REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

MULTI-SENSOR DATA FUSION FOR PATH PREDICTION OF ESCAPING FROM ENGAGEMENT IN COMBAT AIRCRAFT

Enver Nurullah GÖKAL

MASTER OF SCIENCE THESIS

Department of Avionics Engineering
Avionics Engineering

Supervisor
Assoc. Prof. Ufuk SAKARYA

February, 2022

REPUBLIC OF TURKEY
YILDIZ TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

MULTI-SENSOR DATA FUSION FOR PATH PREDICTION OF ESCAPING FROM ENGAGEMENT IN COMBAT AIRCRAFT

A thesis submitted by Enver Nurullah GÖKAL in partial fulfillment of the requirements for the degree of **MASTER OF SCIENCE** is approved by the committee on 04.02.2022 in Department of Avionics Engineering, Avionics Engineering.

Assoc. Prof. Ufuk SAKARYA
Yildiz Technical University
Supervisor

Approved By the Examining Committee

Assoc. Prof. Ufuk SAKARYA, Supervisor
Yildiz Technical University _____

Prof. Şeref Naci ENGİN, Member
Yildiz Technical University _____

Asst. Prof. Ramazan YENİÇERİ, Member
İstanbul Technical University _____

I hereby declare that I have obtained the required legal permissions during data collection and exploitation procedures, that I have made the in-text citations and cited the references properly, that I haven't falsified and/or fabricated research data and results of the study and that I have abided by the principles of the scientific research and ethics during my Thesis Study under the title of Multi-sensor Data Fusion for Path Prediction of Escaping from Engagement in Combat Aircraft supervised by my supervisor, Assoc. Prof. Ufuk SAKARYA. In the case of a discovery of false statement, I am to acknowledge any legal consequence.

Enver Nurullah GÖKAL

Signature

Dedicated to my family

ACKNOWLEDGEMENTS

First of all, i would like to express my gratitude to my advisor Assoc. Prof. Dr. Ufuk SAKARYA for his constant guidence and help during my master's thesis study. He provided me the required motivation and counseling throughout the tough times of this process.

In addition, I would like to express my gratitude to my dear family members who made the conditions and work easier for me and always made me feel their support during this compelling process.

Enver Nurullah GÖKAL

TABLE OF CONTENTS

LIST OF SYMBOLS

$R(\theta)$	2D Rotation Matrix
u	Control Vector
R	Covariance Matrix of Measurement Noise
Q	Covariance Matrix of Process Noise
P	Error Covariance
T	Homogeneous Transformation Matrix
B	Input Control Matrix
K	Kalman Gain
A_eq, b_eq	Linear Equality Constraints
A, b	Linear Inequality Constraints
LB	Lower Boundary
H	Measurement Matrix
v	Measurement Noise
$\tilde{y}$	Measurement Residual
z	Measurement Vector
C_eq	Nonlinear Equality Constraint
C	Nonlinear Inequality Constraint
e_{obs}	Observation Error
P^-	Predicted Error Covariance
$\hat{x}^-$	Predicted State Estimation
w	Process Noise
$R_y(\beta)$	Rotation Matrix of Pitch
$R_x(\gamma)$	Rotation Matrix of Roll

$R_z(\alpha)$	Rotation Matrix of Yaw
$\hat{x}$	State Estimation
F	State Transition Matrix
$R(\alpha, \beta, \gamma)$	Uniform Rotation Matrix of Pitch, Yaw and Roll
P^+	Updated Error Covariance
$\hat{x}^+$	Updated State Estimation
UB	Upper Boundary

LIST OF ABBREVIATIONS

AAM	Air-to-Air Missile
EKF	Extended Kalman Filter
GPS	Global Positioning System
İHA	İnsansız Hava Aracı
KF	Kalman Filter
MALE	Medium Altitude Long Endurance
NLP	Non-Linear Programming
SQP	Sequential Quadratic Programming
SAM	Surface-to-Air Missile
UAV	Unmanned Aerial Vehicle
UCAV	Unmanned Combat Aircraft

LIST OF FIGURES

ABSTRACT

Multi-sensor Data Fusion for Path Prediction of Escaping from Engagement in Combat Aircraft

Enver Nurullah GÖKAL

Department of Avionics Engineering
Master of Science Thesis

Supervisor: Assoc. Prof. Ufuk SAKARYA

Achieving air superiority is one of the key steps to success in warfare. It is necessary for a combat aircraft to have the survivability it needs in an aggressive combat environment. UAVs have now become an essential component of military air operations. When the aircraft is not piloted by a person, nations can use UAVs to conduct military missions with less danger. UAVs can be operated in two ways: by pilots from remote control stations or by flying autonomously. Unmanned aerial vehicles (UAVs) suffer from maintaining the maneuverability and navigational ability in the event of a disconnection from the control station.

In this work, an escape path prediction algorithm developed by fusing multi-sensor data is presented to facilitate the escape of engagement of UAVs. It is attempted to aggregate data from various sensors by processing them under the impact of noise and come up with the best escape path prediction estimation algorithm possible. Data from radars are evaluated in the Extended Kalman Filter and used to make estimations. The estimations made are used in constraint optimization to generate an instantaneous optimal escape route. Since the constraints and objective function are not linear, nonlinear programming is used as a method of constraint optimization. According to the simulation results, the proposed method shows a promising result for escaping from engagement in the selected scenario.

It is also aimed to see how the Extended Kalman Filter influences the functioning of this approach at various noise levels by simulations. This observation is critical in determining the influence of the suggested method on the UAV's resource use.

Keywords: Unmanned aerial vehicle, extended kalman filter, nonlinear programming, sensor fusion, path prediction

ÖZET

Savaş Uçaklarında Angajmandan Kaçış Yolu Kestirimi İçin Çok Sensörlü Veri Füzyonu

Enver Nurullah GÖKAL

Aviyonik Mühendisliği Anabilim Dalı
Yüksek Lisans Tezi

Danışman: Doç. Dr. Ufuk SAKARYA

Savaşta başarı elde etmenin en önemli koşullarından birisi, hava üstünlüğünü sağlamaktır. Saldırgan muharebe ortamında bulunan bir savaş uçağının, gereken hayatta kalma özelliklerine sahip olması gerekmektedir. İHA'lar artık orduda hava operasyonlarının vazgeçilmez bir unsuru haline geldi. Uçağın içinde bir insan pilotun olmaması sayesinde, ülkeler İHA'ları kullanarak daha az riskle askeri operasyonlar gerçekleştirebilir. İHA'ların hareket kontrolü iki şekilde yapılabilmektedir; İHA'lar, uzaktan kumanda istasyonlarından pilotlar tarafından kontrol edilir veya otonom olarak uçabilir. İnsansız hava araçlarında (İHA), kontrol istasyonuyla olan bağlantının kesilmesi durumunda, İHA'nın hareket ve seyrüsefer kabiliyetlerini koruması zorlaşır.

Bu çalışmada, insansız hava araçlarının angajmandan kaçışını sağlamak için çok sensörlü veri füzyonu yöntemiyle geliştirilen bir kaçış yolu kestirimi algoritması sunulmaktadır. Çeşitli sensörlerden gelen verileri gürültünün etkisi altında işleyerek toplamaya ve mümkün olan en iyi kaçış yolu tahmin algoritması bulmaya çalışılır. Gelen radar verileri, tahmin yapmak üzere Genişletilmiş Kalman Filtresine sokularak değerlendirilir. Yapılan tahminler, doğrusal olmayan programlama yönteminde kullanılır ve anlık optimal kaçış yolu belirlenir. Sahip olunan kısıtlamalar ve amaç fonksiyonu lineer olmadığı için kısıtlı optimizasyon yöntemi olarak doğrusal olmayan programlama kullanılır. Simülasyon sonuçlarına göre, önerilen yöntem seçilen senaryoda angajmandan kaçış için umut verici sonuçlar sunmuştur.

Ayrıca Genişletilmiş Kalman Filtresinin çeşitli gürültü seviyelerini simüle ederek bu yaklaşımın işleyişini çeşitli gürültü seviyelerinde nasıl etkilediğini görmek

amaçlanmaktadır. Bu gözlem, önerilen yöntemin İHA'nın kaynak kullanımı üzerindeki etkisini belirlemede kritik öneme sahiptir.

Anahtar Kelimeler: İnsansız hava aracı, genişletilmiş kalman filtresi, doğrusal olmayan programlama, sensör füzyonu, kaçış yolu kestirimi

1
INTRODUCTION

One of the most important conditions for success in warfare is to achieve air superiority. It is of great importance for an aircraft to be able to use the survivability features hidden in the aircraft when it is in a man-made hostile environment, in order to provide air dominance. This research is focused on a decision-making algorithm to develop maneuverability of the combat aircraft in engagement conditions. The term "engagement", which is often used in military matters, refers to a combat between two sides. Engagement is initiated by the attacking force to perform a task. Engagement ends when the attacker completes the mission or quits the mission [1]. For an aircraft to successfully exit the engagement, it is important to make quick decisions based on the aircraft's maneuverability and its opponent's position. This research is based on a dogfight engagement with an attacking aircraft. The aim is to ensure that the attacked aircraft escape from engagement in the most optimized way by making fast and accurate decisions. In the scenario discussed in this study, both the attacked and the attacking parties are UAVs. The attacked the attacking party may also be Air-to-Air Missile (AAM) or Surface-to-Air Missile (SAM), rather than a UAV.

UAV is an abbreviation for Unmanned Aerial Vehicle. The most basic feature of UAVs is that there is no human inside the UAV; thus avoiding the risk of pilots being harmed in battle [2]. UAVs have now become an indispensable element of air operations in the military. Due to the absence of a human pilot operating the aircraft, nations can perform military operations with less risk using UAVs. In addition, UAVs, with their functions, provide great efficiency contribution to armies in military operations. Some of the uses of UAVs, which can be used in a wide variety of ways in war, are as follows; surveillance, assigning a target to other armed systems, attacking the target by itself. UAVs can attack fixed or moving targets [3]. Medium Altitude Long Endurance (MALE) Anka UAV produced by Turkish Aerospace Industries can be given as an example of a UAV [4]. Anka is a military UAV designed and manufactured to perform surveillance, reconnaissance, target detection and target recognition tasks.

Movement control of UAVs can be performed in two ways; UAVs are controlled by

pilots from remote control stations or can fly autonomously. In addition to the remote control station, communication satellites and GPS satellites can also be used in flight control. When a UAV controlled from a remote control station is disconnected from the control station, it should be able to continue its flight autonomously in order to avoid a catastrophic result [5]. The basic methods that provide autonomous flight in UAVs are as follows; artificial neural networks, machine learning and real-time path planning with artificial intelligence or numerical methods [6].

1.1 Literature Review

There are other studies with various approaches on this subject. Capello et al. [7] proposed a Particle Filter based navigation and guidance system based on Remotely Piloted Aircraft Systems (RPAS). In the proposed method, Particle Filter based estimation method was examined and a comparison is made between particle filter, Extended Kalman Filter and Unscented Kalman Filter.

López & Żbikowski [8] proposed an autonomous decision-making algorithm for unmanned combat aircraft (UCAV) with 14 different maneuvering options. Each decision is evaluated based on a score equation considering external constraints.

Zhang et al. [9] proposed a sequential convex programming method for path planning to be used in UAVs. When the problem is a nonlinear control problem which is non-convex, a method for optimization by conversion of the non-convex problem to a convex problem or series of convex problems is proposed.

Tisdale et al. [10] proposed a camera-based estimation and path prediction method for a search and locating task for a group of UAVs. In the proposed method, recursive Bayesian estimation was used as the estimation method, and non-convex constrained optimization was used in the path planning stage.

Bortoff [11] proposed a UAV path planning algorithm. Aim of the proposed two-stage path planning algorithm is to determine an optimal path for UAVs by making a trade-off between stealth and path length in areas with radar.

Jiang & Liang [12] has proposed a case based path planning algorithm with a standoff distance for autonomous UAVs. In the proposed method, target trajectory estimation has been made with quadratic functions. For target localization, a nonlinear least squares estimation method is used. Finally, a case based decision making algorithm stepped in to accomplish the path planning with a standoff distance. In the proposed method, sensor noise is taken into account when using the measured values of the

sensors. A new case-based guidance method has been proposed to make a balanced trade-off between the optimal path planning method and real-time performance.

Yang et al. [13] proposed a path planning method using passive detection system for target detection. The use of radar in the target detection system makes it easier for the aircraft to reveal its position by other systems, so a method using a passive detection system as a target detection system has been proposed. In the proposed method, Partially Observable Markov Decision Process is used as the decision making algorithm.

Kang et al. [14] proposed a Kalman filter algorithm for UAV path planning. In this study, it is aimed to eliminate the negativities caused by the uncertainties in the flight environment. With a Kalman filter-based module developed by taking into account sensor noises, GPS sensor noise and model noise, threats were modeled and a safe distance for the UAV has attempted to be determined.

Wu et al. [15] proposed a path planning method which uses a Kalman filter-based prediction algorithm for collision avoidance in UAV groups. In order to eliminate the inconsistencies coming from the noise caused by the communication of many UAVs in a dynamic environment, a Kalman filter based estimation method has been proposed.

Luo et al. [16] proposed a position estimation and collision avoidance method for UAVs. Due to the colored noise seen in the received signal strength measurement coming from the communication module, the distance estimation was performed using the colored noise model in the Extended Kalman Filter.

Mao et al. [17] proposed an Extended Kalman Filter design for position estimation for UAVs. The proposed design in this study is based on the situation where the UAVs temporarily lost their GPS connection. In the scenario created, a group of UAVs cooperating with each other is considered.

Goh et al. [18] has proposed a Weighted Measurement Fusion Kalman Filter for UAV navigation. In the proposed method, each measurement value from the sensor is used by weighting it according to the distance it travels. Compared to the standard Kalman Filter, the proposed method gave better results in terms of position calculation error.

1.2 Objective of the Thesis

This study aims to develop a path planning algorithm for the unmanned aerial vehicle (UAV) to survive the engagement on its own in case the unmanned aerial vehicle (UAV) which is disconnected from the ground control station during an engagement. It is

important for the UAV to use it's autonomous flight features to prevent any catastrophic results in an engagement when the connection link from ground control station or satellite has cut down. These autonomous flight features include optimal escape route estimation. When the scenario stated above is occured, UAV shall generate it's path using optimal path prediction algorithm to escape from engagement. For these reasons, in this thesis, it is aimed to combine data from different sensors by processing them under the influence of noise, and to come up with an optimal escape path prediction estimation algorithm.

In addition, in this study, the standard deviation values of the measurement noise representing the noise in the sensors and the process noise representing the noise in the environment are given as input to the Extended Kalman Filter model used for estimation. It is aimed to observe how the Extended Kalman Filter affects the operation of this method at different noise levels by making simulations at different noise values. This observation is important to determine the impact of the proposed method on the resource utilization of the UAV.

As the constraint optimization method of this study, nonlinear programming method is used. In this study, it is also aimed to observe the effects of the following parts of the nonlinear programming algorithm; objective function, linear & nonlinear equality & inequality constraints on the optimal escape path of the UAV. While constraint optimisation is made, weight values of the distance and the angle have a decisive effect on the planned escape path. Which means the weight values of the distance and the angle effects the resource utilization of the UAV.

1.3 Hypothesis

In this thesis, a UAV guidance and navigation method based on Extended Kalman Filter and Nonlinear Programming is presented. In the first step, to estimate the enemy aircraft, position the multiple-sensor values are fused using the Extended Kalman Filter. Two sensors are used: One of them is the range sensor and the other is the angle sensor. After the prediction of the enemy aircraft position and direction a constraint optimization was made with the help of nonlinear programming. In this step, the constraints are defined according to the coordinate axes of the escaping UAV. Hence, UAV maneuverability can change according to the pitch, the roll, and the yaw axes of the UAV. Thus, the nonlinear programming is applied on the coordinate axes of the escaping UAV. After the obtaining the solution according to the coordinate axes of the escaping UAV, the solution is converted to the original coordinate axes. In briefly, based on these two methods, an escape path prediction algorithm has been developed.

1.4 Structure of The Thesis

The rest of the thesis is outlined as follows: In Chapter 2, necessary background information about the methods and algorithms used in this study is given. The titles which the information about is given are: Kalman Filter, Extended Kalman Filter, Nonlinear Programming and 3D Transformations.

In Chapter 3, proposed method given in this study is explained. Three base steps of the proposed method which examined in this chapter are: Position Estimation, Constrained Optimization and Vector Transformation.

In Chapter 4, the four scenarios run on the simulation are described first. Then, results of the simulation are examined for the four different scenarios separately. In addition, a ground truth calculation is given after the simulation results.

In Chapter 5, conclusion and the final thoughts for this work is presented.

2
BACKGROUND

2.1 Kalman Filter

2.1.1 Introduction

The Kalman Filter method was proposed by Rudolf E. Kalman in 1960 [19]. The Kalman Filter is an algorithm which takes continuous measurements as inputs in order to estimate desired unknown variables. Kalman Filter is often used when a state of some system cannot be measured directly [20]. To estimate the state of the system optimally, Kalman Filter is used. When there is multiple sensors with noise, Kalman Filter can be used to combine the data from those sensors and make an estimation of the signal that can't be directly measured. This is called sensor fusion [21]. Some of the main areas where the Kalman Filter is used are; tracking, navigation and guidance in aviation, vehicle control, position estimation applications with inertial measurement unit, statistics, and economics [22]. Kalman Filter is also very important in multi sensor data fusion.

To help understand the Kalman gain variable of the Kalman Filter, the simple block diagram is shown in Figure 2.1. In Figure 2.1, K represents the Kalman gain, u the control input, y the output, and x the internal variable that cannot be measured and needs to be estimated. In addition, $\hat{x}$ is the estimated state of the variable x, $\hat{y}$ is the estimated state of the output y. In order to predict the state change in the most accurate way, it is aimed to bring the $\hat{x}$ closer to x, in other words, the mathematical model is tried to be closest to the system. The e_{obs} in Formula 2.5 represents the observation error, which is the difference between the actual state x and the estimated state $\hat{x}$. The extended representation of e_{obs} is formed using Equations 2.1, 2.2, 2.3, 2.4, 2.5 and 2.6. The extended representation of e_{obs} is shown in Equation 2.7. An exponential solution for this problem is provided in Equation 2.8. In the Equation 2.8, when $A-KC$ is less than zero, our observation error converges to zero. By having a controller like Kalman gain controller in a feedback loop like in this example, the error conversion rate can be controlled as desired [21].

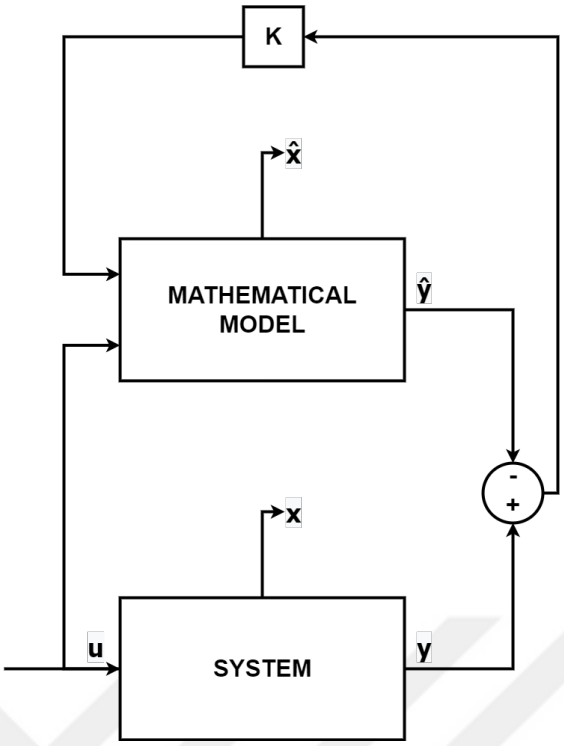

Figure 2.1 Overview of Kalman gain: K and the state estimation model (adapted from [21])

$$x_k = Ax_{k-1} + Bu_{k-1} \tag{2.1}$$

$$y_k = Cx_{k-1} \tag{2.2}$$

$$\hat{x}_k = A\hat{x}_{k-1} + Bu + K(y_{k-1} - \hat{y}_{k-1}) \tag{2.3}$$

$$\hat{y}_k = C\hat{x}_{k-1} \tag{2.4}$$

$$e_{obs} = x - \hat{x} \tag{2.5}$$

$$y - \hat{y} = C(x - \hat{x}) = Ce_{obs} \tag{2.6}$$

$$(e_{obs})_k = x_k - \hat{x}_k = Ax_{k-1} - A\hat{x}_{k-1} + Bu_{k-1} - Bu_{k-1} - K(y_{k-1} - \hat{y}_{k-1}) = (A-KC)(e_{obs})_{k-1} \tag{2.7}$$

$$e_{obs}(t) = e^{(A-KC)t} e_{obs}(0) \tag{2.8}$$

Kalman Filters are used to predict new states of the system, taking into account of previous states of the system and current noise. The Kalman Filter consists of two stages. These stages are prediction and update. The first stage is prediction, at this stage, an estimation of the stage of the system is made. After the prediction, the update stages comes next. At update stage, the estimation from first step is updated using the measurements coming from sensors under the noise [23]. Kalman Filter shows it's advantage mostly in situations like this, where it is aimed to make state estimations within the systems under influence of measurement noise and process noise. The transition from the k-1 state to the k state is defined as:

$$x_k = Fx_{k-1} + Bu_{k-1} + w_{k-1} \tag{2.9}$$

In the equation 2.9, the state vector is denoted by x, the state transition matrix is denoted by F, control matrix of the control vector input u is denoted by B and the zero mean Gaussian process noise is denoted by w [24]. Q is the covariance matrix of the process noise vector w_{k-1}.

If it is desired to associate the measurement and the state with each other at the current time step k, the association shown below in the Equation 2.10 is used.

$$z_k = Hx_k + v_k \tag{2.10}$$

In the Equation 2.10, the measurement model and the process model have put together. The measurement vector is denoted by z_k, measurement matrix is denoted by H, and the measurement noise is denoted by v_k. The measurement noise is Gaussian with zero mean. R is the covariance matrix of the measurement noise.

By using the measurement series $z_{1,2,\ldots,k}$, Kalman Filter makes estimations of x_k at time step k. The initial estimate x_0 is given in order to help the estimation process.The state transition matrix F, control input matrix B, measurement matrix H, process noise

covariance matrix Q and the measurement noise covariance matrix R are the matrices that characterize the system [24].

2.1.2 Algorithm

The prediction and update stages are two main branches of the Kalman Filter algorithm. In this section, the prediction and update stages are examined and it is explained how these two stages are used together and intertwined.

2.1.2.1 Prediction Stage

There are two outputs of the prediction stage: Predicted state estimate and predicted error covariance.

$$\hat{x}_k^- = F\hat{x}_{k-1}^+ + Bu_{k-1} \tag{2.11}$$

$$P_k^- = FP_{k-1}^+F^T + Q \tag{2.12}$$

The hat operator in the Equation 2.11, denotes the estimation. In Equations 2.11 and 2.12, the '-' superscript represents the predicted estimate and the '+' superscript represents the updated estimate. In the Equation 2.11, estimation of the x from the last update stage is used to make a new estimation of the x for a new prediction stage. Predicted error covariance P denotes the state error covariance according to the filter. Since P is aggregated with Q, it has a higher value in the prediction stage.

2.1.2.2 Update Stage

Update stage has four outputs, these are: measurement residual, Kalman gain, updated state estimate and updated error covariance.

$$\tilde{y}_k = z_k - H\hat{x}_k^- \tag{2.13}$$

$$K_k = P_k^-H^T(R + HP_k^-H^T)^{-1} \tag{2.14}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k\tilde{y}_k \tag{2.15}$$

$$P_k^+ = (I - K_k H)P_k^- \tag{2.16}$$

As examined in this section before, Kalman gain is calculated using measurements and errors to update the state estimate and error covariance. In the Equation 2.14, Kalman gain K is obtained at given time step k). In the Equation 2.13, measurement residual $\tilde{y}_k$ is obtained at given time step k. The goal of the measurement residual is to indicate the difference between the actual measurement and the estimated measurement. In the Equation 2.15, update of the state estimate $\hat{x}_k$ is made. A correction is generated by multiplying the Kalman gain K_k and the measurement residual $\tilde{y}_k$, then updated state estimate $\hat{x}_k^+$ is obtained by adding correction $K_k\tilde{y}_k$ to predicted state estimate $\hat{x}_k^-$ from the prediction state. In the Equation 2.16, error covariance P_k at given time step k is updated after updating the state estimate. In the update phase of the Kalman filter, the measured values were used and the existing noise values were brought together. Thus, Kalman gain was created, state estimate and error covariance were updated. With the use of measurement values, the state estimation produced by the Kalman filter has reached a more precise value. This can also be observed by the updated error covariance P_k^+ having a smaller value [24].

2.2 Extended Kalman Filter

2.2.1 Introduction

The Kalman Filter described in the previous section can only be used for linear systems. Since the Kalman Filter is based on the Gaussian distribution, the Gaussian distribution is preserved after a linear transformation [21]. However, real-world problems are mostly nonlinear, including the problem in this thesis. When a Gaussian distribution is applied to a nonlinear function, the output does not have a Gaussian distribution [25]. Thus, Extended Kalman Filter (EKF) is used when predicting nonlinear functions. In Extended Kalman Filter, system model is linearized around the estimation of the mean of the current state. The linearization is made by using Taylor series expansion [26]. This locally linearized model is used to give an approximation of the optimal prediction [25]. Navigation systems is one of the numerous real-time applications where Extended Kalman Filters are used. Compared to other non-linear filtration methods such as particle filters and point-mass filters, the Extended Kalman Filter is less expensive in terms of computational cost [24].

State transition and measurement models for EKF are:

$$x_k = f(x_{k-1}, u_{k-1}) + w_{k-1} \tag{2.17}$$

$$z_k = h(x_k) + v_k \tag{2.18}$$

Where f provides the current state x_k and is the function of x_{k-1} and u_{k-1}. The measurement function is denoted by h, z_k is the measurement and v_k is the measurement noise. In the Equation 2.18, w_{k-1} denotes the process noise. Q is the covariance matrix of the measurement noise w_{k-1} and R is the covariance matrix of the process noise v_k.

2.2.2 Algorithm

Jacobian matrices must be obtained to linearize the model around the current estimation. By obtaining the Jacobian matrices, first partial derivatives of the vectors can be derivable [24]. The Jacobians of the F and H matrices are obtained as shown in the Equations 2.19 and 2.20.

$$F_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}^+_{k-1}, u_{k-1}} \tag{2.19}$$

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}^-_k} \tag{2.20}$$

Rest of the algorithm is similar to Kalman Filter. In Extended Kalman Filter, there are also two stages; prediction stage and update stage as in Kalman Filter. The output of the previous update stage becomes the input to the prediction stage. With the outputs of the prediction stage, Kalman gain, and updated state estimates are calculated in the update stage. As in the Kalman Filter, state estimate $\hat{x}_k$ is updated using the Kalman gain K_k and measurement residual $\tilde{y}_k$. In this section, models of these prediction and update stages are examined.

2.2.2.1 Prediction Stage

Prediction stage is modeled as in the Equations 2.21 and 2.22. Predicted state estimate and predicted error covariance are obtained to use in the update stage.

$$\hat{x}_k^- = f(\hat{x}_{k-1}^+, u_{k-1}) \tag{2.21}$$

$$P_k^- = F_{k-1} P_{k-1}^+ F_{k-1}^T + Q \tag{2.22}$$

Where the hat operator means the estimate, '+' signifies the updated estimate, '-' signifies the predicted estimate, $\hat{x}_k^-$ is predicted state estimate and P_k^- is predicted error covariance [24].

2.2.2.2 Update Stage

Update stage is modeled as in the Equations 2.23, 2.24, 2.25 and 2.26. In the update stage, the predicted state estimate $\hat{x}_k^-$ and the predicted error covariance P_k^- are used to obtain the measurement residual $\tilde{y}_k$, Kalman gain K_k, updated state estimate $\hat{x}_k^+$ and updated error covariance P_k^+.

$$\tilde{y}_k = z_k - h(\hat{x}_k^-) \tag{2.23}$$

$$K_k = P_k^- H_k^T (R + H_k P_k^- H_k^T)^{-1} \tag{2.24}$$

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \tilde{y}_k \tag{2.25}$$

$$P_k^+ = (I - K_k H_k) P_k^- \tag{2.26}$$

Where y_k is measurement residual, K_k is Kalman gain, $\hat{x}_k^+$ is updated state estimate and P_k^+ is updated error covariance [24]. F and H are Jacobian matrices of f and h. The covariance matrix of process noise is represented by Q, and the covariance matrix of measurement noise is represented by R. In order to obtain detailed information about this topic, [24] can be studied.

2.3 Nonlinear Programming

2.3.1 Introduction

Before talking about nonlinear programming, it is helpful to examine linear programming. In linear programming, the objective function given as a mathematical model is tried to be minimized or maximized. It is aimed to make this optimization within the limits of a constraint set given in the form of linear relations [27]. In linear programming, real life problems are mathematically modeled to an objective function to get maximized or minimized.

Nonlinear Programming is an optimization problem solving method where the constraints or objective function are nonlinear [28]. In a nonlinear optimization problem, minimization or maximization of an objective function is made depending on a set of constraints, where these constraints can be equality or inequality constraints [29]. To solve nonlinear problems with constraints, there are numerous methods. Some of the methods are: interior-point, sequential quadratic programming (SQP) and trust-region reflective [29]. Large-scale nonlinear optimization problems with sparseness or structure are where the Interior-point method is particularly useful. Sequential Quadratic Programming (SQP), which is used as a general solution to nonlinear problems, takes constraints into account at each iteration [29]. The trust-region reflective method is used for solving linear and nonlinear problems where constraints are bounds only [29].

2.3.2 Formulation

The formulation for nonlinear programming used in this thesis is defined in Equations 2.27, 2.28, 2.29, 2.30, 2.31, and 2.32 [30].

$$\min_{x} f(X) \tag{2.27}$$

$$LB \leq X \leq UB \tag{2.28}$$

$$A_{eq} * X = b_{eq} \tag{2.29}$$

$$A * X \leq b \tag{2.30}$$

$$C_{eq}(X) = 0 \tag{2.31}$$

$$C(X) \leq 0 \tag{2.32}$$

In the Equation 2.27, $f(X)$ is the objective function. In the nonlinear programming method, the objective function $f(x)$ is tried to be minimized by taking into account the given constraints. In the Equation 2.28, terms LB and UB, denotes the lower and upper boundaries of the input X. This means that at each iteration, the minimum and maximum values that X, the input value of the objective function $f(X)$, can take are determined by LB and UB. In the Equation 2.29, A_{eq} and b_{eq} are linear equality constraints. A_{eq} and b_{eq} determine the restrictions in form of linear equality equations for the X. In the Equation 2.30, A and b are linear inequality constraints. A and b determine the restrictions in form of linear inequality equations for the X. In the Equation 2.31, $C_{eq}(X)$ is nonlinear equality function. $C_{eq}(X)$ determines the restrictions in form of nonlinear equality function for the X. In the Equation 2.32, $C(X)$ is nonlinear inequality function [30]. $C(X)$ determines the restrictions in form of nonlinear inequality function for the X.

2.4 3D Transformations

2.4.1 Introduction

In this study, three-dimensional transformation matrices are used to transfer three-dimensional points. In this section, three-dimensional rotation, translation and homogeneous transformation matrix is explained.

2.4.2 Translation

Moving an object or a point from one position to another position in a 2D or a 3D space is called translation. The point A is translated by x_t, y_t and z_t with the translation [31]:

$$(x, y, z) \longrightarrow (x + x_t, y + y_t, z + z_t). \tag{2.33}$$

2.4.3 Rotation

To rotate a point in a 2D or a 3D space, a rotation matrix can be applied to the point. A 2D point A is rotated counterclockwise direction by using angle $\theta \in [0, 2\pi)$ [31].

The rotation is mapped to $(x, y) \in A$ as:

$$(x, y) \longrightarrow (xcos\theta - ysin\theta, xsin\theta + ycos\theta). \tag{2.34}$$

The rotation matrix used for rotating 2D point A in counterclockwise direction by the angle $\theta \in [0, 2\pi)$ is shown as:

$$R(\theta) = \begin{pmatrix} cos\theta & -sin\theta \\ sin\theta & cons\theta \end{pmatrix}. \tag{2.35}$$

The transformation of the rotation matrix $R(\theta)$ applied on the points of A is expressed as:

$$\begin{pmatrix} xcos\theta - ysin\theta \\ xsin\theta + ycos\theta \end{pmatrix} = R(\theta) \begin{pmatrix} x \\ y \end{pmatrix}. \tag{2.36}$$

It should be noted that the rotation operation is applied around the origin. When inverse rotation is intended, the inverse rotation matrix to be applied is $R(-\theta)$[31].

When it comes to the 3D rotation process, things get more complicated. A 3-dimensional point can be rotated in three different axes. These axes are the x, y and z axes that are orthogonal to each other. These rotations are called pitch, roll, and yaw. Pitch, roll and yaw rotations used in avionics terminology can be seen in the Figure 2.2.

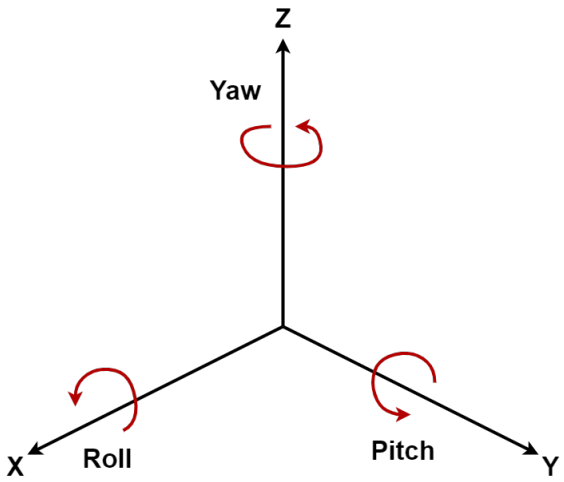

Figure 2.2 Pitch, yaw and roll rotations.

The rotation around the Y axis by an angle of β, counterclockwise is called pitch. The

rotation matrix $R_y(\beta)$ of pitch is shown in Equation 2.37.

$$R_y(\beta) = \begin{pmatrix} cos\beta & 0 & sin\beta \\ 0 & 1 & 0 \\ -sin\beta & 0 & cos\beta \end{pmatrix} \tag{2.37}$$

The rotation around the Z axis by an angle of α, counterclockwise is called yaw. The rotation matrix $R_z(\alpha)$ of yaw is shown in Equation 2.38.

$$R_z(\alpha) = \begin{pmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.38}$$

The rotation around the X axis by an angle of γ, counterclockwise is called roll. The rotation matrix $R_x(\gamma)$ of roll is shown in Equation 2.39.

$$R_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & cos\gamma & -sin\gamma \\ 0 & sin\gamma & cos\gamma \end{pmatrix} \tag{2.39}$$

Each rotation matrix shown above can be regarded as a extension of the 2D rotation matrix shown in Equation 2.35. The rotation matrix $R_z(\alpha)$ basically applies a 2D rotation on the X and Y coordinate axis leaving out the Z axis. The rotation matrix $R_y(\beta)$ basically applies a 2D rotation on the X and Z coordinate axis leaving out the Y axis. The rotation matrix $R_x(\gamma)$ basically applies a 2D rotation on the Y and Z coordinate axis leaving out the X axis [31]. By multiplying these three rotation matrices $R_z(\alpha)$, $R_y(\beta)$and $R_x(\gamma)$, a single uniformed rotation matrix can be created. The order in which these three matrices are multiplied with each other is important, when they are multiplied in different orders, a different resultant rotation matrix is obtained. For example, the yaw, pitch, and roll matrices are multiplied with each other, respectively. The resulting rotation matrix $R(\alpha,\beta,\gamma) = R_z(\alpha)R_y(\beta)R_z(\gamma)$ is shown in the Equation 2.40.

$$R(\alpha,\beta,\gamma) = \begin{pmatrix} cos\alpha cos\beta & cos\alpha sin\beta sin\gamma - sin\alpha cos\gamma & cos\alpha sin\beta cos\gamma + sin\alpha sin\gamma \\ sin\alpha cos\beta & sin\alpha sin\beta sin\gamma + cos\alpha cos\gamma & sin\alpha sin\beta cos\gamma - cos\alpha sin\gamma \\ -sin\beta & cos\beta sin\gamma & cos\beta cos\gamma \end{pmatrix} \tag{2.40}$$

To summarize 2D and 3D rotations, 3D rotation depends on α, β and γ parameters while 2D rotation depends on θ parameter only [31].

2.4.4 Homogeneous Transformation Matrices

The transformation of a point to another point in three-dimensional space can be expressed practically with the homogeneous transformation matrix [32]. The three-dimensional transformation matrix consists of the rotation matrix and the transformation vector. A homogeneous transformation matrix in 3D space is 4x4 dimensional. For example, let's take the rotation matrix $R(\alpha,\beta,\gamma) = R_z(\alpha)R_y(\beta)R_z(\gamma)$ in the Equation 2.40 and the x_t, y_t, z_t translation in the Equation 2.33 and create a homogeneous transformation matrix. The obtained T homogeneous transformation matrix is seen in the Equation 2.41.

$$T = \begin{pmatrix} cos\alpha cos\beta & cos\alpha sin\beta sin\gamma - sin\alpha cos\gamma & cos\alpha sin\beta cos\gamma + sin\alpha sin\gamma & x_t \\ sin\alpha cos\beta & sin\alpha sin\beta sin\gamma + cos\alpha cos\gamma & sin\alpha sin\beta cos\gamma - cos\alpha sin\gamma & y_t \\ -sin\beta & cos\beta sin\gamma & cos\beta cos\gamma & z_t \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{2.41}$$

In the T homogeneous transformation matrix in the formula, first the rotation process $R(\alpha,\beta,\gamma)$, and then the translation process x_t, y_t, z_t is applied. The order of operations is as follows: roll as γ, pitch as β, yaw as α and translate as x_t, y_t, z_t, respectively. A point A that is transformed has six degrees of freedom, including rotation in three axes and translation in three axes [31]. The T homogeneous transformation matrix can be shown as in the Equation 2.42.

$$T = \begin{pmatrix} R_{3x3} & P_{3x1} \\ 0_{1x3} & 1 \end{pmatrix} \tag{2.42}$$

In the matrix shown in Equation 2.42, R is the Rotation matrix $R(\alpha,\beta,\gamma)$ and P is the translation x_t, y_t, z_t. When the inverse of the T homogeneous transformation matrix is to be applied, the inverse T^{-1} matrix is shown in the Equation 2.43.

$$T^{-1} = \begin{pmatrix} R_{3x3} & -R^T P_{3x1} \\ 0_{1x3} & 1 \end{pmatrix} \tag{2.43}$$

3
PROPOSED METHOD

In this chapter, the proposed method of this work is examined. In this proposed method, The Extended Kalman Filter and nonlinear programming method are used together to develop the escape path estimation algorithm. MATLAB program is used to develop this proposed algorithm and make the necessary simulations [33]. A simple architecture of the proposed method is shown in the Figure 3.1 below:

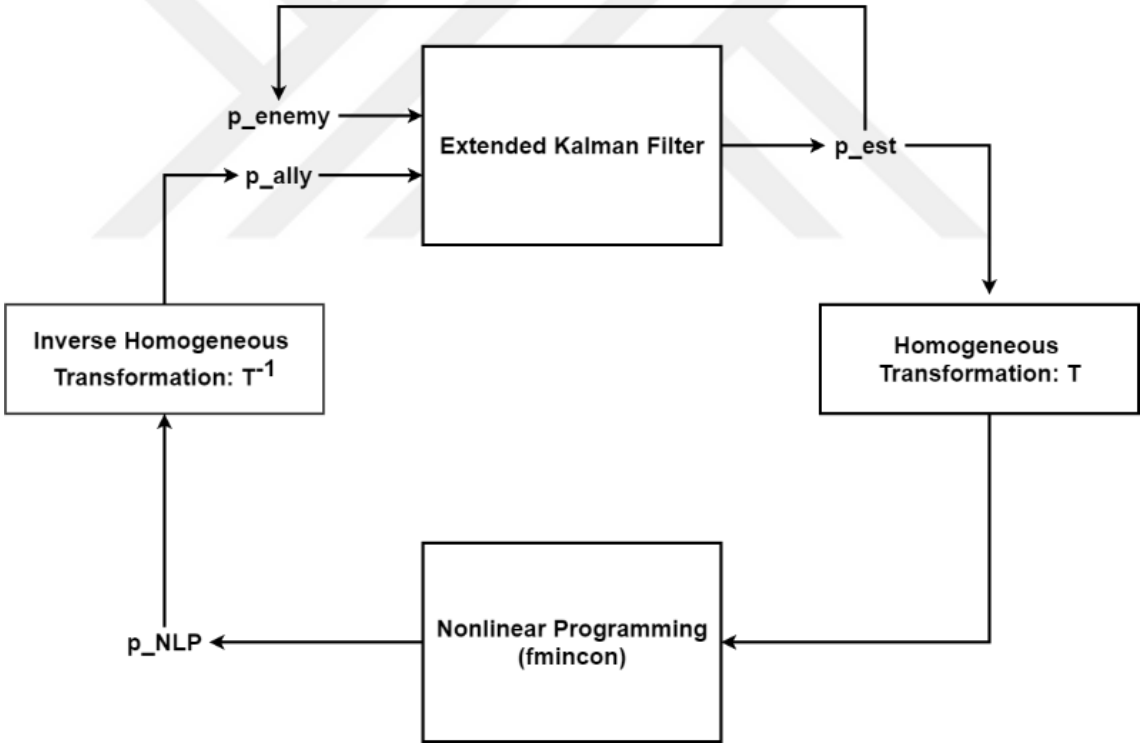

Figure 3.1 Overview of the proposed method

Where p_ally represents the position data of the friendly aircraft, which the escape path prediction algorithm is applied on. The position data of the attacking enemy aircraft that has been evaded is represented by p_enemy. The system calculates p_est and p_NLP for each time point, working in a loop for each time interval. “p_est” is the prediction value of the p_enemy position data obtained from The Extended Kalman Filter. This p_est prediction value is given as an input to the fmincon function [30].

Fmincon is a built-in MATLAB function for nonlinear constraint optimization. Fmincon outputs the optimal escape vector p_NLP. Adding p_NLP to the value p_ally had in the previous time interval gives the new value of p_ally at the current time. Thus, an optimal escape position for a time interval is assigned to the friendly aircraft based on the estimated position of the attacking aircraft. The coordinate axis of the ally UAV changes at each time step, as the ally UAV performs different maneuvers. For this reason, at each time step, the nonlinear programming constraints and solution must be relative to the ally UAV's coordinate system. The relative distance between the ally UAV and the attacking UAV undergoes a homogeneous transformation from the original coordinate system to the coordinate system of the ally UAV before entering the nonlinear programming method fmincon. Once an optimal solution is found, it must go through an inverse homogeneous transformation to be converted back to the original coordinate system.

3.1 Position Estimation

The Extended Kalman Filter model used in this thesis is a suitable model for the scenario in the thesis. As mentioned before on the previous chapter, compared to other non-linear filtration methods such as particle filters and point-mass filters, the Extended Kalman Filter is less expensive in terms of computational cost [24]. In this model, there are two sensors on the friendly aircraft: the range sensor and the angle sensor. The position of the attacking aircraft is estimated with the help of Extended Kalman Filter, using the distance and angle data detected from these sensors on the friendly aircraft under noise [24]. Standard deviation of process noise w_{k-1} and standard deviation of measurement noise v_k are given to the Extended Kalman Filter as inputs. Thus, the EKF could be tested under different noise conditions as desired. A detailed diagram of the Extended Kalman Filter is shown in Figure 3.2.

Initial points of the friendly and enemy air crafts are given in the initialization stage. In the prediction stage, predictions of state estimate and the error covariance are made as in equations 2.21 & 2.22. In the update stage, measurement residual and Kalman gain are calculated as in equations 2.23 & 2.24. State estimate and the error covariance are also updated as in equations 2.25 & 2.26.

In the Extended Kalman Filter model used in this study, the radar distance sensor and angle sensor of the ally UAV measure the distance and angle values of the enemy UAV. The distance sensor measures the distance r between two planes of the spherical coordinate system shown in Figure 3.3. The angle sensor measures the azimuth ϕ and polar θ angles shown in Figure 3.3. EKF makes the position estimation of the enemy

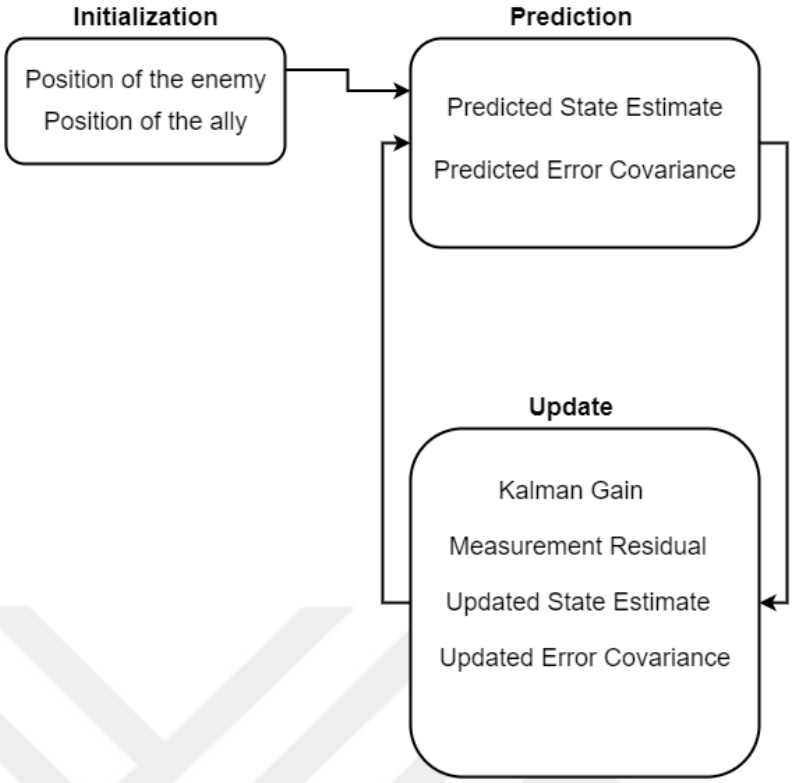

Figure 3.2 Extended Kalman Filter model

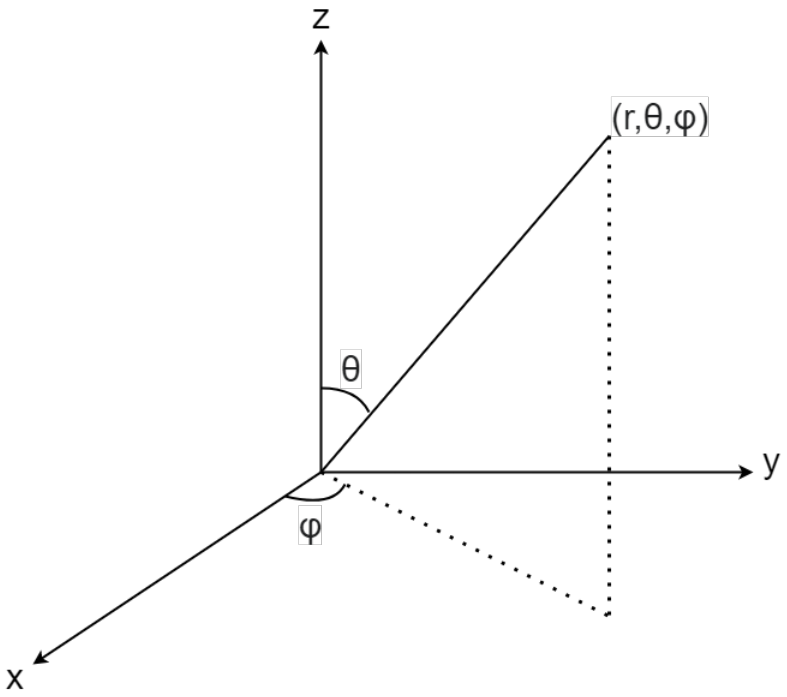

Figure 3.3 Spherical coordinate system

UAV using these distance and angle values.

3.2 Constrained Optimization

In this thesis, the built-in MATLAB function fmincon is used as a nonlinear programming method. Fmincon finds the minimum of a multi variable function with constraints. The constraints of the mentioned function and the objective function are specified for fmincon as stated in equations 2.27, 2.28, 2.29, 2.30, 2.31 and 2.32. Fmincon starts from a starting point x0 and tries to reach the value x that will bring the function with defined constraints to its minimum value [33]. The purpose of fmincon is to find a minimizer x value. The objective function used in this proposed method is an essential score function used in aviation to evaluate the relative distance and collision between two air crafts [34], [35]. The objective score function is defined as [8]:

$$Sc = (1 - \frac{|\epsilon + \lambda|}{\pi})(e^{-\frac{d - d_{opt}}{K\pi}}) \tag{3.1}$$

In the equation 3.1, Sc denotes score resulting from the positions and angles of the two air crafts relative to each other. The ϵ & λ represents the angles formed between the movement vectors of two aircraft and the LOS (Line of Sight) line. The relative ϵ & λ angles of the two air crafts are shown in Figure 3.4. The distance between two air crafts is denoted by d. Desired optimal distance is denoted by d_{opt}. Finally, the constant K is used to create a proportional adjustment between the angle and the distance. In this proposed method, d_{opt} value is 700 and K value of 600 is used based on an effective K value used in relative study [8].

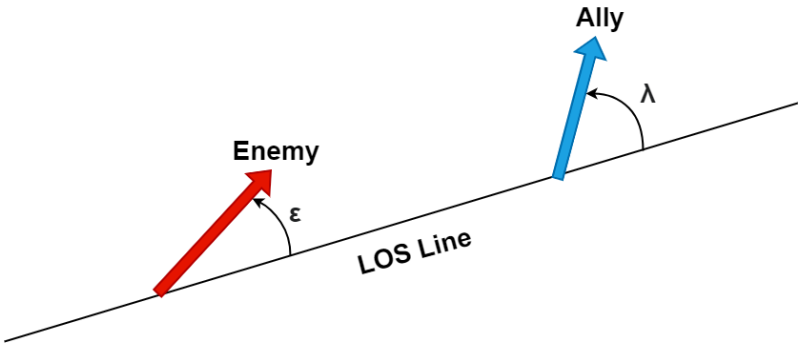

Figure 3.4 Relative angles of the two air crafts (adapted from [8])

3.2.1 Constraints

In the fmincon function used in Constrained Optimization, linear inequality constraints are used to give the escaping UAV a conical angle. In addition, non-linear inequality constraints are applied to constrain the resultant velocity vector of the escaping UAV. These constraints will be discussed later in this section.

3.2.1.1 Angular Constraints

Angular constraints have been applied to the Constrained Optimization function fmincon to ensure that the escaping plane's motion vector stays inside the circular sector with a desired angle. These constraints are applied for the angles of the x-axis of the motion vector with the y and z axes. The Figure 3.5 shows the motion vector remaining inside the representative circular sector.

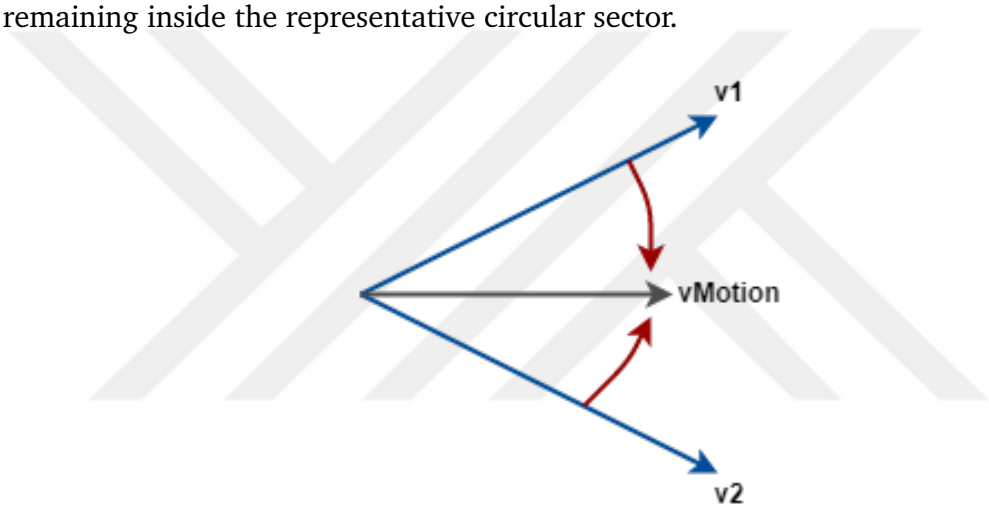

Figure 3.5 Motion vector inside the circular sector

It is aimed to ensure that vMotion, which is the motion vector shown in the Figure 3.5, stays between v1 and v2 vectors representing a circular sector with the desired angle. To do this, it is necessary to determine whether the motion vector vMotion stays clockwise relative to the v1 vector and counterclockwise relative to the v2 vector. To understand whether the vMotion vector remains clockwise relative to the v1 vector, the normal vector of the v1 vector is calculated and the dot product of the vMotion vector and the normal vector is taken. If the result from the dot product is positive, the vMotion vector is located counterclockwise relative to the v1 vector. If the result is negative, the vMotion vector is located clockwise relative to the v1 vector. The same process is applied to the v2 vector. With this method, the matrices required to apply the angular constraints to fmincon are determined. These matrices are the A and b matrices in Equation 2.30.

A and b matrices seen in Equation 2.30 are given to fmincon as in Equations 3.2 and

3.3, according to the angular constraint finding method described in this section.

$$A = \begin{pmatrix} -277 * sin\alpha & 277 * cos\alpha & 0 \\ -277 * sin\alpha & -277 * cos\alpha & 0 \\ -277 * sin\beta & 0 & 277 * cos\beta \\ -277 * sin\beta & 0 & -277 * cos\beta \end{pmatrix} \tag{3.2}$$

$$b = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.3}$$

In the Equation 3.2, 277 is the maximum resultant velocity constraint of input $X = x, y, z$ in terms of m/s. The angle α is the yaw angle between the x and y axis. The angle β is the pitch angle between the x and z axis.

3.2.1.2 Velocity Constraint

To represent the maximum speed the escaping UAV can reach, a resultant velocity constraint to be applied in fmincon is given. This constraint is a constraint in form of non-linear inequality. The non-linear inequality constraint C(X) is shown in Equation 3.4.

$$C(X) = X(1)^2 + X(2)^2 + X(3)^2 - (1000 * 0.2777778)^2 \leq 0 \tag{3.4}$$

In the Equation 3.4, X is the input of the coordinates x, y, z of the escaping UAV. The value 1000 is the maximum velocity of the escaping UAV in terms of km/h. The value 0.2777778 is the constant to convert km/h to m/s. It is desired to convert km/h to m/s since a time step in the simulation is 1 second.

3.3 Vector Transformation

After the prediction of the enemy aircraft position and direction, a constraint optimization was made. The constraints must be defined according to the coordinate axes of the escaping UAV. Since, UAV maneuverability can change according to the pitch, the roll, and the yaw axes of the UAV. Therefore, the nonlinear programming is applied on the coordinate axes of the escaping UAV. On the other hand, each time step the UAV can make a different maneuver, so the coordinate axes of the escaping

UAV are changed for each time step. The coordinate axes of the escaping UAV and the original coordinate axes can be managed in the algorithm in carefully. Therefore, the solution UAV position and direction found in nonlinear programming according to the coordinate axes of the escaping UAV must be translated and rotated with respect to the original coordinate axes. To achieve this coordinate axes translation, a similar approach like the one given in [36] is used in this thesis.

To find the direction of the UAV, the flight mechanism must be determined. In this thesis, the following approach is used for UAV maneuverability. The UAV has 3 axes: the pitch, the roll and the yaw. To change the altitude (the pitch control) the elevator is used. After the coming the desired altitude position, the UAV is updated its position as the flat position according to the ground. Thus, z axis in the UAV coordinate axes and z axis in the original coordinate axes are the same all nonlinear solution process. On the other hand, to arrive some points in 3-dimensional space, in addition to z axis movement, x and y axes movements are needed. It is assumed that the UAV has a high maneuverability capacity, and thus, to turn left or right the following the flight mechanism is used: Instead of the use of rudder for yaw control, the firstly the roll control is used by ailerons or flaperons and then using elevator the desired direction is aligned with the nose of the UAV. After the coming the desired direction position in x-y plane, the UAV is updated its position as the flat position according to the ground.

To further explain this mechanism, one scenario is explained. Let's the UAV only changes its direction to the left side according to the nose in x-y plane without changing the altitude position. In this case, the 1st command is the right aileron up and the left aileron down. The 2^{nd} command is the right aileron flat and the left aileron flat. The 3^{rd} command is the elevator up. The 4^{th} command is the elevator flat. The 5^{th} command is the right aileron down and the left aileron up. The 6^{th} command is the right aileron flat and left aileron flat. At the end of these commands, the UAV is flat position according to the ground and the direction of the UAV is changed an angle in x-y plane.

After the obtaining the solution according to the coordinate axes of the escaping UAV, the solution is converted to the original coordinate axes. In briefly, based on these two methods, an escape path prediction algorithm has been developed.

4
SIMULATIONS

In this chapter, the simulation scenarios are explained and the test results are examined. The simulation results are realized using MATLAB [33]. It has been assisted from [37] in the implementation of Extended Kalman Filter for selected scenario.

4.1 Scenarios

The selected scenarios are defined as follows: In the selected scenario, there are two UAVs traveling in open space, one ally and one hostile enemy. The Enemy UAV's starting point is assumed to be the point (0,0,0) in meters and moves at 1000km/h in the x-axis only. The initial point of the ally UAV is the point (500,0,0) in meters. Main input of the ally UAV for the target tracking is a radar system with range and angle measurements. The standard deviation of the process noise used in the Extended Kalman Filter is 0.5m/s for velocity in all three axes. Since we have the target position data of the hostile enemy UAV when this escape path prediction system is activated on the UAV, the EKF's initial guess of target position is the measured position of the enemy UAV.

Simulations were made for a period of 30 seconds. In nonlinear programming, the lower and upper boundaries for input X are given as follows for x, y, and z: LB = [-69.4445, -69.4445, -69.4445], UB = [277.7778, 69.4445, 69.4445] in meters. These values are the maximum and minimum meters the solution of the fmincon can be for each second. LB and UB were determined based on the mobility of the ally UAV and were obtained by converting 1000 km/h and 250 km/h to m/s.

In the first three scenarios, there are no linear equality or inequality constraints. The nonlinear inequality function $C(X)$ is the maximum resultant value that fmincon's solution in x, y and z combined. $C(X)$ is equal to the expression in Equation 3.4 where 1000 is the maximum value of velocity in km/h and 0.277778 is the constant value to convert km/h to m/s.

In the last three scenarios, linear inequality constraints are used as in the Equation 2.30 to constrain the movement angle of the motion vector in a desired angle. The linear inequality constraints used are equal to the expression in Equations 3.2 and 3.3. Values of the α and the β in Equation 3.2 are given $\pi/9$ radians.

4.2 Results

4.2.1 Scenario 1

First, the scenario where the user input of standard deviation of measurement noise is given [0.1, 0.1, 5] for two angles and distance respectively is run on MATLAB. The measurement covariance matrix is constructed by the noise characteristic [0.5, 0.5, 10]. The xy-axis trajectories of enemy and Ally UAVs are shown in Figure 4.1. Figure 4.2 shows the trajectories of UAVs in three dimensions. The values on the axes in the figures are in meters. As can be seen in Figure 4.1 and Figure 4.2, the ally UAV made its escape by making deviations in the y and z axes in the escape route. As the enemy UAV's position estimation sways in the z axis over time, the Ally UAV slopes to the z direction and escapes. Since the measurement noise characteristic values are low, Ally UAV escaping by drawing a trajectory close to ground truth. The x,y and z Position values for the ally UAV and the enemy UAV is shown in Figure 4.3 over time step k.

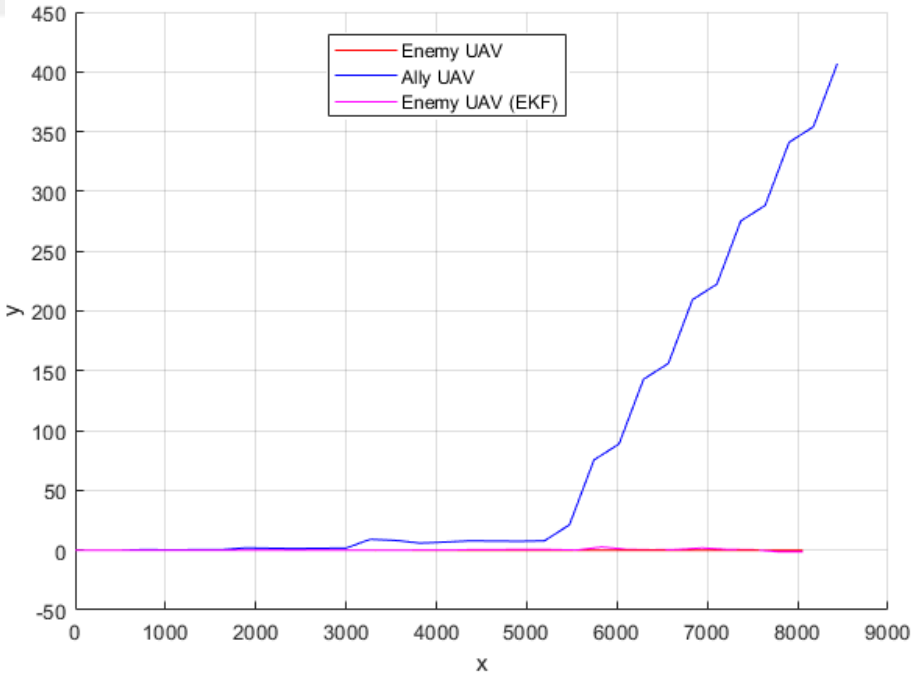

Figure 4.1 Xy-axis trajectory of the UAVs in the Scenario 1

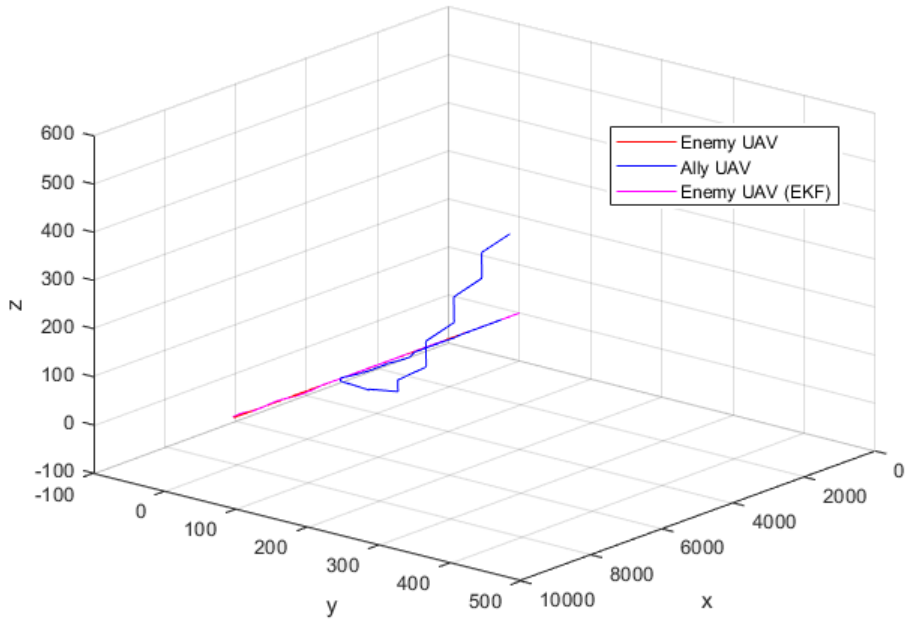

Figure 4.2 3D trajectory of the UAVs in the Scenario 1

	Ally UAV			Enemy UAV (Estimated)		
Time Step (k)	x	y	z	x	y	z
1	500,00	0,00	0,00	0,00	0,00	0,00
2	777,66	0,45	0,00	277,78	0,00	0,00
3	1055,43	0,23	0,00	555,56	0,00	0,00
4	1333,17	0,39	0,00	833,32	0,00	0,00
5	1610,76	0,39	0,00	1111,09	0,01	0,00
6	1887,90	1,92	0,00	1388,86	0,00	0,00
7	2165,68	1,62	0,01	1666,47	-0,01	0,00
8	2443,21	1,39	0,01	1943,93	-0,03	0,01
9	2720,99	1,61	0,02	2222,60	0,00	0,03
10	2998,76	1,72	0,06	2500,19	-0,03	0,07
11	3266,10	9,11	0,15	2777,39	-0,05	0,11
12	3543,40	8,20	0,30	3055,32	-0,04	0,07
13	3816,47	5,91	0,52	3332,92	-0,07	0,17
14	4094,25	6,79	0,98	3610,96	0,00	0,21
15	4372,01	7,81	1,69	3889,52	0,04	0,28
16	4649,79	7,60	2,90	4167,27	0,07	0,49
17	4916,93	7,46	5,01	4445,17	0,49	0,68
18	5194,68	7,80	8,56	4722,70	0,48	0,61
19	5472,08	21,18	14,18	5000,12	0,65	0,45
20	5744,36	75,46	22,92	5277,38	0,50	0,65
21	6021,45	88,83	37,28	5555,08	0,12	0,40
22	6292,97	142,96	59,83	5833,29	2,69	0,63
23	6568,02	156,22	96,30	6110,47	0,63	-2,67
24	6835,17	209,48	150,68	6388,34	0,30	-1,38
25	7104,80	222,49	216,16	6666,01	0,54	1,21
26	7369,55	275,27	281,63	6943,25	1,90	3,41
27	7639,18	288,27	347,10	7221,81	0,61	0,42
28	7903,93	341,05	412,58	7499,81	0,40	-0,69
29	8173,56	354,06	478,05	7777,53	-1,34	1,88
30	8438,31	406,84	543,52	8055,44	-1,30	1,36
31	8707,94	419,84	608,99	8333,90	-3,51	1,92

Figure 4.3 Ally UAV and enemy UAV positions over time step k from the Scenario 1

4.2.2 Scenario 2

The second scenario where the user input of standard deviation of measurement noise is given [0.5, 0.5, 10] for two angles and distance respectively is run on MATLAB. The measurement covariance matrix is constructed by the same noise characteristic [0.5, 0.5, 10]. The xy-axis trajectories of enemy and Ally UAVs are shown in Figure 4.4. Figure 4.5 shows the trajectories of UAVs in three dimensions. The values on the axes in the figures are in meters. As can be seen in Figure 4.4 and Figure 4.5, the ally UAV made its escape by making deviations in the y and z axes in the escape route. As the enemy UAV's position estimation sways in the z axis over time, the Ally UAV slopes

to the z direction and escapes. Since the measurement noise characteristic values in the second scenario were slightly higher than those in the first scenario, in Figure 4.5 it was observed that the ally UAV deviated slightly more in the y and z axes during escape. Position values for the ally UAV and the enemy UAV is shown in Figure 4.6 over time step k.

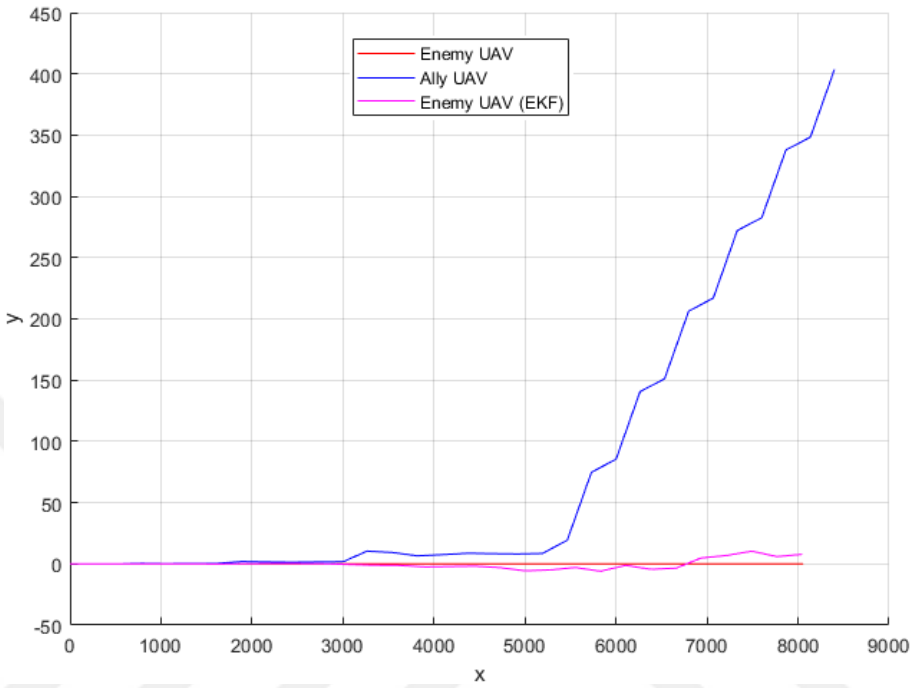

Figure 4.4 Xy-axis trajectory of the UAVs in the Scenario 2

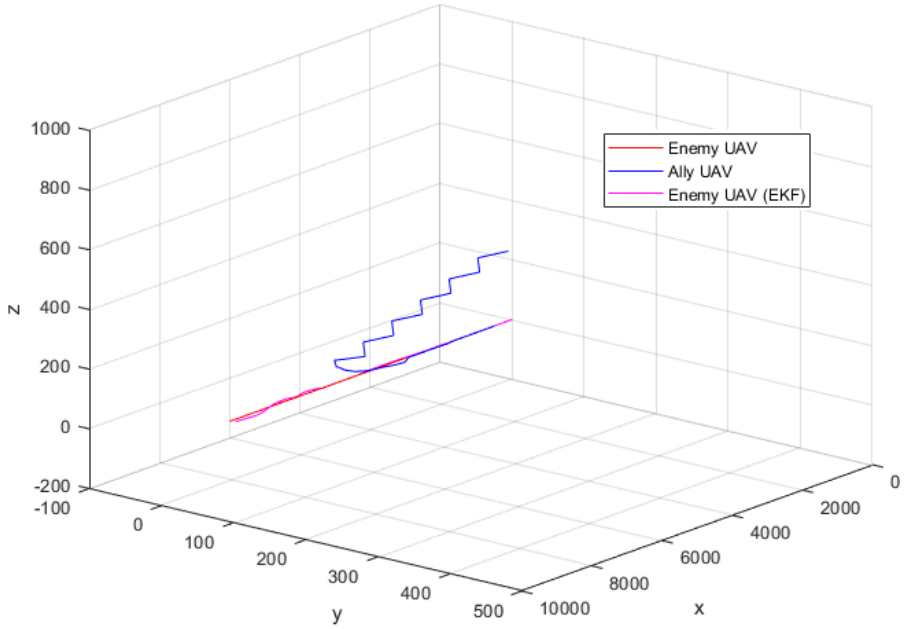

Figure 4.5 3D trajectory of the UAVs in the Scenario 2

	Ally UAV			Enemy UAV (Estimated)		
Time Step (k)	x	y	z	x	y	z
1	500,00	0,00	0,00	0,00	0,00	0,00
2	777,66	0,45	0,00	277,78	0,00	0,00
3	1055,43	0,23	0,00	555,53	0,00	0,00
4	1333,20	0,39	0,00	833,09	0,00	0,00
5	1610,98	0,39	0,00	1110,75	-0,02	0,01
6	1888,13	1,93	0,01	1387,88	0,00	0,05
7	2165,90	1,63	0,06	1665,21	-0,04	0,06
8	2443,67	1,39	0,14	1939,31	0,22	0,19
9	2721,45	1,65	0,38	2215,98	0,33	0,25
10	2999,22	1,78	0,77	2494,65	0,30	0,55
11	3264,96	10,41	1,68	2777,04	0,17	0,91
12	3542,30	9,33	3,16	3057,62	-0,45	1,19
13	3816,12	6,57	5,99	3336,61	-0,98	1,27
14	4093,86	7,56	10,25	3617,16	-1,19	1,72
15	4371,53	8,72	17,76	3895,18	-2,41	-0,01
16	4649,14	8,36	27,63	4170,66	-2,14	-1,73
17	4912,51	8,15	42,87	4445,12	-1,80	-2,78
18	5189,19	8,51	67,52	4722,75	-3,02	-4,48
19	5464,15	19,29	105,53	4997,35	-5,60	-8,13
20	5730,40	74,87	161,94	5274,84	-4,91	-8,37
21	6000,14	85,45	227,41	5552,74	-2,93	0,67
22	6264,40	140,60	292,89	5829,63	-5,96	-0,10
23	6534,14	151,19	358,36	6106,96	-1,17	-0,58
24	6798,40	206,34	423,83	6384,29	-4,41	1,19
25	7068,14	216,92	489,31	6660,63	-3,41	0,65
26	7332,40	272,08	554,78	6934,24	4,84	10,44
27	7602,14	282,66	620,25	7215,23	6,88	1,63
28	7866,40	337,82	685,72	7491,21	10,40	2,28
29	8136,15	348,40	751,20	7769,58	5,97	-1,32
30	8400,40	403,56	816,67	8044,53	7,79	3,34
31	8670,15	414,14	882,14	8322,25	5,95	0,58

Figure 4.6 Ally UAV and enemy UAV positions over time step k from the Scenario 2

4.2.3 Scenario 3

The third scenario where the user input of standard deviation of measurement noise is given [1.5, 1.5, 50] for two angles and distance respectively is run on MATLAB. The measurement covariance matrix is constructed by the noise characteristic [0.5, 0.5, 10]. Figure 4.7 and Figure 4.8 show the trajectories of UAVs in the xy-axis and three-dimensional axis. If we compare this scenario to previous scenarios, the standard deviation of measurement noise values in this scenario are higher than the values in the previous scenarios. Therefore, in this scenario, the estimation of the location of the enemy UAV with the EKF is more deviated. Thus, ally uses a more erroneous enemy

UAV trajectory estimate when calculating the UAV escape route. In this case, as the ally UAV experiences more confusion during escape, it flies in opposite directions over time in the y and z axes. A more inefficient escape route is observed in terms of results and resource use. Position values for the ally UAV and the enemy UAV is shown in Figure 4.9 over time step k.

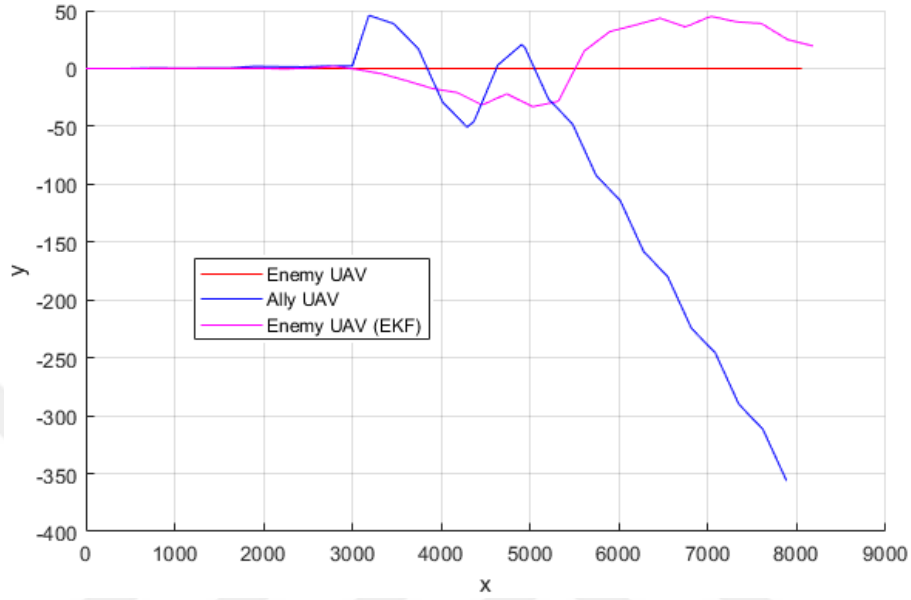

Figure 4.7 Xy-axis trajectory of the UAVs in the Scenario 3

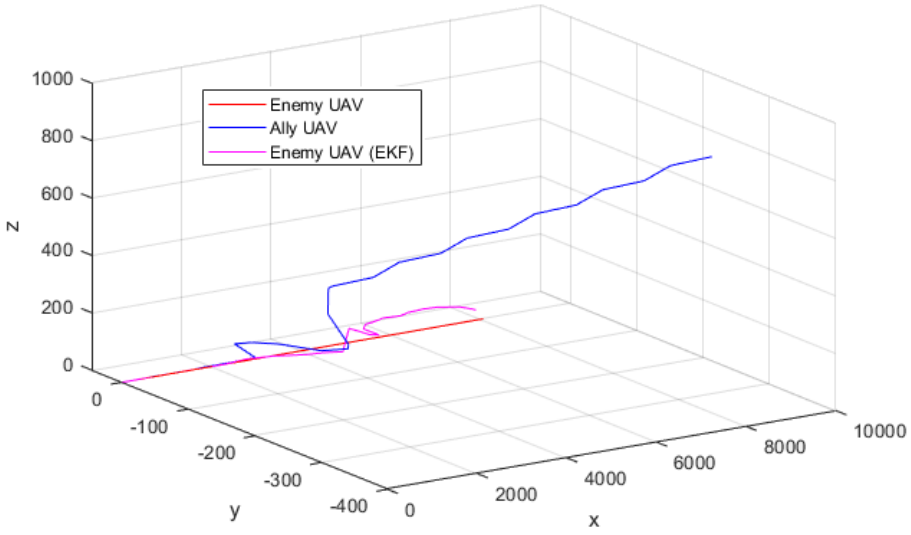

Figure 4.8 3D trajectory of the UAVs in the Scenario 3

	Ally UAV			Enemy UAV (Estimated)		
Time Step (k)	x	y	z	x	y	z
1	500	0	0	0	0	0
2	777,6596	0,445573	0,000161	277,7778	0	0
3	1055,425	0,234432	0,000161	555,7134	0,002834	-0,00543
4	1333,176	0,387659	-0,0051	833,7336	0,014495	-0,01259
5	1610,765	0,392536	-0,01752	1111,046	0,067347	0,052556
6	1887,859	1,777148	0,029867	1388,789	0,106269	0,279669
7	2165,637	1,514522	0,289711	1670,565	0,074334	0,272413
8	2442,702	1,312516	0,591538	1963,064	0,144322	0,108181
9	2720,063	2,044966	0,989008	2229,994	-0,46494	0,274154
10	2997,839	2,016741	1,846787	2519,119	0,255991	1,809671
11	3186,326	45,90916	3,921964	2800,172	1,862728	2,204304
12	3463,991	38,64025	7,797159	3066,234	-1,11773	3,359397
13	3740,828	16,95731	14,96609	3330,889	-4,64545	4,603462
14	4014,493	-29,1221	26,9852	3618,375	-10,8596	4,870627
15	4290,761	-50,7605	46,18362	3898,886	-17,3262	6,320817
16	4367,737	-45,7073	55,84857	4176,871	-20,8267	8,273326
17	4636,242	2,993505	107,7492	4458,251	-31,5548	13,43287
18	4905,649	20,67889	173,0769	4737,699	-21,9606	78,60227
19	4940,435	17,91824	182,0865	5029,192	-33,1158	59,08974
20	5206,685	-26,6303	247,5593	5316,326	-28,3395	45,44928
21	5475,79	-47,987	313,0322	5609,577	15,24827	17,65622
22	5742,041	-92,5356	378,505	5896,386	31,87329	9,857512
23	6011,146	-113,892	443,9778	6182,079	37,46578	7,069648
24	6277,396	-158,441	509,4507	6463,914	43,35536	6,234707
25	6546,502	-179,798	574,9235	6744,172	35,8019	13,34902
26	6812,752	-224,346	640,3964	7032,018	45,01782	7,704962
27	7081,857	-245,703	705,8692	7318,062	40,30005	16,95662
28	7348,107	-290,251	771,3421	7600,241	38,78892	17,01134
29	7617,213	-311,608	836,8149	7894,896	24,93322	23,00098
30	7883,463	-356,157	902,2878	8182,101	19,40634	9,939883
31	8152,568	-377,513	967,7606	8462,308	27,08373	16,23458

Figure 4.9 Ally UAV and enemy UAV positions over time step k from the Scenario 3

4.2.4 Scenario 4

The fourth scenario where the user input of standard deviation of measurement noise is given [0.1, 0.1, 5] for two angles and distance respectively is run on MATLAB. The measurement covariance matrix is constructed by the noise characteristic [0.5, 0.5, 10]. Figure 4.10 and Figure 4.11 show the trajectories of UAVs in the xy-axis and three-dimensional axis. The user input of standard deviation of measurement noise of this scenario is the same as in the first scenario. The difference between this scenario and the first scenario is that, in this scenario, angular constraints are applied to fmincon as linear inequalities. With the application of angular constraints,

the movements of the escaping UAV for each time step are more constrained than in the first scenario. The turns of the escaping UAV for each time step are also limited by the angles determined in these constraints.

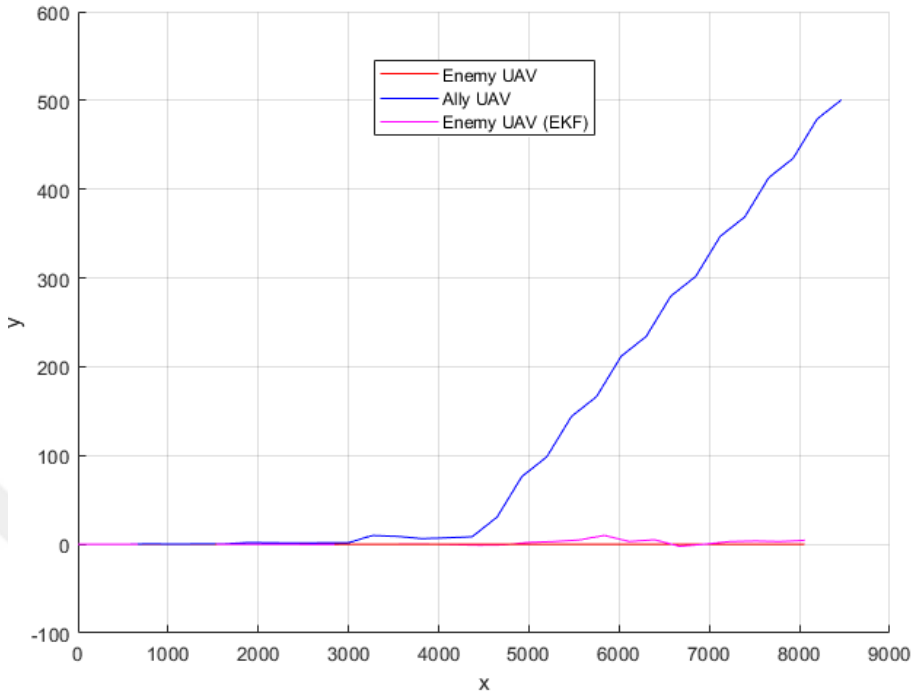

Figure 4.10 Xy-axis trajectory of the UAVs in the Scenario 4

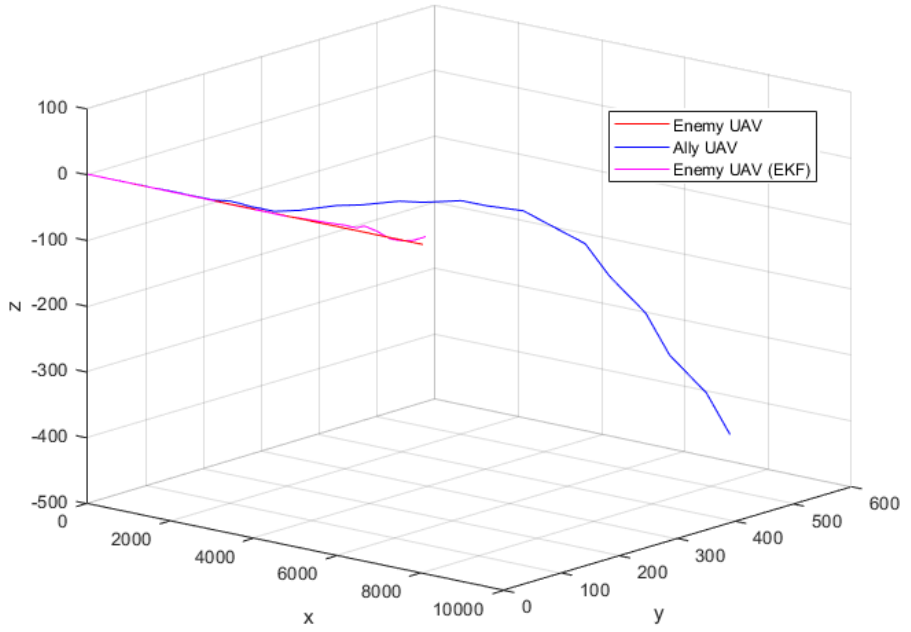

Figure 4.11 3D trajectory of the UAVs in the Scenario 4

	Ally UAV			Enemy UAV (Estimated)		
Time Step (k)	x	y	z	x	y	z
1	500	0	0	0	0	0
2	777,6596	0,445573	0,000161	277,7778	0	0
3	1055,425	0,234432	0,000161	555,541	0,000112	8,27E-05
4	1333,196	0,386897	0,000405	833,2626	0,000667	-0,00154
5	1610,78	0,389356	-0,00097	1110,968	6,66E-05	-0,00194
6	1887,914	1,939572	-0,0025	1388,897	0,00291	-0,006
7	2165,692	1,630106	-0,00939	1666,635	-0,00818	0,004269
8	2443,222	1,397666	-0,00763	1945,316	0,000223	0,024574
9	2721	1,625985	0,007551	2222,42	-0,03453	0,025477
10	2998,777	1,737082	0,025323	2499,572	-0,07941	-0,0004
11	3264,253	9,924532	0,032212	2776,532	-0,10413	0,031379
12	3541,571	8,906683	0,082056	3054,278	-0,03086	-0,1006
13	3814,651	6,333752	0,014671	3332,129	0,008259	-0,2922
14	4092,427	7,319005	-0,18865	3610,06	0,159799	-0,53191
15	4370,202	8,46079	-0,69107	3889,545	0,139706	-0,47713
16	4647,088	30,68605	-1,35449	4166,648	-0,377	-0,24468
17	4921,097	76,27904	-2,26229	4443,692	-0,9462	-0,02103
18	5197,982	98,50413	-3,59672	4721,634	-0,63481	-0,42767
19	5471,98	144,0954	-6,20626	5000,361	2,102531	-0,40485
20	5748,84	166,3185	-10,088	5278,517	3,087489	0,199324
21	6022,79	211,9017	-15,9143	5556,701	4,831907	0,119013
22	6299,51	234,1136	-25,5797	5835,361	10,0063	-0,88466
23	6573,05	279,6286	-41,8491	6112,962	3,095216	2,96028
24	6849,077	301,7849	-63,725	6390,749	5,127887	1,799732
25	7120,446	346,9386	-102,204	6669,818	-2,32024	10,50998
26	7392,672	368,7898	-152,961	6947,291	0,098471	5,510704
27	7658,962	413,0985	-218,434	7223,826	3,008207	-3,25684
28	7928,048	434,6976	-283,907	7501,184	3,706389	-2,67151
29	8194,338	479,0063	-349,38	7779,921	3,065072	1,792908
30	8463,424	500,6055	-414,852	8058,488	4,451855	10,97613
31	8729,715	544,9142	-480,325	8336,528	4,707497	13,70507

Figure 4.12 Ally UAV and enemy UAV positions over time step k from the Scenario 4

4.2.5 Scenario 5

The fifth scenario where the user input of standard deviation of measurement noise is given [0.5, 0.5, 10] for two angles and distance respectively is run on MATLAB. The measurement covariance matrix is constructed by the noise characteristic [0.5, 0.5, 10]. Figure 4.13 and Figure 4.14 show the trajectories of UAVs in the xy-axis and three-dimensional axis. The user input of standard deviation of measurement noise of this scenario is the same as in the second scenario. The difference between this scenario and the second scenario is that, in this scenario, angular constraints are applied to fmincon as linear inequalities. With the application of angular constraints,

the movements of the escaping UAV for each time step are more constrained than in the second scenario. The turns of the escaping UAV for each time step are also limited by the angles determined in these constraints.

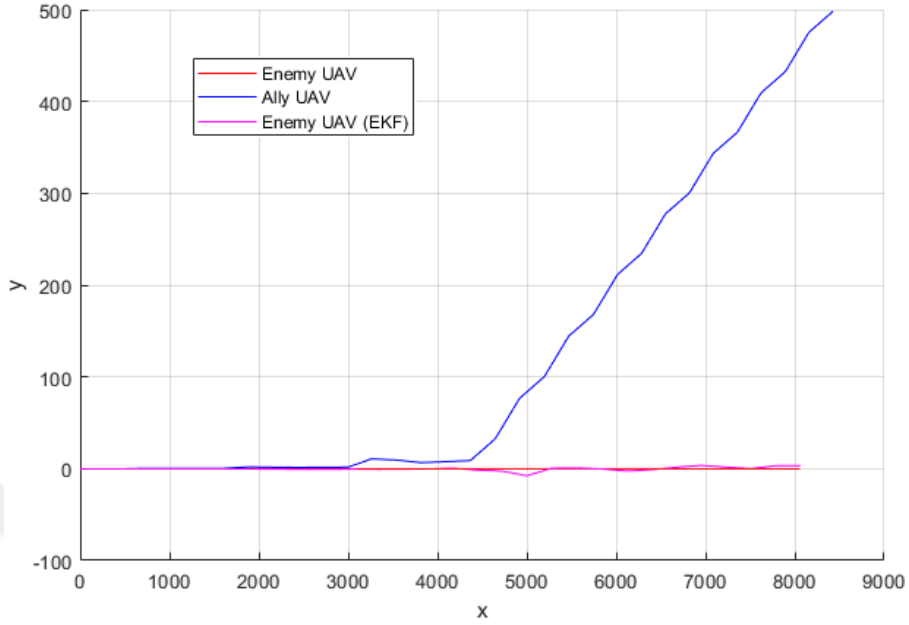

Figure 4.13 Xy-axis trajectory of the UAVs in the Scenario 5

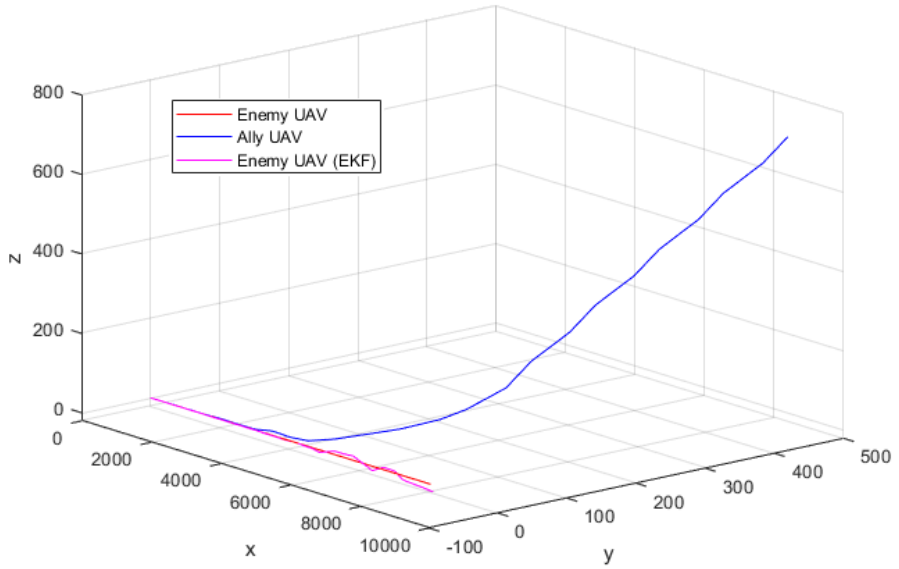

Figure 4.14 3D trajectory of the UAVs in the Scenario 5

	Ally UAV			Enemy UAV (Estimated)		
Time Step (k)	x	y	z	x	y	z
1	500	0	0	0	0	0
2	777,6596	0,445573	0,000161	277,7778	0	0
3	1055,425	0,234432	0,000161	555,5651	0,001864	0,002372
4	1333,174	0,386572	0,002694	833,4921	0,000361	0,009746
5	1610,758	0,389024	0,012587	1111,079	0,002136	0,01323
6	1887,899	1,928968	0,0285	1388,657	-0,05704	0,021496
7	2165,677	1,612211	0,062242	1665,692	-0,17607	0,018494
8	2443,286	1,364747	0,109236	1943,09	-0,33883	-0,00542
9	2721,063	1,566253	0,166025	2219,392	-0,49	0,210892
10	2998,839	1,654378	0,485089	2494,842	-0,70496	0,317685
11	3260,754	10,67394	0,946565	2771,646	-0,74766	0,531317
12	3538,086	9,501127	1,843751	3052,08	-0,42233	0,111381
13	3812,909	6,590967	2,892674	3334,256	-0,58221	-0,63373
14	4090,691	7,658706	4,051494	3609,656	-0,46304	-0,57825
15	4368,457	8,903995	6,298074	3887,876	-0,17252	-1,26979
16	4645,218	32,47877	9,068512	4169,994	0,646419	-0,07659
17	4919,383	76,72417	15,10282	4445,477	-1,53135	0,027192
18	5196,019	100,2882	23,9225	4723,49	-2,63378	0,299412
19	5469,853	144,4802	38,84328	5000,99	-7,6429	0,539792
20	5745,624	167,9706	62,48904	5281,691	0,582422	7,952369
21	6016,263	211,647	107,2931	5560,448	0,805651	8,22537
22	6285,24	234,5587	172,7659	5839,239	-0,15669	10,74817
23	6551,744	277,5676	238,2387	6118,315	-2,56424	4,117313
24	6820,721	300,4794	303,7116	6396,731	-1,05019	-10,3867
25	7087,224	343,4883	369,1844	6672,35	1,570748	4,312302
26	7356,202	366,4	434,6573	6949,866	3,628464	2,029938
27	7622,705	409,409	500,1301	7231,655	1,705722	-12,2116
28	7891,683	432,3207	565,603	7511,273	0,067611	-12,6159
29	8158,186	475,3297	631,0758	7788,218	3,345718	-17,8013
30	8427,163	498,2414	696,5487	8065,924	3,374817	-20,5133
31	8693,667	541,2504	762,0215	8337,67	10,36332	-18,1577

Figure 4.15 Ally UAV and enemy UAV positions over time step k from the Scenario 5

4.2.6 Scenario 6

The sixth scenario where the user input of standard deviation of measurement noise is given [1.5, 1.5, 50] for two angles and distance respectively is run on MATLAB. The measurement covariance matrix is constructed by the noise characteristic [0.5, 0.5, 10]. Figure 4.16 and Figure 4.17 show the trajectories of UAVs in the xy-axis and three-dimensional axis. The user input of standard deviation of measurement noise of this scenario is the same as in the third scenario. The difference between this scenario and the third scenario is that, in this scenario, angular constraints are applied to fmincon as linear inequalities. With the application of angular constraints,

the movements of the escaping UAV for each time step are more constrained than in the third scenario. The turns of the escaping UAV for each time step are also limited by the angles determined in these constraints.

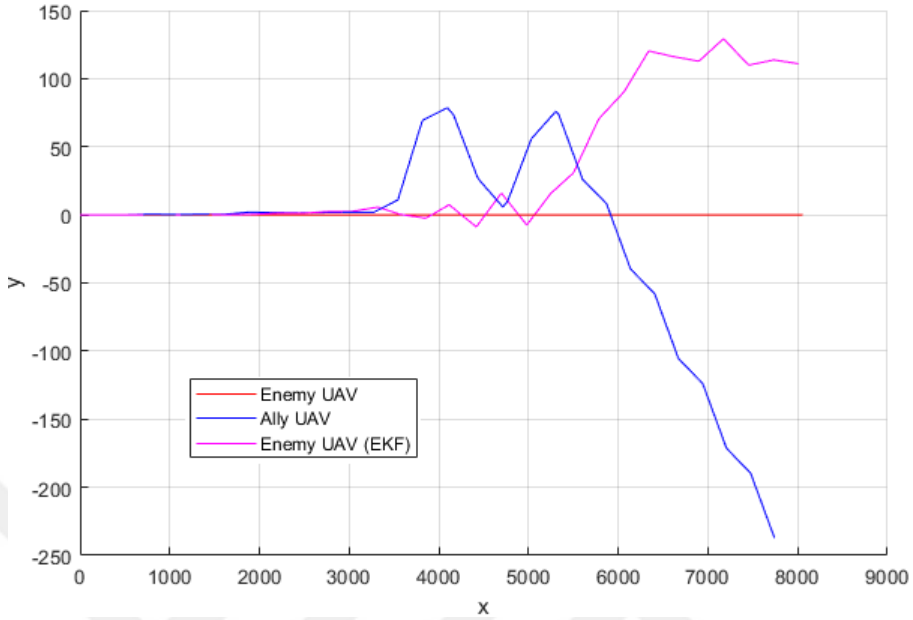

Figure 4.16 Xy-axis trajectory of the UAVs in the Scenario 6

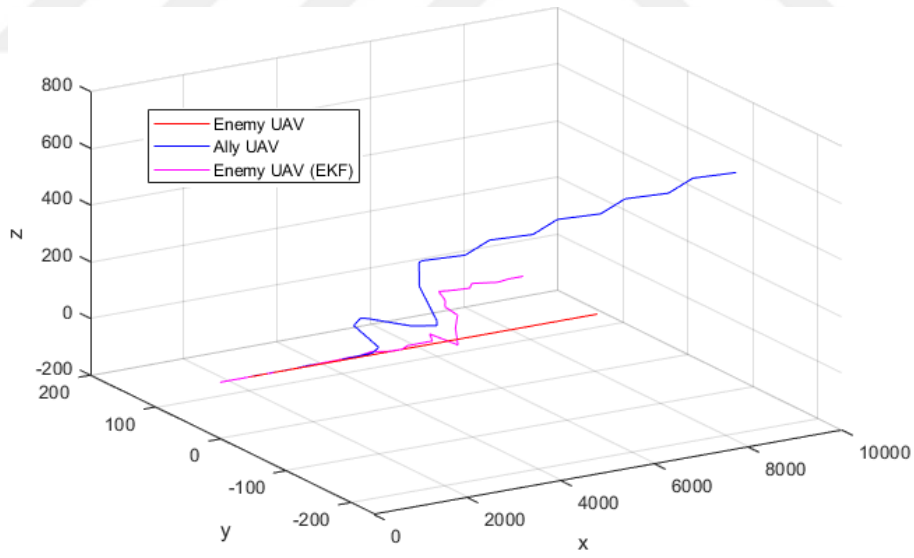

Figure 4.17 3D trajectory of the UAVs in the Scenario 6

	Ally UAV			Enemy UAV (Estimated)		
Time Step (k)	x	y	z	x	y	z
1	500	0	0	0	0	0
2	777,6596	0,445573	0,000161	277,7778	0	0
3	1055,425	0,234432	0,000161	555,5555	-0,00178	-0,00615
4	1333,203	0,386341	-0,00583	834,0418	0,022401	-0,01727
5	1610,795	0,393935	-0,02292	1112,909	-0,00505	-0,01106
6	1887,879	1,973095	-0,03979	1398,214	0,108408	-0,03898
7	2165,656	1,675752	-0,10109	1671,987	0,046867	0,071451
8	2443,124	1,432066	-0,06943	1941,497	0,386056	0,37444
9	2720,902	1,70488	0,203787	2217,211	1,258686	1,101784
10	2998,678	1,931937	1,229439	2487,15	1,215441	1,146885
11	3276,452	1,947768	2,570447	2747,545	2,406994	1,042635
12	3544,032	11,08885	4,775958	3034,241	2,535575	0,917297
13	3815,639	69,21534	8,159313	3317,735	5,647234	2,658128
14	4093,161	78,69612	15,369	3573,771	0,313013	1,205949
15	4162,158	73,49995	17,81493	3854,41	-2,43484	4,215123
16	4435,303	26,66627	36,77004	4113,793	7,422303	-1,87544
17	4711,868	5,83808	52,20972	4416,916	-8,83599	23,73478
18	4764,284	9,711506	64,15878	4696,841	15,96684	12,5173
19	5031,264	55,87446	125,4077	4979,291	-7,46344	-8,80403
20	5303,25	75,97351	178,1334	5248,676	15,82707	15,92649
21	5334,757	73,83828	186,2914	5499,321	30,78274	38,49018
22	5600,476	26,22275	251,7642	5781,665	70,40335	13,76384
23	5869,81	7,970306	317,2371	6064,39	90,38144	6,475594
24	6135,529	-39,6452	382,7099	6338,534	120,2736	-4,45887
25	6404,863	-57,8977	448,1827	6617,273	116,1135	-3,05397
26	6670,582	-105,513	513,6556	6897,917	112,791	-0,61197
27	6939,915	-123,766	579,1284	7172,588	129,2962	-10,807
28	7205,634	-171,381	644,6013	7453,022	109,8487	6,697094
29	7474,968	-189,634	710,0741	7732,638	113,7455	5,392238
30	7740,687	-237,249	775,547	8008,919	110,9265	9,805948
31	8010,021	-255,502	841,0198	8278,907	128,374	-18,1271

Figure 4.18 Ally UAV and enemy UAV positions over time step k from the Scenario 6

4.2.7 Ground Truth

The ground truth calculation was made by calculating the escape route for the ally UAV according to the actual position of the Enemy UAV, not according to the estimated position on which EKF was applied. In other words, constrained optimization method NLP is applied based on real trajectory of the enemy UAV In order to make a comparison with the allied UAV's escape trajectories that planned according to estimated enemy UAV positions by applying EKF, ground truth version of the allied UAV's escape trajectory can be seen in Figure 4.19 & Figure 4.20. Position values for the ally UAV and the enemy UAV is shown in Figure 4.21 over time step k.

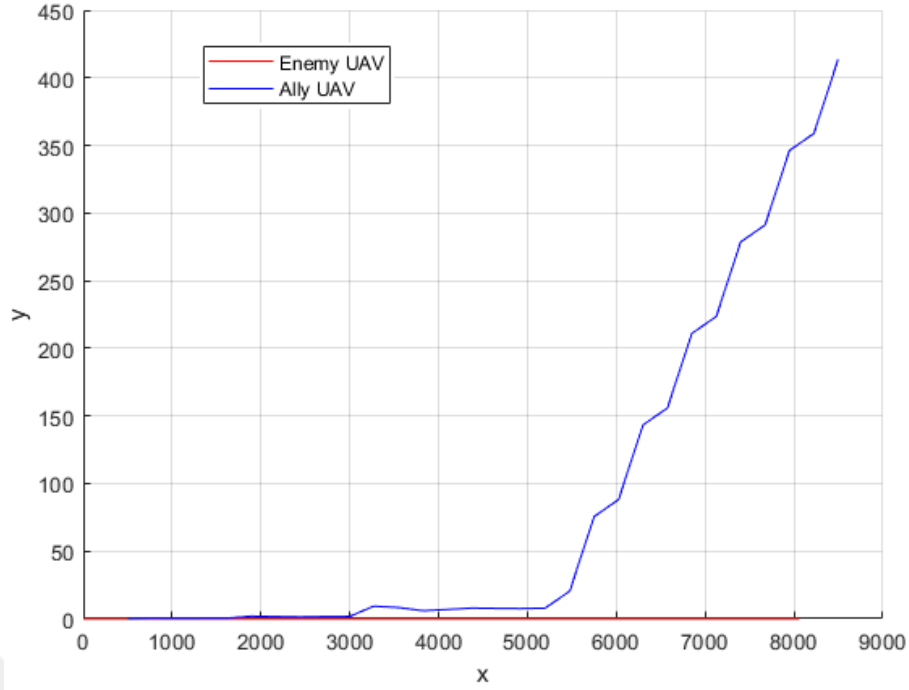

Figure 4.19 Xy-axis trajectory of the UAVs in the ground truth

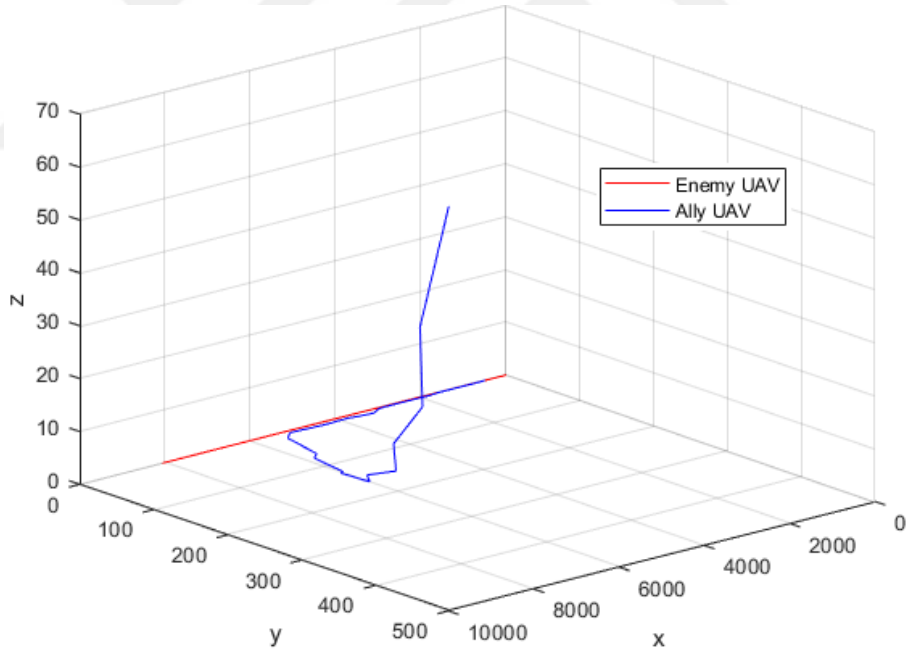

Figure 4.20 3D trajectory of the UAVs in the ground truth

	Ally UAV			Enemy UAV		
Time Step	x	y	z	x	y	z
1	500	0	0	0	0	0
2	777,6596	0,445573	0,000161	277,7778	0	0
3	1055,425	0,234432	0,000161	555,5556	0	0
4	1333,203	0,386379	0,000322	833,3334	0	0
5	1610,79	0,388675	0,000483	1111,111	0	0
6	1887,933	1,931647	0,000805	1388,889	0	0
7	2165,693	1,62338	0,001284	1666,667	0	0
8	2443,221	1,393867	0,002089	1944,445	0	0
9	2720,999	1,620518	0,003373	2222,222	0	0
10	2998,773	1,734324	0,005454	2500	0	0
11	3265,823	9,283878	0,008737	2777,778	0	0
12	3543,129	8,358878	0,013531	3055,556	0	0
13	3820,896	5,964903	0,022286	3333,334	0	0
14	4098,672	6,869783	0,035391	3611,111	0	0
15	4376,448	7,906255	0,057089	3888,889	0	0
16	4654,226	7,683815	0,091233	4166,667	0	0
17	4921,61	7,538183	0,146596	4444,445	0	0
18	5199,385	7,860863	0,237079	4722,223	0	0
19	5476,873	20,54799	0,383515	5000	0	0
20	5749,153	75,54033	0,613522	5277,778	0	0
21	6026,641	88,22746	0,985592	5555,556	0	0
22	6298,92	143,2197	1,580788	5833,334	0	0
23	6576,406	155,9068	2,536954	6111,112	0	0
24	6848,682	210,8983	4,070517	6388,889	0	0
25	7126,159	223,5849	6,531623	6666,667	0	0
26	7398,411	278,5717	10,4802	6944,445	0	0
27	7675,827	291,2556	16,81516	7222,223	0	0
28	7947,925	346,2112	26,97565	7500,001	0	0
29	8224,935	358,8765	43,2607	7777,778	0	0
30	8496,015	413,6264	69,31554	8055,556	0	0
31	8770,389	426,1712	110,8136	8333,334	0	0

Figure 4.21 Ally UAV and enemy UAV positions over time step k from the ground truth

5
RESULTS AND DISCUSSION

In this study, an escape path prediction algorithm that combines Extended Kalman Filter and Nonlinear Programming is presented. Extended Kalman Filter is used as the estimation method since the Extended Kalman Filter is less expensive in terms of computational cost compared to other non-linear filtration methods such as point-mass filters and particle filters [24]. Nonlinear Programming is used for the constraint optimization. Coordinate system transformations are made for the UAV positions by using homogeneous transformation matrices. Distance and angle measurements coming from allied UAV's sensors and standard deviation values for measurement noise are used as inputs for Extended Kalman Filter. Thus, a sensor fusion and escape path planning method has been developed

The algorithm was implemented on MATLAB and the simulations of the scenarios were made on MATLAB. A ground truth of the escape path of allied UAV is calculated. In the ground truth, escape path planned by doing constrained optimization based on real position of the enemy UAV, not the estimations obtained from Extended Kalman Filter.

As a result of the simulations, it has been observed that different standard deviation of measurement noise values cause different results in EKF and accordingly nonlinear programming finds different solutions. For lower values of standard deviation of measurement noise, escape path of the allied UAV becomes more alike with the ground truth. For higher values of standard deviation of measurement noise, escape path of the allied UAV becomes more inefficient in terms of results and resource use. Thus, the quality of radars in UAVs can be an important factor for an artificial intelligence based algorithms like the given in this thesis.

Additionally, the results of scenarios with angular constraints applied and scenarios without angular constraints applied were compared. The comparison is made between the scenarios with the same standard deviation of measurement noise values. It has been observed that, with angular constraints applied considering a desired angle, the

UAV model has more realistic constraints and moves closer to reality.

However, the model used in this study does not have enough inputs to make the inference to determine the end of the engagement. Determining the end of the engagement depends on factors such as the effect of the distance between the two aircraft on the measurements from the radars, the type of enemy, the type of engagement. Besides of these, the most important factor in determining the termination of engagement is knowing the type of missile on the enemy aircraft and how the enemy aircraft will use the missiles on it. Since the enemy missiles was not modeled in this study, the determination of the end of the engagement was not made. However, this process may be added in future studies.

For future studies, more efficient and more flexible Extended Kalman Filter models for various situations can be developed. Also, a system model can be developed in which enemy missiles are analyzed and modeled.

REFERENCES

[1] T. N. Dupuy, *Understanding war : history and theory of combat*. London Leo Cooper, 1992.

[2] A. DOBRESCU, M. OLTEAN, G. UDEANU, "Unmanned aerial vehicle in military operations," *SCIENTIFIC RESEARCH AND EDUCATION IN THE AIR FORCE*, vol. 18, pp. 199–206, 2016. DOI: 10.19062/2247-3173.2016.18.1.26. [Online]. Available: http://www.afahc.ro/ro/afases/2016/RP/UDEANU_DOBRESCU_OLTEAN.pdf (visited on 12/01/2019).

[3] D. Glade, L. Col, *Unmanned aerial vehicles: Implications for military operations*, 2000. [Online]. Available: https://apps.dtic.mil/sti/pdfs/ADA425476.pdf.

[4] *Anka - tusas*, www.tusas.com. [Online]. Available: https://www.tusas.com/en/products/uav/operative-strategic-uav-systems/anka (visited on 01/05/2022).

[5] H. Chen, X.-m. Wang, Y. Li, "A survey of autonomous control for uav," *2009 International Conference on Artificial Intelligence and Computational Intelligence*, 2009. DOI: 10.1109/AICI.2009.147. [Online]. Available: https://www.semanticscholar.org/paper/A-Survey-of-Autonomous-Control-for-UAV-Chen-Wang/19cdff0601a2c52f7a8ff0ecae45649880652e9b (visited on 10/06/2021).

[6] S. Y. Choi, D. Cha, "Unmanned aerial vehicles using machine learning for autonomous flight; state-of-the-art," *Advanced Robotics*, vol. 33, pp. 265–277, Mar. 2019. DOI: 10.1080/01691864.2019.1586760. (visited on 08/10/2020).

[7] F. Cappello, R. Sabatini, S. Ramasamy, M. Marino, "Particle filter based multi-sensor data fusion techniques for rpas navigation and guidance," *2015 IEEE Metrology for Aerospace (MetroAeroSpace)*, Jun. 2015. DOI: 10.1109/metroaerospace.2015.7180689.

[8] N. Ramírez López, R. Żbikowski, "Effectiveness of autonomous decision making for unmanned combat aerial vehicles in dogfight engagements," *Journal of Guidance, Control, and Dynamics*, vol. 41, pp. 1021–1024, Apr. 2018. DOI: 10.2514/1.g002937. (visited on 10/16/2020).

[9] Z. Zhang, J. Li, J. Wang, "Sequential convex programming for nonlinear optimal control problems in uav path planning," *Aerospace Science and Technology*, vol. 76, pp. 280–290, May 2018. DOI: 10.1016/j.ast.2018.01.040. (visited on 01/04/2022).

[10] J. Tisdale, Z. Kim, J. Hedrick, "Autonomous uav path planning and estimation," *IEEE Robotics & Automation Magazine*, vol. 16, pp. 35–42, Jun. 2009. DOI: 10.1109/mra.2009.932529. (visited on 01/04/2022).

[11] S. Bortoff, *Path planning for uavs*, IEEE Xplore, Jun. 2000. DOI: 10.1109/ACC.2000.878915. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=878915 (visited on 01/04/2022).

[12] H. Jiang, Y. Liang, "Online path planning of autonomous uavs for bearing-only standoff multi-target following in threat environment," *IEEE Access*, vol. 6, pp. 22 531–22 544, 2018. DOI: 10.1109/access.2018.2824849. (visited on 07/10/2020).

[13] Q. Yang, J. Zhang, G. Shi, "Path planning for unmanned aerial vehicle passive detection under the framework of partially observable markov decision process," *2018 Chinese Control And Decision Conference (CCDC)*, Jun. 2018. DOI: 10.1109/ccdc.2018.8407800. (visited on 01/05/2022).

[14] M. Kang, Y. Liu, Y. Zhao, "A threat modeling method based on kalman filter for uav path planning," *2017 29th Chinese Control And Decision Conference (CCDC)*, May 2017. DOI: 10.1109/ccdc.2017.7979170. (visited on 01/05/2022).

[15] Z. Wu, J. Li, J. Zuo, S. Li, "Path planning of uavs based on collision probability and kalman filter," *IEEE Access*, vol. 6, pp. 34 237–34 245, 2018. DOI: 10.1109/access.2018.2817648. (visited on 01/05/2022).

[16] C. Luo, S. I. McClean, G. Parr, L. Teacy, R. De Nardi, "Uav position estimation and collision avoidance using the extended kalman filter," *IEEE Transactions on Vehicular Technology*, vol. 62, pp. 2749–2762, Jul. 2013. DOI: 10.1109/tvt.2013.2243480. (visited on 01/05/2022).

[17] G. Mao, S. Drake, B. D. O. Anderson, "Design of an extended kalman filter for uav localization," *2007 Information, Decision and Control*, Feb. 2007. DOI: 10.1109/idc.2007.374554. (visited on 01/05/2022).

[18] S. T. Goh, O. Abdelkhalik, S. A. (Zekavat, "A weighted measurement fusion kalman filter implementation for uav navigation," *Aerospace Science and Technology*, vol. 28, pp. 315–323, Jul. 2013. DOI: 10.1016/j.ast.2012.11.012. (visited on 01/06/2022).

[19] G. Welch, G. Bishop, *An introduction to the kalman filter*, Sep. 1997. (visited on 01/18/2022).

[20] D. Simon, "Kalman filtering," *Embedded systems programming*, vol. 14, no. 6, pp. 72–79, 2001.

[21] *Understanding kalman filters*, www.mathworks.com. [Online]. Available: https://www.mathworks.com/videos/series/understanding-kalman-filters.html.

[22] R. J. Meinhold, N. D. Singpurwalla, "Understanding the kalman filter," *The American Statistician*, vol. 37, pp. 123–127, May 1983. DOI: 10.1080/00031305.1983.10482723. (visited on 05/09/2020).

[23] *Kalman filter*, Mathworks.com, 2019. [Online]. Available: https://www.mathworks.com/discovery/kalman-filter.html.

[24] Y. Kim, H. Bang, "Introduction to kalman filter and its applications," *Introduction and Implementations of the Kalman Filter*, Nov. 2018. DOI: 10.5772/intechopen.80600. [Online]. Available: https://www.intechopen.com/books/introduction-and-implementations-of-the-kalman-filter/introduction-to-kalman-filter-and-its-applications.

[25] M. I. Ribeiro, *Kalman and extended kalman filters: Concept, derivation and properties*, Feb. 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.214.809&rep=rep1&type=pdf.

[26] D. J. Simon, "Using nonlinear kalman filtering to estimate signals," *Embedded Systems Design*, vol. 19, no. 7, p. 38, 2006.

[27] H. Nasrabadi, A. Morales, D. Zhu, "Well placement optimization: A survey with special focus on application for gas/gas-condensate reservoirs," *Journal of Natural Gas Science and Engineering*, vol. 5, pp. 6–16, Mar. 2012. DOI: 10.1016/j.jngse.2011.10.002. (visited on 01/15/2022).

[28] S. P. Bradley, A. C. Hax, T. L. Magnanti, *Applied mathematical programming*. Addison-Wesley, 1977.

[29] *Nonlinear programming*, www.mathworks.com. [Online]. Available: https://www.mathworks.com/discovery/nonlinear-programming.html.

[30] *Find minimum of constrained nonlinear multivariable function - matlab fmincon*, www.mathworks.com. [Online]. Available: https://www.mathworks.com/help/optim/ug/fmincon.html.

[31] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[32] T. M. Strat, "Recovering the camera parameters from a transformation matrix," *Readings in Computer Vision*, pp. 93–100, 1987. DOI: 10.1016/b978-0-08-051581-6.50017-9. (visited on 01/15/2022).

[33] *Mathworks - makers of matlab and simulink*, Mathworks.com, 2019. [Online]. Available: https://www.mathworks.com/.

[34] G. H. Burgin, A. J. Owens, A. And, I. Decision Science, *An adaptive maneuvering logic computer program for the simulation of one-on-one air-to-air combat Vol. 2, Program description.* Ntis, 1975.

[35] J. S. McGrew, J. P. How, B. Williams, N. Roy, "Air-combat strategy using approximate dynamic programming," *Journal of Guidance, Control, and Dynamics*, vol. 33, pp. 1641–1654, Sep. 2010. DOI: 10.2514/1.46815. (visited on 12/29/2021).

[36] *Expressing points in different coordinate systems*, www.dgp.toronto.edu. [Online]. Available: http://www.dgp.toronto.edu/~neff/teaching/418/transformations/transformation.html (visited on 11/26/2021).

[37] *Introduction to kalman filter and its applications*, www.mathworks.com. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/68262-introduction-to-kalman-filter-and-its-applications (visited on 12/29/2021).

PUBLICATIONS FROM THE THESIS

Conference Papers

1. E.N.Gökal, U. Sakarya, “Multi-Sensor Data Fusion for Path Prediction of Escaping from Engagement in Unmanned Aerial Vehicle Scenario”, European Journal of Science and Technology, Special Issue 32 (International Conference on Design, Research and Development (RDCONF) 2021 – 15-18 December 2021), pp. 705-710, December 2021. DOI: 10.31590/ejosat.1039358