 6 different NSGA-II implementations under different crossover operators. Worst MSE results are obtained under NSGA-II implementation 4 and NSGA-II implementation 5, which both uses SR and in NSGA-II implementation 4, initial population is randomly selected and in NSGA-II implementation 5 the final population of the previous time step $t - 1$ is solely used as the initial population of the current time step t . In NSGA-II implementation 4, since NSGA-II is randomly initialized, it can not reach the near pareto-optimal front when the stopping rule is met in few generations. On the other hand, in NSGA-II implementation 5, there is a quite limited solution diversity since most of the solutions in the previous population reflects the same solution. Then, NSGA-II stucks and unable to produce better solutions. Note that $G = 30$ is a reasonable generation size for all NSGA-II implementation 1, NSGA-II implementation 2 and NSGA-II implementation 3 cases, where NSGA-II can reach the optimal or near optimal Pareto-optimal front independent from the initial population. The use of generational distance based stopping rule then becomes meaningful when we use NSGA-II implementation 6, where the initial population at each

Table 5.2. Average number of sensors selected at each time step of tracking, the total number of NSGA-II generations used to get the pareto-optimal front, and different total number of sensors in the WSN

CX	N_{pop}	N	Implementations	Number of Selected Sensors - Mean	Generations Mean
1	100	16	1 -G30:Random Init.	3.1070	30
1	100	16	2-G30:Previous Population	3.0570	30
1	100	16	3-G30:Previous Unique + Random Population	3.0030	30
1	100	16	4-SR:Random Init.	3.5300	10.2070
1	100	16	5-SR:Previous Population	3.8560	9.1010
1	100	16	6-SR:Previous Unique + Random Population	3.2410	9.0810
2	100	16	1-G30:Random Init.	2.9420	30
2	100	16	2-G30:Previous Population	3.0900	30
2	100	16	3-G30:Previous Unique + Random Population	3.1400	30
2	100	16	4-SR:Random Init.	3.4580	9.9900
2	100	16	5-SR:Previous Population	3.9330	9.1410
2	100	16	6-SR:Previous Unique + Random Population	3.3060	9.0990
1	250	16	1-G30:Random Init.	3.1070	30
1	250	16	2-G30:Previous Population	3.1750	30
1	250	16	3-G30:Previous Unique + Random Population	3.1750	30
1	250	16	4-SR:Random Init.	4.1620	9.6900
1	250	16	5-SR:Previous Population	3.0980	8.9380
1	250	16	6-SR:Previous Unique + Random Population	3.5510	8.8840
2	250	16	1-G30:Random Init.	3.0490	30
2	250	16	2-G30:Previous Population	3.1410	30
2	250	16	3-G30:Previous Unique + Random Population	3.2260	30
2	250	16	4-SR:Random Init.	3.4000	9.3180
2	250	16	5-SR:Previous Population	3.0790	8.9600
2	250	16	6-SR:Previous Unique + Random Population	3.3210	9.0320
2	250	25	1-G30:Random Init.	3.4640	30
2	250	25	6-SR:Previous Unique + Random Pop.	3.5700	10.2810

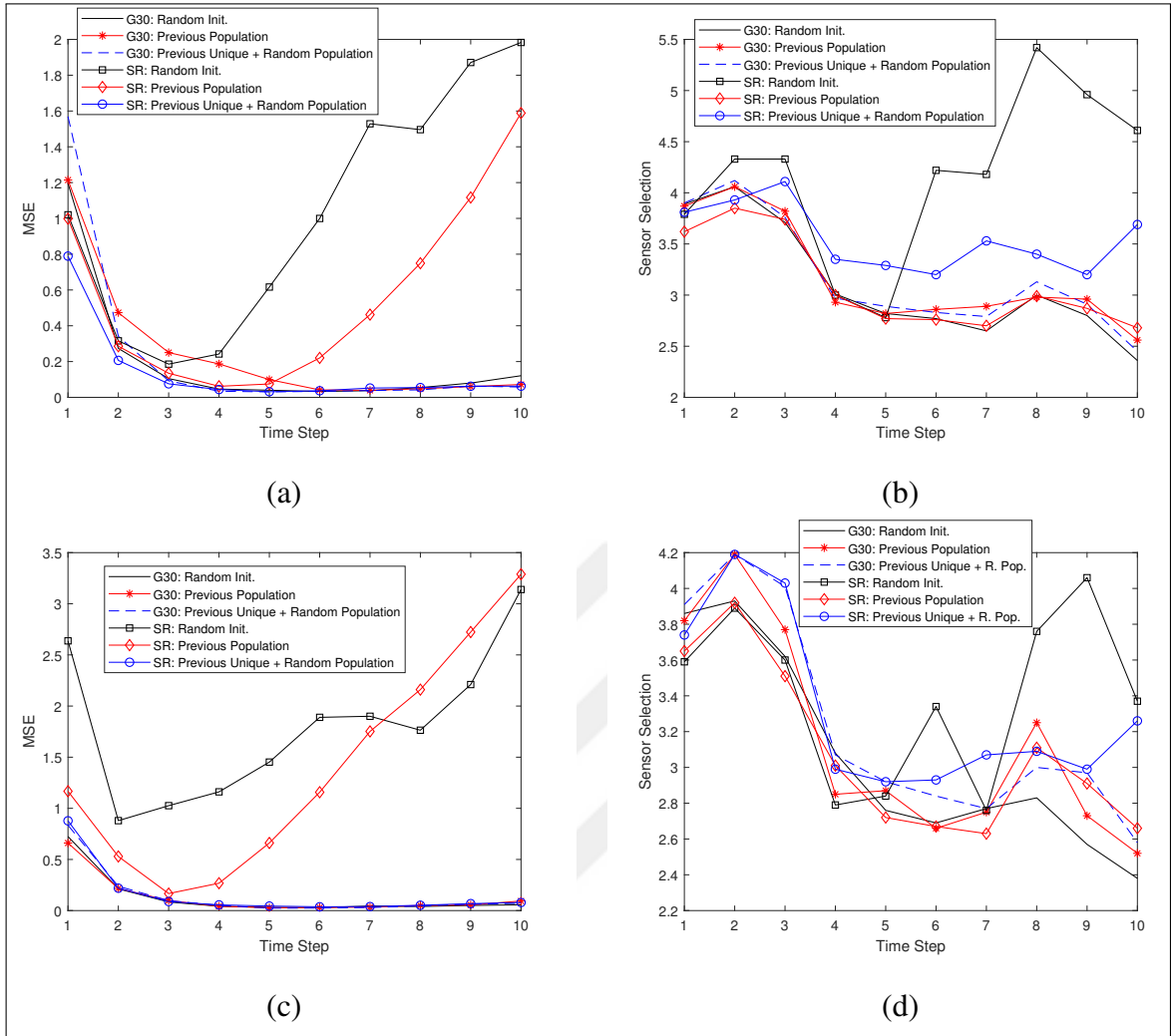


Figure 5.7. Performances of 6 methods by (a) MSE for crossover 1 and 250 populations, (b) Sensor selections for crossover 1 and 250 populations, (c) MSE for crossover 2 and 250 populations, (d) Sensor selection for crossover 2 and 250 populations

time step is formed with combining the unique solutions of the previous time step with the random solutions. Therefore when the SR is met, NSGA-II reaches to the near optimal Pareto-optimal front and the performance of NSGA-II implementation 6 becomes very similar to the implementations with excessive number of $G = 30$ generations.

Next, in Fig. 5.8-(a) and Fig. 5.8-(b), we respectively compare the MSE and total number of selected sensors at each time step of tracking when different population sizes and different crossover operators are selected. As previously shown in Fig. 5.6, the GD for $N_{pop} = 250$ becomes smaller than the GD for $N_{pop} = 100$, where smaller GD indicates better convergence.

For $N_{pop} = 250$ case, since CX1 selects more sensors than that of CX2, the MSE of CX1 is marginally better than MSE of CX2.

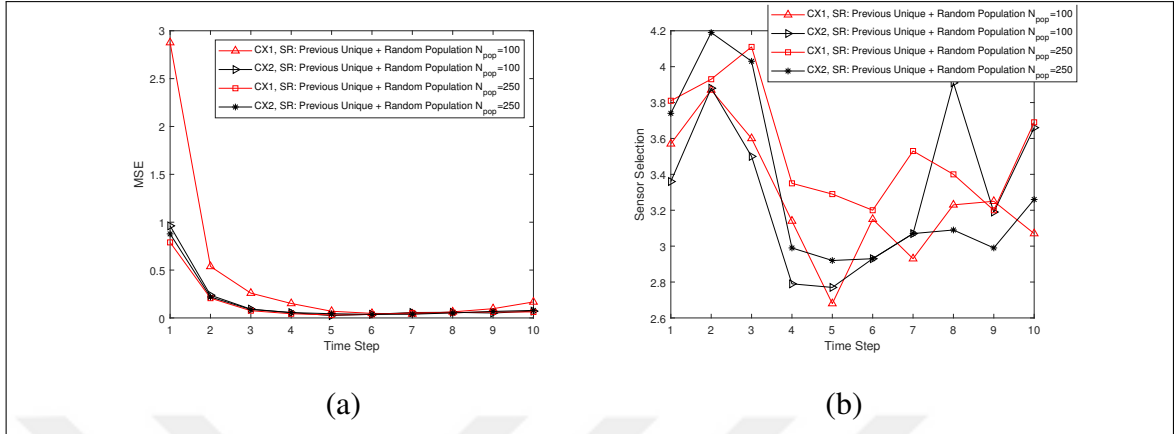


Figure 5.8. Performances of 6th Implementations by (a) MSE and (b) Total number of selected sensors

Since the NSGA-II algorithm on the average selects around 3 sensors for $N = 16$ case, in Fig. 5.9, we compare the MSE of implementation 6 with the case where $A = 3$ sensors closest to the estimated target location is selected. Simulation results show that, the MSE of implementation 6 is very close to the MSE when all sensors transmit and significantly better than $A = 3$ case. Therefore, we can conclude that it is better to use an MOP strategy rather than selecting nearest $A = 3$ sensors all the time, since MOP has the flexibility to select more informative sensors when needed or select few sensors when the number of informative sensors is few along the target trajectory.

Finally, we increase the total number of sensors to $N = 25$ as shown in Fig. 5.10 and compared implementation 1 and implementation 6 with population size $N_{pop} = 250$ and using simple crossover (CX2). Fig. 5.11-(a) and (b) then respectively show the MSE and total number of selected at each time step of tracking. Numerical results show that there is no significant difference between two methods in terms of MSE and total number of sensors selected at each time step. On the other hand, implementation 1 uses fixed number of $G = 30$ while implementation 6 terminates much earlier in about 10 generations as shown in Table 5.2. Furthermore as compared to NSGA-II implementation 6, $N = 16$ case, average number of selected sensors increases from 3.32 to 3.57 and NSGA-II terminate in around 10 generations rather than 9 generations.

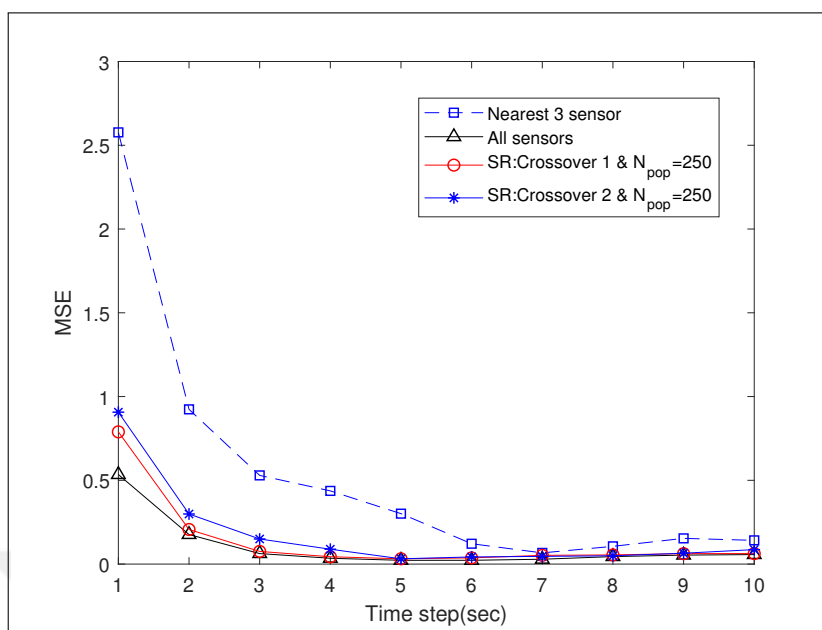


Figure 5.9. MSE comparison for 6th Implementations versus while all and only the most informative sensors are active

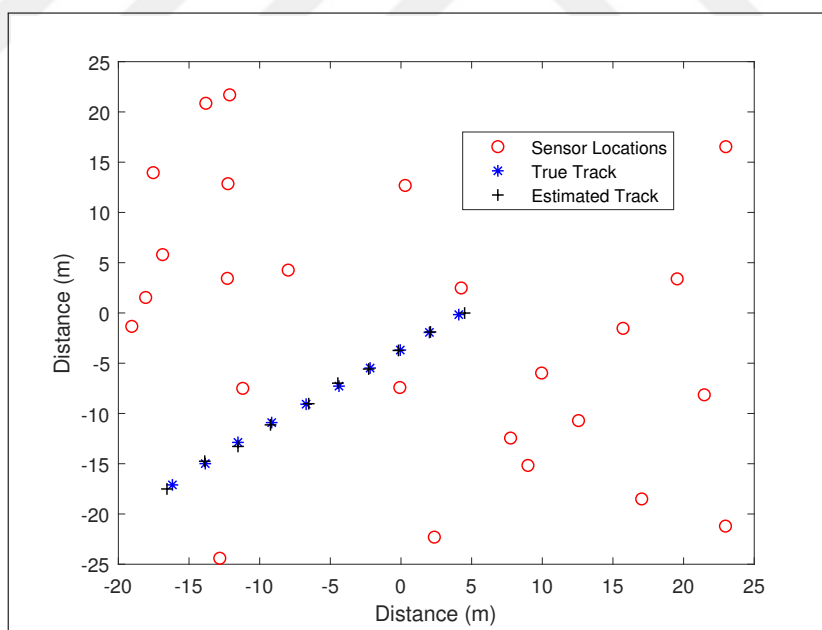


Figure 5.10. An example WSN model for target tracking where $N = 25$ sensors are deployed randomly in the ROI

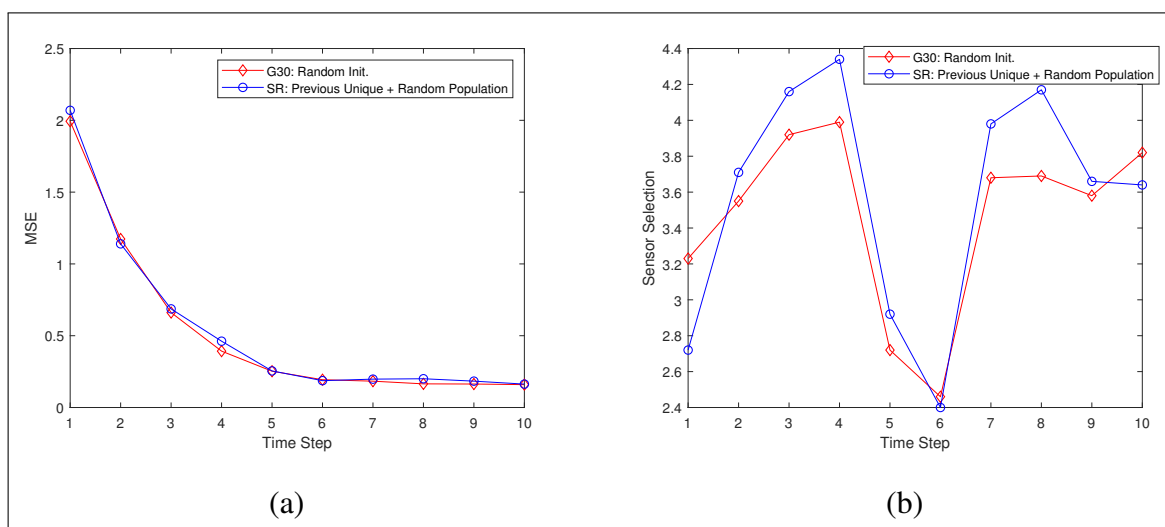


Figure 5.11. Performance comparison between implementation 1 and implementation 6 with CX2, $N_{pop} = 250$ and $N = 25$ by (a) MSE and (b) Total number of selected sensors

6. CONCLUSIONS AND FUTURE RESEARCH DIRECTIONS

In this thesis, tracking a single energy emitting target based on received sensor measurements has been studied. Rather than gathering all sensor decisions at the fusion center, only informative sensors near the target sent measurements to the fusion center and rest of the sensors stayed silent. The informative sensors selected for transmission were selected as a solution to a multiobjective optimization problem, where the objectives are minimizing the trace of the error covariance matrix, in order to minimize the estimation error and minimizing the total number of sensors selected at each time step of tracking. Formulated MOP were then solved with NSGA-II algorithm. We then examine the following effects of different MOP parameters on the performance of target tracking.

We first examined different solution selection strategies on the Pareto-optimal front. Minimum distance solution selected more sensors as compared to knee-point solution and pseudo-weight solution. Therefore, MSE with respect to the minimum distance solution was better than the MSE of both knee-point solution and pseudo-weight solution.

We then defined a stopping rule based on GD between two successive generations. We observed that as the population size increased, the Pareto-optimal front achieved with the stopping rule became similar to the Pareto-optimal front achieved with excessive NSGA-II generations.

Rather than initializing NSGA-II algorithm with random solutions at the beginning of each time step of tracking, we used the optimal solutions of the previous time step in order to reduce the total number of NSGA-II generations until the algorithm meets the stopping rule. Our numerical results showed that when only the final population of the previous time step was used as the initial of the current time step, NSGA-II might get stuck and unable to find good trade-off solutions due to the limited number of diverse solutions. On the other hand, if the initial population of the current time step was combined with the unique solutions of the previous time step and rest of the solutions were generated randomly, it became possible to achieve the Pareto-optimal front with less number of generations.

The Extended Kalman Filtering approach presented in this thesis can be easily extended to other Bayesian filtering approaches such as the Particle Filter in a straightforward fashion. Here we did not consider additional constraints while minimizing the two objective functions. To save resources over time, we can define realistic constraints on sensor selections and execute a multiobjective evolutionary algorithm with further resource constraints. Furthermore in this work we assume that sensors track single target send their measurements directly to a central node, Fusion center. As a future work, we can consider a distributed network without any fusion center and the task of the WSN might be to track multiple targets in the given ROI.



REFERENCES

1. Sohraby K, Minoli D, Znati T. *Wireless sensor networks: technology, protocols, and applications*. New Jersey: John Wiley & Sons; 2007.
2. Bokareva T, Hu W, Kanhere S, Ristic B, Gordon N, Bessell T, Rutten M, Jha S. Wireless sensor networks for battlefield surveillance. *Proceedings of the Land Warfare Conference*; 2006.
3. Yick J, Mukherjee B, Ghosal D. Wireless sensor network survey. *Computer Networks*. 2008;52(12):2292-330.
4. Gungor VC, Hancke GP. Industrial wireless sensor networks: Challenges, design principles, and technical approaches. *IEEE Transactions on Industrial Electronics*. 2009;56(10):4258-65.
5. Milenkovic A, Otto C, Jovanov E. Wireless sensor networks for personal health monitoring: Issues and an implementation. *Computer Communications*. 2006;29(13-14):2521-33.
6. Masazade E, Niu R, Varshney PK. Dynamic bit allocation for object tracking in wireless sensor networks. *IEEE Transactions on Signal Processing*. 2012;60(10):5048-63.
7. Mochnac J, Marchevsky S, Kocan P. Bayesian filtering techniques: Kalman and extended Kalman filter basics. *19th International Conference Radioelektronika*; 2009: IEEE.
8. Masazade E, Fardad M, Varshney PK. Sparsity-promoting extended Kalman filtering for target tracking in wireless sensor networks. *IEEE Signal Processing Letters*. 2012;19(12):845-8.
9. Arulampalam MS, Maskell S, Gordon N, Clapp T. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*. 2002;50(2):174-88.
10. Zuo L, Niu R, Varshney PK. Posterior CRLB based sensor selection for target tracking in

sensor networks. *IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*; 2007: IEEE.

11. Zuo L, Niu R, Varshney PK. A sensor selection approach for target tracking in sensor networks with quantized measurements. *IEEE International Conference on Acoustics, Speech and Signal Processing*; 2008: IEEE.

12. Luo Z, Jannett TC. A multi-objective method to balance energy consumption and performance for energy-based target localization in wireless sensor networks. *Proceedings of IEEE Southeastcon*; 2012: IEEE.

13. Rajagopalan R, Mohan CK, Mehrotra KG, Varshney PK. Emoca: An evolutionary multi-objective crowding algorithm. *Journal of Intelligent Systems*. 2008;17(1-3):107-24.

14. Rajagopalan R. Multi-objective optimization algorithms for sensor network design. *IEEE 11th Annual Wireless and Microwave Technology Conference (WAMICON)*; 2010:IEEE.

15. Rajagopalan R, Mohan CK, Varshney P, Mehrotra K. Multi-objective mobile agent routing in wireless sensor networks. *IEEE Congress on Evolutionary Computation*; 2005: IEEE.

16. Rajagopalan R, Niu R, Mohan CK, Varshney PK, Drozd AL. Sensor placement algorithms for target localization in sensor networks. *IEEE Radar Conference*; 2008: IEEE.

17. Masazade E, Rajagopalan R, Varshney PK, Mohan CK, Sendur GK, Keskinöz M. A multiobjective optimization approach to obtain decision thresholds for distributed detection in wireless sensor networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*. 2009;40(2):444-57.

18. Kose A, Masazade E. A multiobjective optimization approach for adaptive binary quantizer design for target tracking in wireless sensor networks. *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*; 2015: IEEE.

19. Cao N, Masazade E, Varshney PK. A multiobjective optimization based sensor selection method for target tracking in wireless sensor networks. *Proceedings of the 16th International Conference on Information Fusion*; 2013: IEEE.
20. Cao N, Choi S, Masazade E, Varshney PK. Sensor selection for target tracking in wireless sensor networks with uncertainty. *IEEE Transactions on Signal Processing*. 2016;64(20):5191-204.
21. Deb K, Pratap A, Agarwal S, Meyarivan TA. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*. 2002;6(2):182-97.
22. Rowaihy H, Eswaran S, Johnson M, Verma D, Bar-Noy A, Brown T, La Porta T. A survey of sensor selection schemes in wireless sensor networks. *Unattended Ground, Sea, and Air Sensor Technologies and Applications IX*; 2007: International Society for Optics and Photonics.
23. Joshi S, Boyd S. Sensor selection via convex optimization. *IEEE Transactions on Signal Processing*. 2008;57(2):451-62.
24. Wang H, Yao K, Pottie G, Estrin D. Entropy-based sensor selection heuristic for target localization. *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*; 2004: ACM.
25. Williams JL, Fisher JW, Willsky AS. Approximate dynamic programming for communication-constrained sensor network management. *IEEE Transactions on Signal Processing*. 2007;55(8):4300-11.
26. Hoffmann GM, Tomlin CJ. Mobile sensor network control using mutual information methods and particle filters. *IEEE Transactions on Automatic Control*. 2009;55(1):32-47.
27. Zhao F, Shin J, Reich J. Information-driven dynamic sensor collaboration for tracking applications. *IEEE Signal Processing Magazine*. 2002;19(2):61-72.

28. Masazade E, Niu R, Varshney PK, Keskinöz M. Energy aware iterative source localization for wireless sensor networks. *IEEE Transactions on Signal Processing*. 2010;58(9):4824-35.
29. Mo Y, Ambrosino R, Sinopoli B. Sensor selection strategies for state estimation in energy constrained wireless sensor networks. *Automatica*. 2011;47(7):1330-8.
30. Liu S, Kar S, Fardad M, Varshney PK. Sparsity-aware sensor collaboration for linear coherent estimation. *IEEE Transactions on Signal Processing*. 2015;63(10):2582-96.
31. Liu S, Fardad M, Masazade E, Varshney PK. Optimal periodic sensor scheduling in networks of dynamical systems. *IEEE Transactions on Signal Processing*. 2014;62(12):3055-68.
32. Nahi N. Optimal recursive estimation with uncertain observation. *IEEE Transactions on Information Theory*. 1969;15(4):457-62.
33. Hadidi M, Schwartz S. Linear recursive state estimators under uncertain observations. *IEEE Transactions on Automatic Control*. 1979;24(6):944-8.
34. Hounkpevi FO, Yaz EE. Robust minimum variance linear state estimators for multiple sensors with different failure rates. *Automatica*. 2007;43(7):1274-80.
35. Zhang H, Shi Y, Mehr AS. Robust weighted H filtering for networked systems with intermittent measurements of multiple sensors. *International Journal of Adaptive Control and Signal Processing*. 2011;25(4):313-30.
36. Xu W, Ma K, Trappe W, Zhang Y. Jamming sensor networks: attack and defense strategies. *IEEE Network*. 2006;20(3):41-7.
37. Mariton M. *Jump linear systems in automatic control*. New York: M. Dekker; 1990.
38. Costa OL, Guerra S. Stationary filter for linear minimum mean square error estimator of discrete-time Markovian jump systems. *IEEE Transactions on Automatic Control*. 2002;47(8):1351-6.

39. Sinopoli B, Schenato L, Franceschetti M, Poolla K, Jordan MI, Sastry SS. Kalman filtering with intermittent observations. *IEEE Transactions on Automatic Control*. 2004;49(9):1453-64.
40. Ozdemir O, Niu R, Varshney PK. Channel aware target localization with quantized data in wireless sensor networks. *IEEE Transactions on Signal Processing*. 2008;57(3):1190-202.
41. Maazade E, Niu R, Varshney PK, Keskinöz M. Channel aware iterative source localization for wireless sensor networks. *13th International Conference on Information Fusion*; 2010: IEEE.
42. Nasir M, Sengupta S, Das S, Suganthan PN. An improved multi-objective optimization algorithm based on fuzzy dominance for risk minimization in biometric sensor network. *IEEE Congress on Evolutionary Computation*; 2012: IEEE.
43. Deb K. *Multi-objective optimization using evolutionary algorithms*. New York: John Wiley & Sons; 2001.
44. Coello CA, Lamont GB, Van Veldhuizen DA. *Evolutionary algorithms for solving multi-objective problems*. New York: Springer; 2007.
45. Osyczka A. *Evolutionary algorithms for single and multicriteria design optimization*. Heidelberg: Physica-Verlag; 2002.
46. Zitzler E, Deb K, Thiele L, Coello CAC, Corne DW. Lecture Notes in Computer Science. *Proceedings of the First Evolutionary Multi-Criterion Optimization (EMO-01) Conference*; 2001.
47. Zitzler E, Deb K, Thiele L, Coello CAC, Corne DW. Lecture Notes in Computer Science. *Proceedings of the Second Evolutionary Multi-Criterion Optimization (EMO-01) Conference*; 2001.
48. Coello CAC, Aguirre AH, Zitzler E, editors. Lecture Notes in Computer Science. *Evolutionary Multi-Criterion Optimization: Third International Conference, EMO*; 2005.

49. Obayashi S, Deb K, Poloni C, Hiroyasu T, Murata T, editors. Lecture Notes in Computer Science. *Evolutionary Multi-Criterion Optimization: 4th International Conference, EMO*; 2007.
50. Herrera F, Lozano M, Verdegay JL. Tackling real-coded genetic algorithms: Operators and tools for behavioural analysis. *Artificial Intelligence Review*. 1998;12(4):265-319.
51. Rajagopalan R, Mohan CK, Mehrotra KG, Varshney PK. An evolutionary multi-objective crowding algorithm (EMOCA): Benchmark test function results. *2nd Indian International Conference on Artificial Intelligence*; 2005: IICAI
52. Deb K, Karthik S. Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling. *International Conference on Evolutionary Multi-criterion Optimization*; 2007.
53. Leon-Garcia A. *Probability, statistics, and random processes for electrical engineering*. New Jersey: Pearson; 2008.
54. Kose A. *Resource Aware Adaptive Binary Quantizer Design For Target Tracking In Wireless Sensor Networks*. Yeditepe University: MSc Thesis; 2019.