OPTIMIZED FLIGHT PATH FOR LOCALIZATION USING LINE OF BEARING

THESIS

Namkyu Kim, Captain, Republic of Korea

AFIT-ENY-MS-15-M-246

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense, the United States Government, the Republic of Korea Air Force, the Republic of Korea Ministry of Defence or the Republic of Korea Government. This is an academic work and should not be used to imply or infer actual mission capability or limitations.

AFIT-ENY-MS-15-M-246

OPTIMIZED FLIGHT PATH FOR LOCALIZATION USING LINE OF BEARING

THESIS

Presented to the Faculty

Department of Aeronautics & Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Aeronautical Engineering

Namkyu Kim, BS

Captain, Republic of Korea

March 2015

AFIT-ENY-MS-15-M-246

OPTIMIZED FLIGHT PATH
FOR LOCALIZATION USING LINE OF BEARING

THESIS

Namkyu Kim, BS
Captain, Republic of Korea

Committee Membership:

Richard G. Cobb, PhD
Chair

David R. Jacques, PhD
Member

Matthew D. Sambora, PhD
Member

Abstract

This research develops optimized flight paths for localization of a target using Line Of Bearing (LOB) measurements. The target area is expressed as an error ellipse using the measurement errors of the LOBs. The optimization approach is focused on minimizing the size of the error ellipse. The algorithm for the optimized path is generated and compared with typical flight paths. The optimization routine is based on the results derived from similar research in the literature.

A geometrical method to estimate the error ellipse is combined with optimal control in this research. Each LOB gives a possible target area and this target area can be reduced by overlapping areas developed from multiple LOBs. This geometrical method is easy to understand because its target area can be visualized intuitively. The algorithm based on this method is tested with a single target and with multiple targets in simulation. In addition to analytical simulations of the proposed method, a real-world test is conducted using a remotely controlled truck. From the simulation and a real-world test, the change of the semi-major axis of the error ellipse with increasing number of measurements and the total number of measurements needed to achieved a predefined semi-major axis are verified.

A comparison between the simulation results and an experimental test shows what the similarities and differences are. In addition, a no-fly zone is included into the optimization for the safety of the UAV in real world. Its application and how it improves the performance are described.

Acknowledgements

First of all, I would like to thank my thesis adviser, Dr.Cobb, for his patience and assistance with my thesis. I would also like to thank Dr.Jacques and Col Sambora for their help along the way. Their courses and instruction were instrumental for completion my research. Finally, I want to thank LtCol Agte and Major Dillsaver for working through my optimal control problem.

I owe many thanks to Ms.Robb for helping my family adjust to life in US. My awesome sponsors, Capt Carl Corvin and Capt Charito Corvin provided great support for us. In addition, I would like to thank Victor and Marsha Grazier for helping us.

Finally, I owe my greatest thanks to my wife and daughter for their love, patience and sacrifice. They supported much energy for me to study in US. They have given a lot to follow me to the US for my studies.

Namkyu Kim

Table of Contents

Page

List of Figures

Figure Page

Figure Page

List of Tables

List of Symbols

P_A	$\equiv$	Agent Position, $[\mathrm{x}_a(t), y_a(t)]^T$
P_T	$\equiv$	Target Position, $[\mathrm{x}_t(t), y_t(t)]^T$
$\hat{P}_T$	$\equiv$	The estimated target position
$\tilde{P}_T$	$\equiv$	Difference between the target and the estimated target position
β	$\equiv$	Line of bearing
ρ	$\equiv$	Distance between the agent and the target
ρ_d	$\equiv$	The desired radius of error ellipse
φ	$\equiv$	A unit vector on the line from the agent to the target
$\bar{\varphi}$	$\equiv$	A unit vector perpendicular to φ
ψ	$\equiv$	Heading angle of the agent
ε	$\equiv$	Maximum error of LOB
V	$\equiv$	Velocity of Agent
$\triangle t$	$\equiv$	Measurement time interval
λ	$\equiv$	Eigenvalue of error covariance matrix
v	$\equiv$	Eigenvector of error covariance matrix
θ	$\equiv$	Ellipse rotation angle
g	$\equiv$	Inequality constraint
h	$\equiv$	Equality constraint
λ_g	$\equiv$	Multiplier of inequality constraint
λ_h	$\equiv$	Multiplier of equality constraint
ω	$\equiv$	Angular velocity
τ	$\equiv$	two times of ε
δ	$\equiv$	Avoidance angle

ϕ	$\equiv$	Angle between the x-axis and the vector from the agent to a target
R_{TGT}	$\equiv$	Radius of flight prohibited area around target
P_{obs}	$\equiv$	Obstacle Position
R_{obs}	$\equiv$	Radius of flight prohibited area around an obstacle
ρ_{obs}	$\equiv$	Distance between the agent and an obstacle

List of Abbreviations

UAV	Unmanned Aerial Vehicle
UAS	Unmanned Aircraft Systems
DoD	Department of Defense
AI	Artificial Intelligence
RPA	Remotely Piloted Aircraft
SUAS	Small Unmanned Aircraft Systems
ANT	Advanced Navigation Technology
AFIT	Air Force Institute of Technology
RADAR	RAdio Detection And Ranging
LOB	Line Of Bearing
FIM	Fisher Information Matrix
CRLB	Cramér-Rao Lower Bound
CEP	Circular Error Probability
CPLE	Cartesian Pseudo-Linear Estimator
RRT	Rapidly-exploring Random Tree
KKT	Karush-Kuhn-Tucker

OPTIMIZED FLIGHT PATH

FOR LOCALIZATION USING LINE OF BEARING

I. Introduction

1.1 Background

Throughout the world, Unmanned Aerial Vehicle (UAV) technology is developing very rapidly and has penetrated deeply in various fields. The military has performed much research on UAVs. A few decades ago, UAVs could take only video and still images of a target for surveillance purposes. However, they now can perform military operations by themselves.

> The past decade for Remotely Piloted Aircraft (RPA) mirrors the rapid evolution of combat airpower during World War I: a wave of great ideas, tactics, and technology, brought from air-minded communities flowed in faster than our ability to field them and slower than the land forces would have linked them. But like the Rickenbackers and Lufberys of their day, it was the RPA lieutenants and imperfect as they were, and integrated them into the evolving fight, transitioning the platforms from reconnaissance-only to true multirole Intelligence, Surveillance, and Reconnaissance(ISR) and strike. They delivered disciplined and effective combat airpower every day; another generation of the Air Force's great captains is born.[7]
>
> – RPA Expeditionary Operations Group Commander(2010-2012), Colonel Bill Tart

In other cases, some companies plan to transport products to their customers using UAVs. Clearly, UAV technology is no longer unique. Many countries and companies have established organizations for researching UAV technology and have invested much money for developing UAV technology.

The continued growth of Unmanned Aircraft Systems (UAS) has helped the Department of Defense (DoD) in adopting UAV usage more widely. Especially, cutting-

edge technologies with respect to Artificial Intelligence (AI), communications, propulsion and power enhance its potential capability. In the future, UAVs will be multi-mission and adverse weather capable, so, the DoD is managing its plan for developing RPA and Small Unmanned Aircraft Systems (SUAS). As a part of the DoD organization, the Advanced Navigation Technology (ANT) Center at the Air Force Institute of Technology (AFIT) conducts a wide variety of research for the development of guidance, navigation, and control of UAVs.

Among the various UAV technologies, passive position finding technology is the focus in this thesis. This technology is widely used in military equipment for several reasons. First of all, it is a safe covert way to geolocate in an operational field because it doesn't generate a detectable radio signal. It just receives a radio frequency and determines where it comes from. The operator and/or platform has less possibility to be detected by an enemy's RAdio Detection And Ranging (RADAR). Furthermore, it is more economical and simpler than an active method. For these reasons, this research will discuss passive localization technology, especially concerning what is an efficient and effective UAV flight path for emitter localization.

1.2 Problem Statement

This thesis describes the development of an optimized flight path generation technique for a UAV to find a target's position using passive localization technology. For passive localization, a UAV needs to measure the LOB to an emitter from several places. A single LOB gives the direction to the target emitter. The intersection of several LOBs can be used to localize the target position. When using more LOBs, the target position can be determined more accurately. However, bearing measurements always have some errors due to incorrect equipment operation, equipment precision, equipment calibration, and environmental factors. These errors introduce uncertainty in the LOB measurements. Each LOB has the possibility of being incorrect. Any method of reducing this uncertainty, considering multiple LOBs from multiple dif-

ferent aircraft, can be effective statistically. Developing and efficient method and analyzing the performance for a UAV is the problem considered herein.

Every UAV flight path has a different required flight time for estimating the correct target position within a desired accuracy time. The flight path is a very crucial factor for measuring the LOB. Some collections of LOBs take a short time for calculating the target position with acceptable accuracy. Adversely, some collections of LOBs take a long time to calculate the target position with the same accuracy. In this research, finding out what is an efficient flight path for localization using LOB measurements is a final objective. For this problem, the UAV is referred to as the 'agent' and the emitter is called the 'target'. The 'optimal path' as defined in this research is a UAV flight path that requires the minimum amount of time to localize the target to a given accuracy requirement.

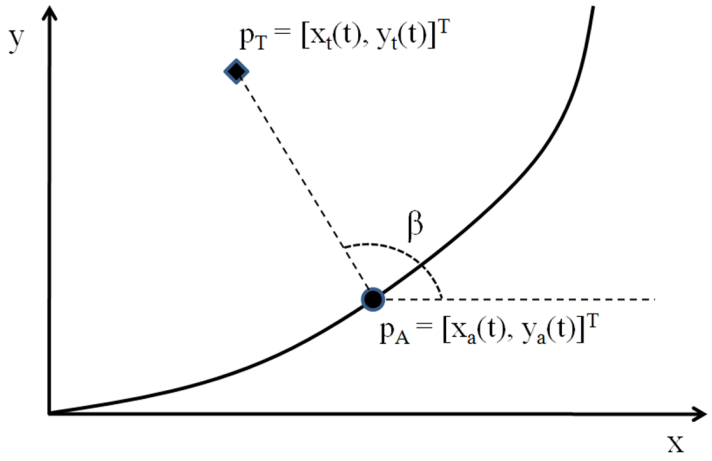


Figure 1. Frame of Problem for Localization

The UAV or Agent position is considered to be a known position $P_A = [x_a(t), y_a(t)]^T \in \mathbf{R}^2$ at time t, and target position is considered to be an unknown position $P_T = [x_t(t), y_t(t)]^T \in \mathrm{R}^2$ at time t. The LOB, β is defined as the angle between the x axis and a line from P_A and P_T as in Figure 1.

For finding the optimized path for determining the target position for a simulated case, P_T , the initial agent position is considered to be the some point in 2D space.

Additionally, the target position is considered to be the other point. The speed of the agent is assumed constant and the angular velocity is constrained, the optimized path for finding the target position will be discussed later.

Initially, the optimized flight path for a single target is analyzed in order to find out the tendencies of the general flight path. After that, the multiple target problem is analyzed comparing with the single target problem. By doing this, the effect of adding a target to the optimized path is explored and discussed.

1.3 Research Objective and Scope

The development of an algorithm forming the optimized flight path for a single target is the main objective of this research. The optimized flight path model can be derived for certain situations using optimal control theory. Additionally, for multiple targets, a method to determine the optimal flight path will be determined.

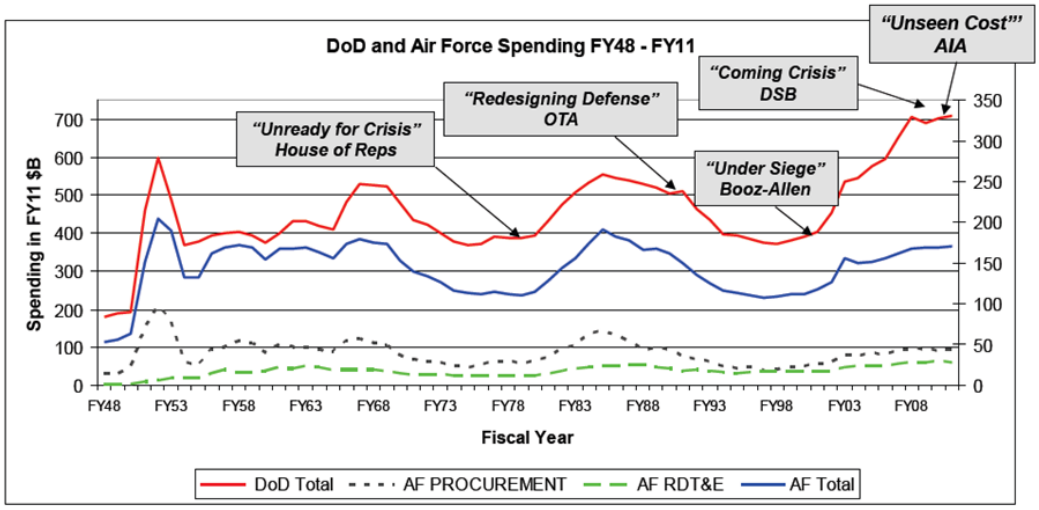
Proof of the optimized path is the next objective in this thesis. A MATLAB simulation model will be developed to provide this proof. With this model, the optimized flight path can be analyzed to determine its performance for single target and multiple target cases.

Finally, a real-world test using a remote controlled truck is the final step of this research. Analysis of the truck's movement and comparison with simulation will follow for determining the optimized path modeling and its application to the UAV problem.

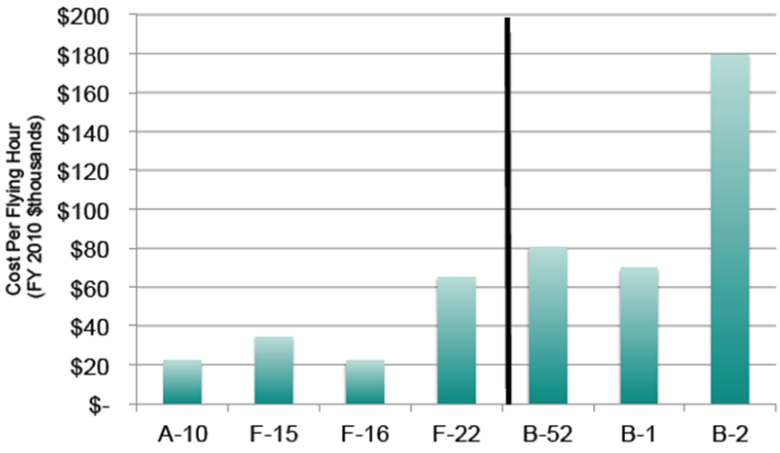
1.4 Significance of Research

Intelligence/information has been the focal point in recent warfare. Therefore, the speed of intelligence/information updating is significant because of the variability of war theater factors. Agility of surveillance is required for the acquisition of information. In this respect, optimized path planning may be the best way for acquiring agility using UAV surveillance.

The UAV is operated inside of enemy territory for achieving target information. The enemy's weapons always threaten the survivability of the UAV. There are many ways to maximize survivability such as maintaining a high altitude, often used with high resolution cameras. Alternatively, optimized flight path planning is a good method to increase survivability by reducing the time of exposure to the enemy's threats.



(a) DOD & Air Force Spending



(b) Cost per Flying Hour

Figure 2. Cost Spending [2]

Recently, managing finances has been a big concern to the Air Force because the cost of operation and maintenance of aircraft has soared tremendously as shown in

Figure 2a. This trend is expected to continue. Especially, newer aircraft require more funds than older ones as shown in Figure 2b. This trend will also continue in future. Therefore, An unmanned and autonomous system is required for saving the cost of Air Force. It will need less people and less cost to be operated.

1.5 Methodology

Every objective has a different methodology for achieving it. First of all, optimal control theory is used for finding the optimized path with constraints like velocity and heading angle rate. There are discrete methods and continuous methods in optimal control theory. Among these methods, the discrete method is used for this research. The cost function will be the time to acquire a desired accuracy and constraints on speed and angular velocity of heading angle will be considered.

MATLAB/Simulink is used for solving the optimal control problem. Among the many functions for solving the optimal problem, the 'fmincon' function will be used for doing this. This function is used for finding the flight path which minimizes the cost function using optimal control theory. It is easy to express the cost function and constraints. By using iteration, the optimal solution can be found which satisfies the constraints.

In a real-world test, a remote controlled(RC) truck and autopilot equipment are used for the test. Additionally, direction finding equipment, including a modem and radio are used for measuring the LOB. A portable radio functions as a target emitting radio signal. By using a RC truck, the real test can be done easier on the ground than in a flight test. From this test, the shape of the optimal path from real-world data can be compared to the result of the MATLAB/Simulink simulation predictions for future flight tests can be made.

1.6 Thesis Overview

This chapter provided a brief background about passive localization technology and the reason why this topic is worth researching. In addition, the specific problem concerning passive localization technology and methodology for targeting is described briefly.

In Chapter II, the theoretical background knowledge will be described from the contents of previous work. Additionally, these previous results are used partially to design the optimal control algorithm and their results can be compared with this research's results.

Chapter III describes an algorithm to find an optimal flight path for targeting. In addition, the specific process of the algorithm is explained using a sample result using MATLAB/Simulink. Sensitivity of the result and specific results of the scenario follows it.

In Chapter IV, the preparation and execution of a real-world test using the algorithm described in this research is presented. The result of the test is compared to the expected results from Chapter III. So, comparing these results gives us the difference between theory and real world. Chapter V includes the conclusions and recommendations.

II. Literature Review

2.1 Overview

This chapter provides two categories of research. The first part is fundamental background research, and the second is related research. The background research provides ideas on how to achieve the goals identified in Chapter I. They suggest the framework of the solution using LOB to localize the target position. What are the control rules of the agent and the constraints on this problem are described in the background research. However, their goals and methodologies are different from the research herein. One of them tries to maintain a certain distance from the target position in the final state with its own control rules. The other goal is to make the agent arrive at the target position minimizing uncertainty in the x-z plane. Their approaches for solving the optimal problem provide an alternative way of solving these different problems. Even though the cost function, constraints and boundary conditions are different, the fundamentals of the problem and what is minimized for an optimal solution are similar.

The related research section deals with similar issues involving geolocation. They suggest different approaches to solving these issues. The first one describes a geolocation problem in 3D space. Even though a 3D problem can be different from a 2D problem, its algorithm to define the estimated target region is very valuable in proceeding with similar research. In addition, it shows what happens to the estimated target region while flying a straight flight path. Another research paper suggests several ways of numerical calculations for passive geolocation. It explains several technical methods for calculating target position and how to analyze the error of the LOBs. These methods provide the technical background for proceeding with this research. The last research proposes a path planning method, rapidly-exploring random trees. An RRT is iteratively formed using random points by applying control inputs and this can be assumed as the moving object's path.

2.2 Background Research

There are articles about target localization and circumnavigation using bearing measurement and stochastic real-time optimal control using a pseudospectral approach which provides the background research for this thesis. First, a paper from the 49th IEEE conference covers the fundamentals of the problem with respect to the mathematical approach in the cartesian coordinate system. In addition, the thesis on stochastic real-time optimal control described by Ross gives ideas about using error ellipses to represent uncertainty of the target position.

The mathematical foundation in the cartesian coordinate system describes how one similar error ellipse representing the uncertainty of the target position for one target differs from a second target ellipse as will be dealt with in this thesis.

2.2.1 Target Localization using LOB [5].

The first paper reviewed, which is from the 49th IEEE conference, proposes an estimator using the bearing angle to the target to solve the localization and navigation problem for a stationary target. As previously described for the current research, the final state of the agent in this problem is to circumnavigate the target with a specified stand off distance. In the cartesian coordinate system, P_A and P_T are the agent position and target position. The distance, ρ_d is the desired radius of the circle around the target in the final state and $\rho(t)$ is the distance between the agent and the target. In addition, $\hat{P}_T$ is the estimated target position, $\hat{\rho}(t)$ is the distance between the agent and the estimated target position, $\varphi(t)$ is a unit vector on the line from the agent to the target and $\bar{\varphi}(t)$ is the unit vector perpendicular to $\varphi(t)$ as shown in Figure 3. With these definitions, estimation error is defined as

$$\tilde{P}_T(t) = \hat{P}_T(t) - P_T(t) \tag{2.1}$$

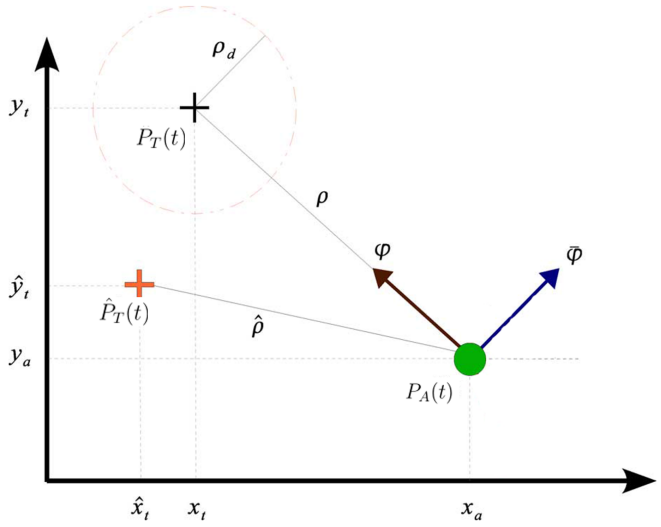


Figure 3. Agent, Target and Estimated Position of the Target (adopted from [5])

The first goal is to find an estimator that estimates the unknown target position, P_T, using LOBs. The estimator is defined as

$$\dot{\hat{P}}_T(t) = k_{est}(I - \varphi(t)\varphi^T(t))(P_A(t) - \hat{P}_T(t)) \tag{2.2}$$

where I is the identity matrix and k_{est} is a constant scalar.

The derivative of the agent position is defined using φ(t) and $\bar{\varphi}$(t) as

$$\dot{P}_A = (\hat{\rho}(t) - \rho_d)\varphi(t) + \alpha\,\bar{\varphi}(t) \tag{2.3}$$

where α is a scalar multiplier for making the agent move the right way. As this equation expresses, φ(t), $\bar{\varphi}$(t) and α effect the direction of P_A. Later, this makes the distance between the agent and target, ρ(t), converge to the value of ρ_d. So, Equation 2.3 functions as the control law of the agent.

Using this control law, $\bar{\varphi}$(t) is varied which makes $\tilde{P}_T$ exponentially approach zero. This guarantees the norm of the estimation error is less than ρ_d. Additionally, it prevents the agent from colliding with the target.

The simulation result shows that the agent approaches with the target following a spiral flight path until it meets a certain radius as seen in Figure 4. Furthermore, the norm of $\tilde{P}_T$ goes to a very small value.

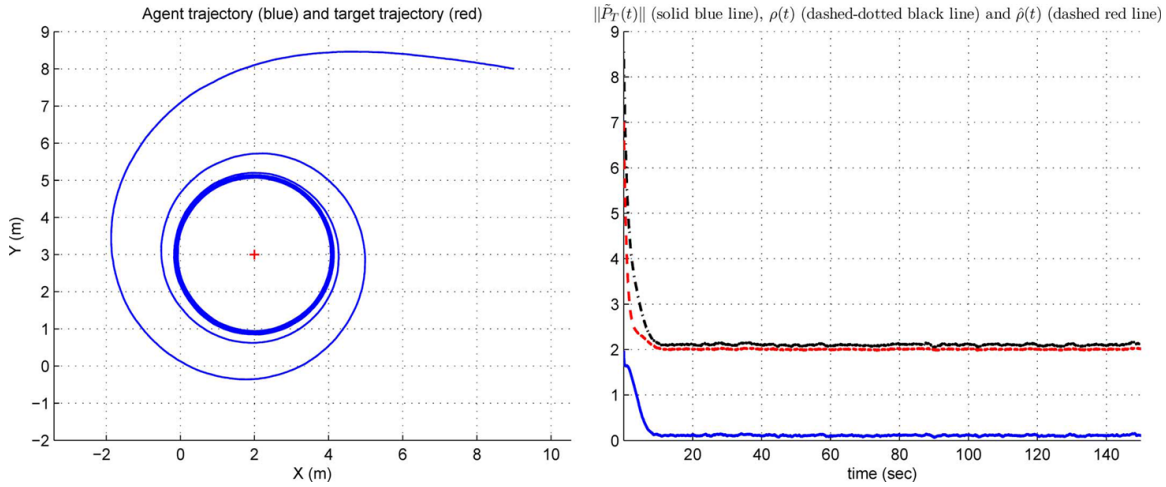


Figure 4. Simulation Result [5]

2.2.2 Stochastic Real-Time Optimal Control for Bearing-Only Trajectory Planning [11].

In his dissertation, Ross proposes a method to deal with the optimal control problem and the final estimation requirements at the same time. The goal of his research was to provide an optimal flight path for landing at a certain place in the x-z plane.

For solving this optimal problem, the Fisher Information Matrix (FIM) was used. When the agent moves for a certain amount of time, the amount of information is gathered by the agent's sensor. The FIM measures the amount of information along each direction, and this is expressed by the probability density of the measurements. So, observability can be assessed from the geometry of the problem. It is defined as,

$$Z_k = E\left[\left(\frac{\partial}{\partial x_k} ln\, p_{z_k|x_k}\right)^2 \mid x_k\right] \tag{2.4}$$

which is a byproduct of the Cramer-Rao Lower Bound (CRLB). For minimizing the uncertainty ellipsoid of the target position, the smallest FIM eigenvalue needs to be maximized. Eigenvalues of the FIM are the radius size of the information ellipse. So, maximizing the eigenvalues means maximizing the information ellipse, and it will minimize uncertainty.

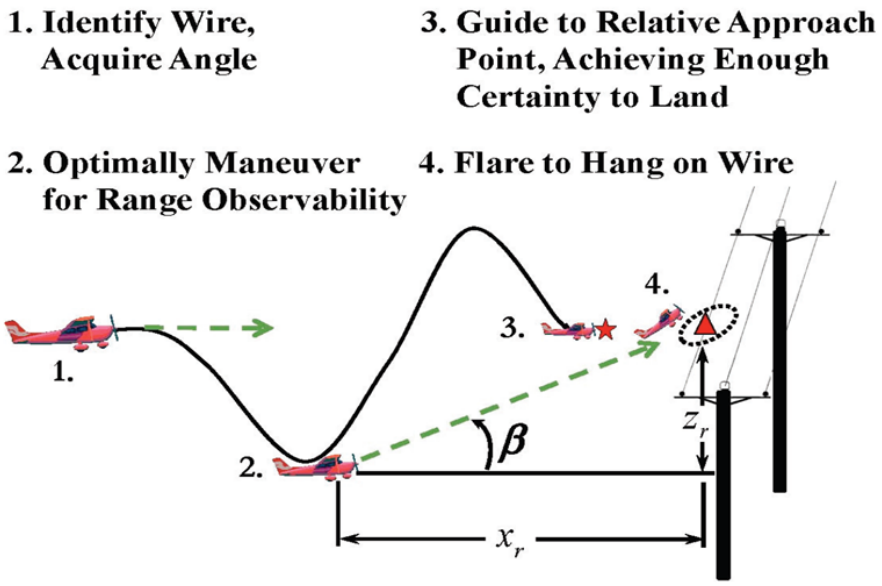


Figure 5. Conceptual Approach [11]

The control mechanism for landing on the wire was divided into 4 steps. The first two steps are the acquisition & maneuver segments where control is provided for moving to the approximated position. The next step is the approach segment, where control is provided to an offset approach point for maximizing observability. Finally, the flare segment tries to land the agent on a predetermined target position safely. These steps are shown in Figure 5. This problem is solved in the x-z plane as shown in Figure 5. Note that pitch angle is the control for this problem.

In this problem, the position of the agent is defined as

$$x = [\, x \quad z \,]^T \tag{2.5}$$

which denotes the position in the x-z plane. The relative position of the agent to the target is defined as

$$x_r = \begin{bmatrix} x_t - x \\ z_t - z \end{bmatrix} \tag{2.6}$$

The real measurement value of the angle from the horizontal inertial level to the target is,

$$h(x) = \beta = tan^{-1}(z_r/x_r) \tag{2.7}$$

and the Jacobian of it, H, contains the information for determining each new measurements. The Jacobian of the real measurement value is given as,

$$H_k = \bigtriangledown_{x_k} h(x_k) = \begin{bmatrix} \frac{z_{r_k}}{\rho_k^2} & -\frac{x_{r_k}}{\rho_k^2} \end{bmatrix} \tag{2.8}$$

where ρ represents the range from the agent to the target, $\rho_k = \sqrt{x_{r_k}^2 + y_{r_k}^2}$.

By gathering information on the movement of the agent, the entire FIM becomes,

$$\begin{aligned} Z_k &= Z_0 + \frac{1}{\rho_\beta^2} \begin{bmatrix} \sum_{i=1}^{n} \frac{sin^2\beta_i}{\rho_i^2} & -\sum_{i=1}^{n} \frac{sin\beta_i cos\beta_i}{\rho_i^2} \\ -\sum_{i=1}^{n} \frac{sin\beta_i cos\beta_i}{\rho_i^2} & \sum_{i=1}^{n} \frac{cos^2\beta_i}{\rho_i^2} \end{bmatrix} \\ &\equiv Z_0 + \begin{bmatrix} \int_{t_0}^{t_k} \dot{\zeta_1}(t)dt & \int_{t_0}^{t_k} \dot{\zeta_3}(t)dt \\ \int_{t_0}^{t_k} \dot{\zeta_3}(t)dt & \int_{t_0}^{t_k} \dot{\zeta_2}(t)dt \end{bmatrix} \end{aligned} \tag{2.9}$$

where ζ_i(t) is the information state.

The optimal control problem can be then solved with the state vector :

$$\tilde{x} = [x\ z\ \ v_x\ \ v_z\ \ \zeta_1\ \ \zeta_2\ \ \zeta_3]^T \tag{2.10}$$

with this constraint vector,

$$\begin{bmatrix} -9m \\ 0.8m \\ -0.5m/s \\ -0.5m/s^2 \\ -30^\circ \end{bmatrix} \leq \begin{bmatrix} x \\ z \\ v_x, v_z \\ u_x, u_z \\ \beta \end{bmatrix} \leq \begin{bmatrix} x_{offset} + \tilde{x} \\ 5.5m \\ 0.5m/s \\ 0.5m/s^2 \\ 40^\circ \end{bmatrix} \tag{2.11}$$

where the constraints were specific for the application Ross was using. The simulation result from this optimal solution gives the optimal path for safe landing at the predetermined position. Figure 6 shows the experimental flight test result. The ellipse on each graph of Figure 6 denotes the estimated predetermined location for landing.

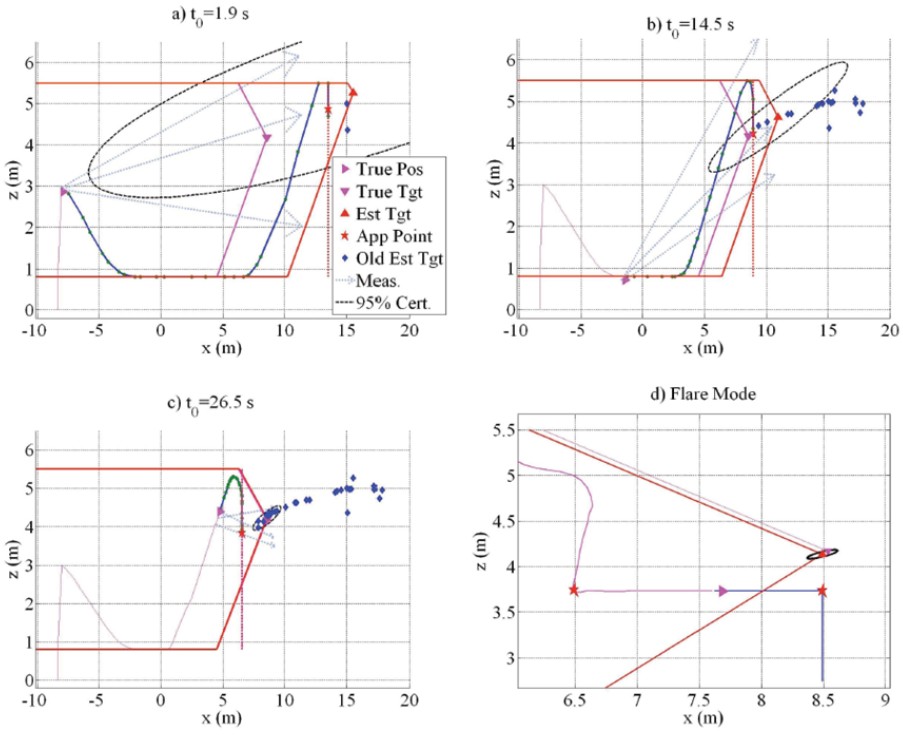


Figure 6. Result from real-world flight test [11]

The indicated ellipse size is closely related to the uncertainty of the final location. The size of the ellipse is proportional to the uncertainty. As shown in Figure 6,

the ellipse becomes smaller with increments of time because the LOB from several different positions makes it smaller. The uncertainty of the final target location is shrinking with an increasing number of LOBs.

Minimizing the size of the uncertainty ellipse is the goal of the research herein using optimal control theory. So, the next chapter will develop minimizing the uncertainty ellipse for a target localization with the LOB, using some of the concepts just reviewed.

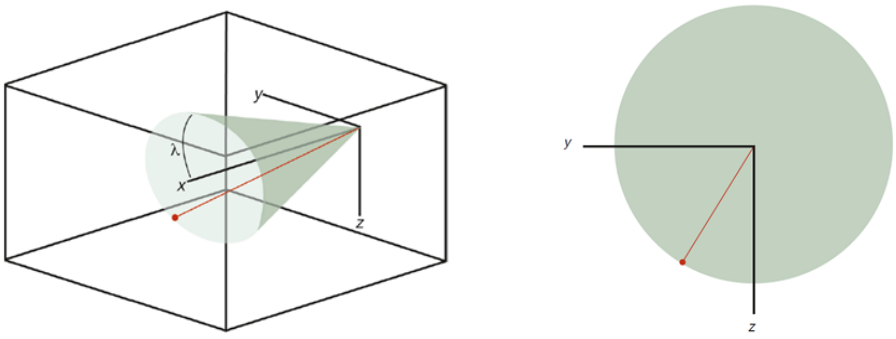
2.3 Related Research

In this section, research about a mathematical model for geolocation is described. The first one is geolocation using direction finding angles. It describes a real method of geolocation from a direction finding angle in 3D space. How the estimation ellipse is generated in 3D space is described mathematically. In addition, the result of a simulation provide a good understanding of the nature of geolocation with LOBs. The second paper reviewed is on numerical calculations for passive geolocation scenarios. The paper introduces several statistical processing methods for bearing data and algorithms. These methods could suggest several new ideas and support for developing them. This mathematical research presented is very helpful for approaching the goal of this thesis. The last research paper reviewed suggests an algorithm for path planning. Tge method uses a random state for a new state and a set of these new states is assumed as a vertex. This process is more efficient in time than existing techniques, so it is widely used in real-world applications.

2.3.1 Geolocation Using Direction Finding Angles [6].

Geolocation using direction finding angles (Grabbe, 2013) provides geolocation algorithms for the estimated target location and estimation error covariance matrix. The error covariance matrix is a statistical uncertainty in location estimation. Using error covariance, the error ellipse illustrates the possible region for the target position.

In their paper, LOB was measured in 3D space and expressed as a scalar angle λ. Considering the error of measurement, it can be thought of in 3D space and 2D



(a) 3D view (b) 2D view

Figure 7. LOB measurement [6]

space as shown in Figure 7. With respect to position, ψ is defined as longitude and θ as latitude. So, the target position is described as

$$x = \begin{bmatrix} \psi \\ \theta \end{bmatrix} \tag{2.12}$$

and measurement function h_i and z_i are defined as

$$h_i(x) = f_i(p_{tgt}(x)) \tag{2.13}$$

$$z_i(x) = h_i(x) + v_i \tag{2.14}$$

where v_i represents the measurement error. If these errors are unbiased, uncorrelated and normally distributed, they can be expressed as

$$v_i \sim N(0, \sigma_i^2) \tag{2.15}$$

where σ_i(the standard deviation) is assumed known for each measurement i.

Next, these stacks of LOBs are described in vector form as

$$z = h(x) + v \tag{2.16}$$

$$v \sim N(0, R) \tag{2.17}$$

where the covariance of all the measurements are contained in R as

$$R = \begin{bmatrix} \sigma_1^2 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sigma_n^2 \end{bmatrix} \tag{2.18}$$

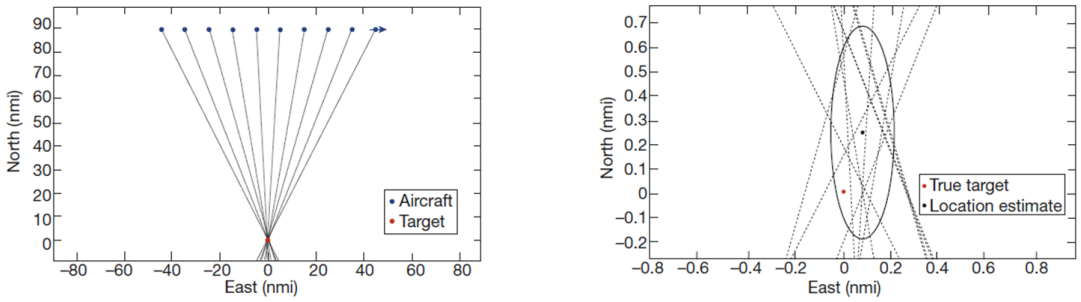
Using this data, the recursive estimated position of the target is

$$\hat{x}_{k+1} = \hat{x}_k + [H_k^T R^{-1} H_k]^{-1} H_k^T R^{-1} (z - h_k) \tag{2.19}$$

where $H_k = \frac{\alpha h}{\alpha x}(\hat{x}_k)$ and the covariance matrix for the target position error is calculated by

$$P_x = E[(x - \hat{x})(x - \hat{x})^T] = [H^T R^{-1} H]^{-1} \tag{2.20}$$

where $E[\ \cdots\ ]$ is the statistical expectation operator.



(a) Flight Path & TGT Position (b) Estimated TGT Position & Error Ellipse

Figure 8. Straight Flight Path Result [6]

This statistical calculation gives the error ellipse surrounding the estimated target position. Figure 8 shows the result of this method from a straight flight path of the aircraft. In Figure 8a, points on same direction describes the movement of aircraft and the point located in bottom of this figure shows the target position. Figure 8b

expresses the error ellipse and target position. The center of the error ellipse is the estimated target position and other point (0, 0) is the true target position.

The error ellipse has the long radius directed towards the straight flightpath from the real target position because the LOBs give more error information in the horizontal direction to the flightpath. Considering the positions of the aircraft and target, the perpendicular direction to flightpath has more ambiguity. So, this result is very reasonable.

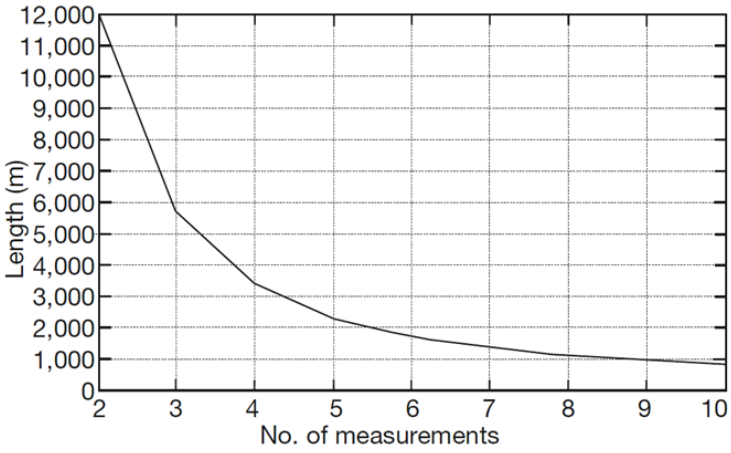


Figure 9. 95% Elliptical Error Probability [6]

As shown Figure 9, more LOBs can reduce the length of the semi-major axis of the error ellipse. Each LOB confines the possible target position region. This error ellipse can be changed by varying the aircraft flightpath. The flightpath will affect the semi-major axis of the error ellipse. The relation between the flight path and the error ellipse is discussed in Chapter III, and will provide a method to choose a 'best' flightpath to achieve the desired error ellipse size.

2.3.2 Numerical Calculations for Passive Geolocation [8].

In Koks, he suggests a method of passive geolocation by the analysis of LOBs considering measurement noise. The LOBs are measured from various points following the flightpath to a stationary radio emitter positioned at a certain point. These

LOBs from different points make the Circular Error Probability (CEP) smaller. The probable emitter point is described by the CEP. This circle has a 50% probability of the estimated position of the emitter. A smaller CEP means more accurately estimated position than a larger CEP. For analysis, the CRLB was used in Koks research.

The CRLB shows how well a parameter can be assumed as described in Ross's work. If we want to extract a parameter x, the unknown data contained in the signal has an effect on the parameter x. To determine x, an estimator $\hat{x}$ is calculated when a new LOB is measured. The Cramér-Rao theory is used for calculating this estimator. The variance of the estimator and the inverse of the Fisher Information Maxtix, J, is used as the CRLB.

$$var(\hat{x}) > J^{-1} \equiv CRLB \tag{2.21}$$

The specific calculation steps taken in Koks are similar to Ross's paper.

Another method for analyzing error is the least squares method. Each measurement, z, includes noise. Several measurements can be expressed as,

$$\begin{bmatrix} z_1 \\ \vdots \\ z_n \end{bmatrix} = H \begin{bmatrix} x_1 \\ \vdots \\ x_m \end{bmatrix} \tag{2.22}$$

where H is n×m matrix. The best estimated position of the emitter can be chosen by minimizing the distance between Hx and z.

$$\bigtriangledown \{|Hx - z|^2\} = \bigtriangledown[(Hx - z)^T(Hx - z)] \tag{2.23}$$

So, the least squares solution, $\tilde{x}$, to Equation 2.22 is

$$\tilde{x} = (H^T H)^{-1} H^T z \tag{2.24}$$

In this case, the difference between $H\tilde{x}$ and z is assumed to be the error for each measurement. The CEP can be extracted from these errors by using several measurements in a batch process.

Another method for analysis is the Cartesian Pseudo-Linear Estimator (CPLE). This method uses the orthogonality of vectors indicating the LOB. In 2D space, x is defined as

$$x = \begin{bmatrix} s_{0,x} \\ s_{0,y} \\ v_{0,x} \\ v_{0,y} \\ a_x \\ a_y \end{bmatrix} \tag{2.25}$$

where s_0 is the initial estimated emitter position, v_0 is the initial velocity and a is the acceleration. The next estimated emitter position is calculated as

$$s(t) = A(t)x \tag{2.26}$$

where

$$A(t) = \begin{bmatrix} 1 & 0 & t & 0 & \frac{t^2}{2} & 0 \\ 0 & 1 & 0 & t & 0 & \frac{t^2}{2} \end{bmatrix} \tag{2.27}$$

Similarly to the least squares method, CPLE minimizes $|Hx\text{-}z|^2$ to arrive at the estimated emitter position. $|Hx\text{-}z|$ is defined as $\sum_k (b_k^{\perp} v_k)$ in the CPLE method. b_k denotes the vector of the LOB and v_k describes error of LOB as shown in Figure 10. So, for analysis with error, $b_k^{\perp} v_k$ gives the CEP shape on the graph.

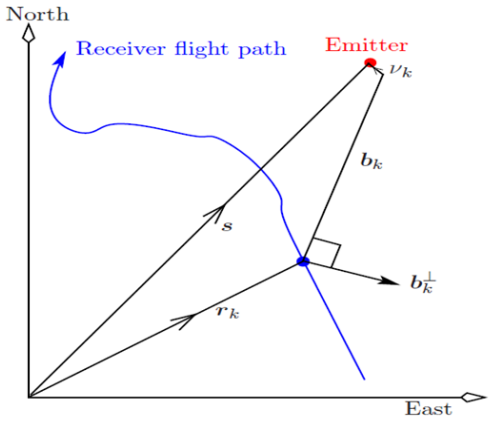


Figure 10. Cartesian Pseudo-Linear Estimator Approach [8]

2.3.3 Rapidly-Exploring Random Trees: A new Tool for Path Planning [9].

The Rapidly-exploring Random Tree (RRT) provides a broad class of path planning as a randomized data structure. A RRT can expand iteratively by applying control laws with randomly, selected points. So, this iterative method is different than the point-to-point convergence method. This point-to-point convergence method is hard to naturally extend to the general nonholonomic planning problem. In addition, the connection problem in path planning is as difficult as designing a nonlinear controller. With respect to path expansion, the RRT method suggests a very efficient path planning solution.

Path planning generates a continuous path from initial state, x_{init}, to a goal state, x_{goal}, in the configuration space, C. A state transition equation is defined as

$$\dot{x}(t) = f(x, u) \tag{2.28}$$

where vector u is a set of inputs and x_{new} is defined as

$$x_{new} \simeq x + f(x, u) \cdot \triangle t \tag{2.29}$$

using Euler integration. This process is denoted as NEW_STATE(x, $u, \triangle t$) in Table 1. Additionally, an initial state is described as x_{init}, RRT is shown as τ and K is the number of vertices as shown in Table 1.

Table 1. Algorithm of Rapidly-Exploring Random Trees [9]

GENERATE_RRT($x_{init}, K, \triangle t$)	
1	τ.init(x_{init});
2	for $k = 1$ to K do
3	$x_{rand} \leftarrow$ RANDOM_STATE();
4	$x_{near} \leftarrow$ NEAREST_NEIGHBOR(x_{rand}, τ);
5	$u \leftarrow$ SELECT_INPUT(x_{rand}, x_{near})
6	$x_{new} \leftarrow$ NEW_STATE(x_{near}, $u, \triangle t$)
7	τ.add_vertex(x_{new})
8	τ.add_vertex(x_{near},x_{new}, u)
9	Return τ

In this algorithm, x_{rand} is chosen from the configuration space in each iteration and the closest vertex x_{near} from x_{rand} is selected in Step 4. u is calculated recursively for moving x_{near} in Step 5. With this u value, NEW_STATE gives x_{new} value and it is iteratively accumulated in vertex, τ. This algorithm is widely used in path planning and its application example is shown in Figure 11.

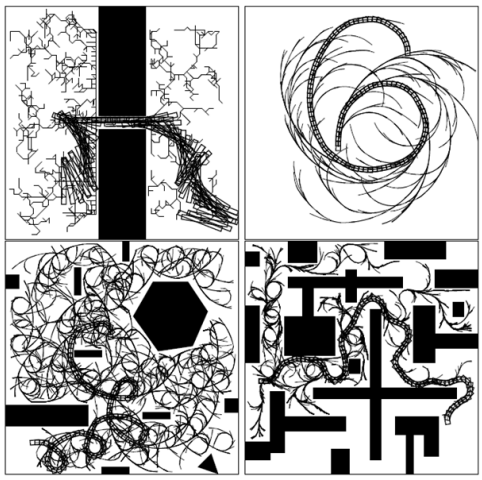


Figure 11. Application Example of Rapidly-Exploring Random Trees [9]

2.4 Summary

The objective of this research is to determine methods applicable to UAVs with direction finding equipment for localization of the target. Background research shows how previous researchers met this objective. They explained what can be a control rule, constraints and a cost function in the optimal problem. The overall framework of solution became fundamental to this research. Related research described several ways of numerical methods about geolocation. This discussion of methods gave motivation to this research.

In addition, results of the background research and related research supported the proposed research direction to develop an algorithm to minimize uncertainty. These results suggested both the shape of the flightpath as an optimal control solution and how the error ellipse is changed by varying the flightpath. This information was a good reference to proceed this research. How this previous research effects the current research is shown in the next chapter.

III. Methodology

3.1 Overview

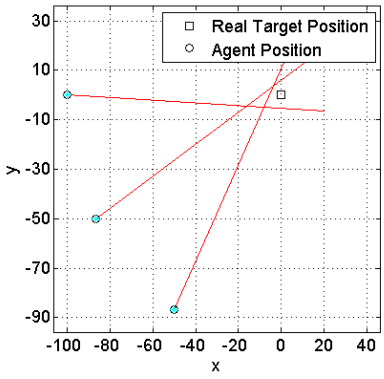
This chapter discusses the proposed method of targeting using LOBs. Each LOB includes measurement error as previously described. This thesis uses a geometric method to eliminate error for targeting. This geometric approach is different from other research work described in Chapter II. In addition, a method for determining the optimal flightpath for targeting is discussed after explaning the method. However, the optimal flightpath process using a geometric method has several singularities. These singularities and solutions are described in this chapter. Furthermore, simple results drived from the presented algorithm are analyzed and a sensitivity on several factors is described. Finally, application of this algorithm and resulting performance is discussed for specific scenarios of single target and multiple targets.

3.2 Geometric Targeting Approach

3.2.1 Introduction.

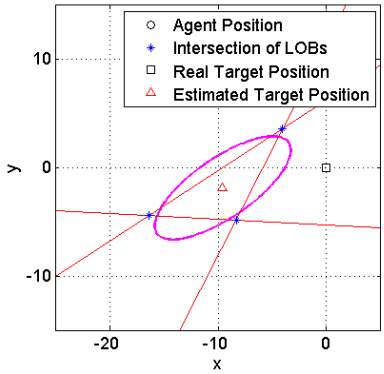
Targeting methods using LOBs discussed in Chapter II normally use a line from the agent position to the direction of measured LOB. Using a single line per single LOB gives an estimated target position directly by the least square method. However, the error ellipse which indicates the possible target region cannot be derived directly by considering error of the measurement. This is explained as follows. First of all, the error matrix must be calculated before an error ellipse can be made. The error matrix requires a set of vectors from the estimated target position to the intersections of the LOBs or the nearest point to line of LOBs. The directly calculated error ellipse from this error matrix are shown in Figure 12. In this case, the errors of LOBs are assumed to be 3°. In some cases, the direct ellipse of error doesn't include the real target position as shown in Figure 12c and 12e. Using only estimates of the real target position but without uncertainty results in a region identified that does not contain

the target. This is a result of using a small number of LOBs and not including the uncertainty with each LOB. So, a method is needed to enlarge these ellipses to fix the accuracy problem using a probability method.



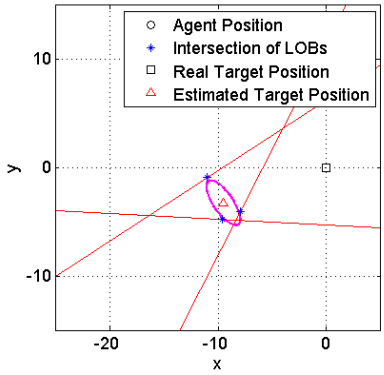
(a) Situation Example

(b) Intersections of LOBs



(c) Intersections of LOBs

(d) Nearest Points



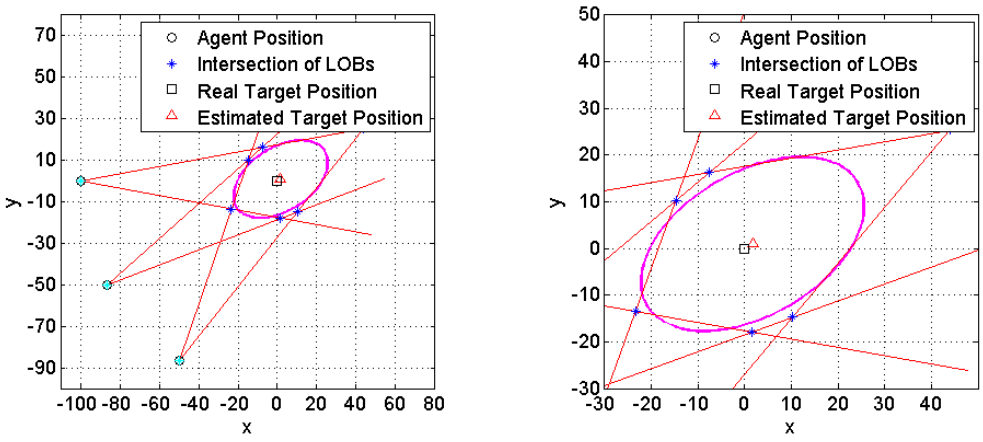
(e) Nearest Points

Figure 12. Error Ellipses of Different Methods

An alternative approach is to use a new targeting method using a geometric approach which uses two lines per single LOB. When using two lines, the angles which add and subtract from the maximum error value, ε, to measured LOB value, β, are used.

$$\tilde{\beta} = \beta \pm \varepsilon \tag{3.1}$$

The region between these two lines indicate the possible target region. Even though β has error, $\tilde{\beta}$ indicates the real target position. So, $\tilde{\beta}$ can be used for targeting. Using a single LOB created only one line as a possible target position but using two lines per single LOB creates an area as a possible target position. Overlapped areas by LOBs indicate a possible target position as shown in Figure 13. This overlapped area can be reduced by increasing the number of LOBs from different agent locations. In this case, the intersections of lines are used for constructing the error matrix and the error ellipse can be derived from this error matrix directly. For these reasons, this method is simpler to apply and more intuitive than a non-geometric approach.



(a) Overall Situation (b) Error Ellipse

Figure 13. Two Line use Method

3.2.2 Error Ellipse Generation.

Every intersection is determined by two lines resulting from the different LOBs using Equation 3.1. The error vector is a vector from the estimated target position

to the intersection. So, for creating the error vector, the estimated target position is needed to be calculated. The estimated target position will become the center of the error ellipse. The estimated target position, $\hat{P}_T$, is calculated by taking the average of intersections. The error matrix is composed of a set of these error vectors. Using this $\hat{P}_T$, the error matrix is defined as

$$\begin{bmatrix} Error \\ Matrix \end{bmatrix} = \begin{bmatrix} \hat{P}_T - P_{inter,1} \\ \vdots \\ \hat{P}_T - P_{inter,n} \end{bmatrix} = \begin{bmatrix} x_{err,1} & y_{err,1} \\ \vdots & \vdots \\ x_{err,n} & y_{err,n} \end{bmatrix} \tag{3.2}$$

where $P_{inter,n}$ is the position of intersection and n is the number of intersections.

Next, the error covariance matrix is needed for analyzing the relation between the intersections of the x and y values. The covariance shows how much the variables change related to the average value in this matrix. Defining the error matrix as in Equation 3.3,

$$A = \begin{bmatrix} x_{err,1} - \frac{x_{err,1}+\cdots+x_{err,n}}{n} & y_{err,1} - \frac{y_{err,1}+\cdots+y_{err,n}}{n} \\ \vdots & \vdots \\ x_{err,n} - \frac{x_{err,1}+\cdots+x_{err,n}}{n} & y_{err,n} - \frac{y_{err,1}+\cdots+y_{err,n}}{n} \end{bmatrix} \tag{3.3}$$

the error covariance matrix can be calculated by

$$Covariance\, error\, of\, matrix = \frac{A^T \cdot A}{n-1} \tag{3.4}$$

In this case, the terms $\frac{x_{err,1}+\cdots+x_{err,n}}{n}$ and $\frac{y_{err,1}+\cdots+y_{err,n}}{n}$ are zero by the definition of error matrix(Equation 3.2). So, the error covariance matrix can be simply calculated as

$$\begin{bmatrix} Error \\ Covariance \\ Matrix \end{bmatrix} = \begin{bmatrix} \frac{(x^2_{err,1}+\cdots+x^2_{err,n})}{(n-1)} & \frac{(x_{err,1}\cdot y_{err,1}+\cdots+x_{err,n}\cdot y_{err,n})}{(n-1)} \\ \frac{(x_{err,1}\cdot y_{err,1}+\cdots+x_{err,n}\cdot y_{err,n})}{(n-1)} & \frac{(y^2_{err,1}+\cdots y^2_{err,n})}{(n-1)} \end{bmatrix} \tag{3.5}$$

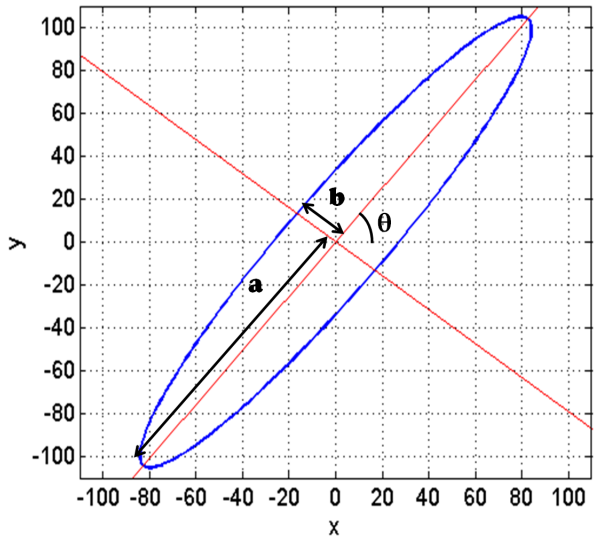


Figure 14. General Ellipse

Next, the error ellipse is extracted from the error covariance matrix. For making an ellipse, semi-major axis, semi-minor axis and rotation angle for expressing it in the cartesian coordinate system like Figure 14 is required. The radii can be calculated using the eigenvalues of the error covariance matrix and the rotation angle can be calculated with the eigenvector of the error matrix. The eigenvector, v, is a non-zero vector and eigenvalue, λ, is a scalar multiplier in

$$A \cdot v = \lambda \cdot v \tag{3.6}$$

where A is a square matrix. In this case, the covariance matrix is a 2 $\times$ 2D matrix and eigenvalues of it gives two values. So, the radii are defined as

$$radius = \sqrt{eigenvalue\,(\,covariance\,(\,Error\ matrix\,)\,)} \tag{3.7}$$

which is computed for each eigenvalue to produce the major and minor axis radii.[13] So, the largest value among them is the semi-major axis, a and small one is the short radius, b. In addition, the eigenvector matrix of the covariance matrix is composed as

$$\begin{bmatrix} Eigenvector \\ Matrix \end{bmatrix} = \begin{bmatrix} cos\,\theta & sin\,\theta \\ sin\,\theta & -cos\,\theta \end{bmatrix} = \begin{bmatrix} V_1 & V_2 \\ V_2 & -V_1 \end{bmatrix} \tag{3.8}$$

$$\begin{aligned} tan\,\theta &= \frac{V_2}{V_1} \\ \theta \;&=\; tan^{-1}\,\frac{V_2}{V_1} \end{aligned} \tag{3.9}$$

where θ is a rotation angle. With two radii and a rotation angle, θ, the error ellipse can be expressed in the cartesian coordinate system as shown in Figure 14.

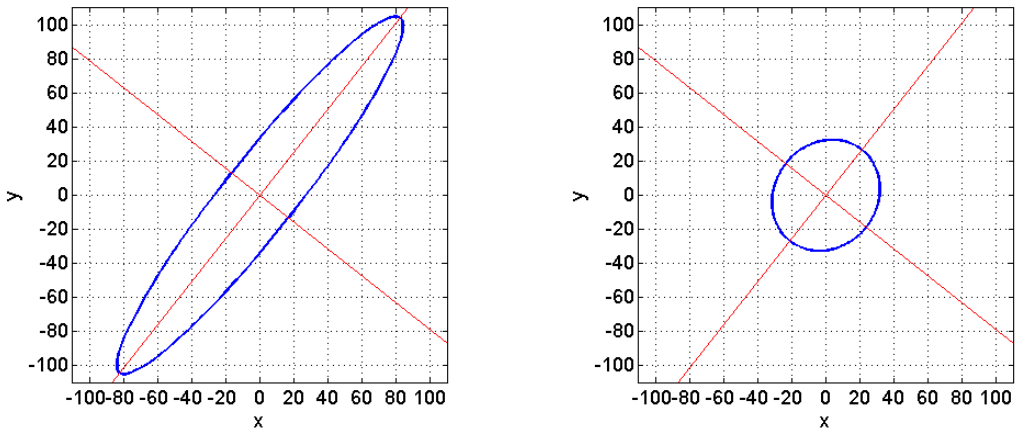
3.2.3 Approximation of Optimal Flight Path Contour.

In the battlefield, operation time spent in the enemy's territory has to be minimized because exposure for a long time to the enemy's threat means that it has less of a probability to survive. Therefore, the optimal flight path needs to minimize the operation time while satisfying the mission constraints. In the problem posed in this research, satisfying the mission constraint means reducing the possible target location to an acceptable area of an ellipse. The area of ellipse is defined as

$$The\ Area\ of\ Ellipse = \pi \cdot a \cdot b \tag{3.10}$$

where a and b are as shown in Figure 14. The area of the ellipse can be used as a cost function in the optimal control problem. However, it has drawbacks when used as a cost function. As shown in Figure 15, it is hard to say that the smaller ellipse is always better than bigger ellipse area. Even though the area of 15a is smaller than that of 15b, the range in the direction of semi-major axis is too wide for determining the target position. So, the case like 15a is harder to use than 15b. As a result, area of an ellipse is still worth analyzing and can be used as a standard for targeting, but the semi-major axis of the ellipse is more adequate for use as a cost function than the area of it. It is easier and more efficient to use in the real world. The optimal

control problem in this research will be solved by minimizing the semi-major axis of the error ellipse.

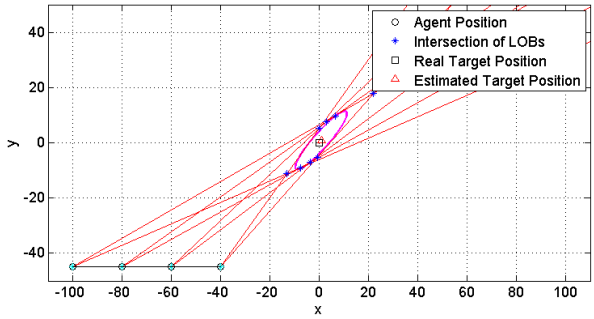


(a) Thin Ellipse (area = $1000 \cdot \pi$)

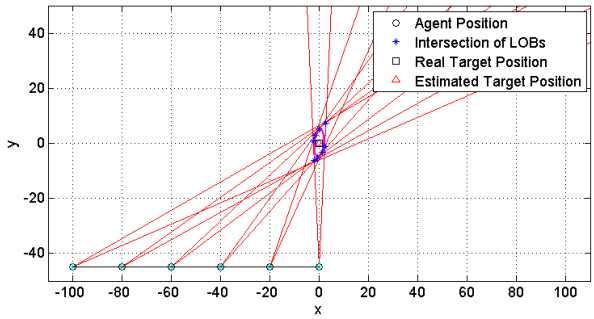
(b) Fat Ellipse (area = $1020 \cdot \pi$)

Figure 15. Comparison of Ellipse

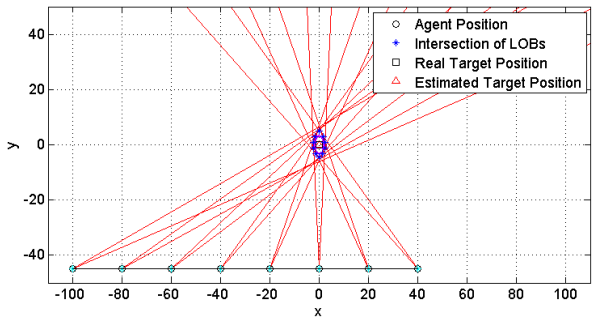
Before solving the optimal control problem, several typical flight paths can be assumed as an optimal path. The first one is a straight flight path. The distance from the agent to the real target is varied but the direction is the same on the flight path. In this example using a straight path, the velocity of the agent is assumed as 10 m/s, $\triangle t$ between measurements is 2 seconds, total measurement number is 11 and max measurement error is 3°. The LOB measurement is assumed that it has no error for analysis purposes, i.e., the two lines are $\pm$ 3° of truth.



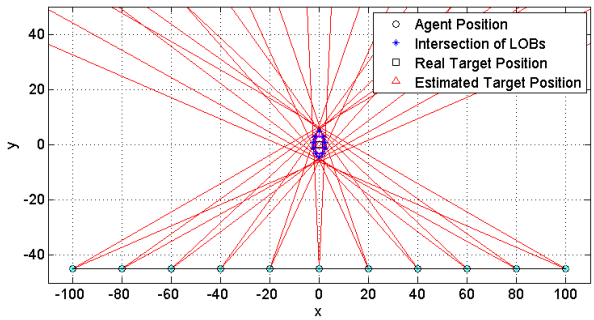
(a) t = 6



(b) t = 10

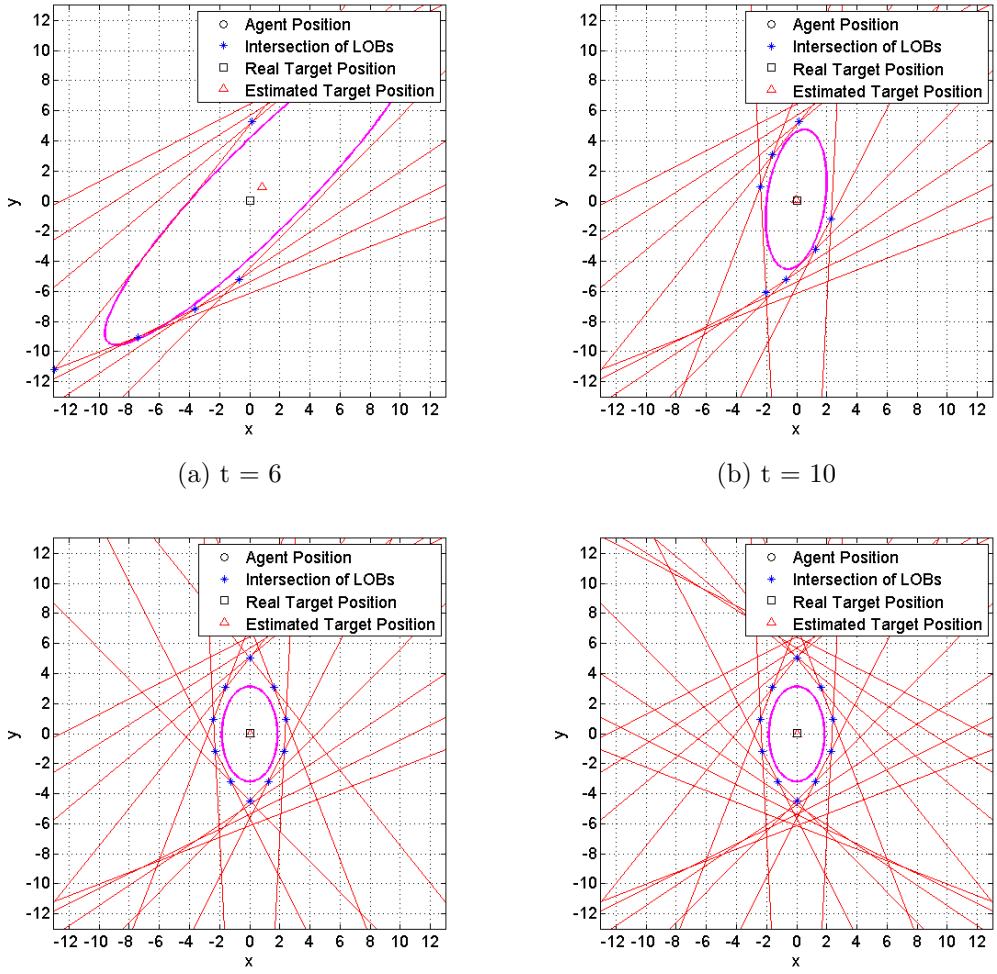


(c) t = 14



(d) t = 20

Figure 16. Straight Path & Error Ellipse

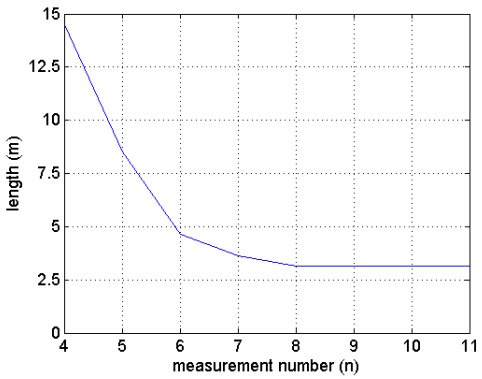


(a) t = 6 (b) t = 10

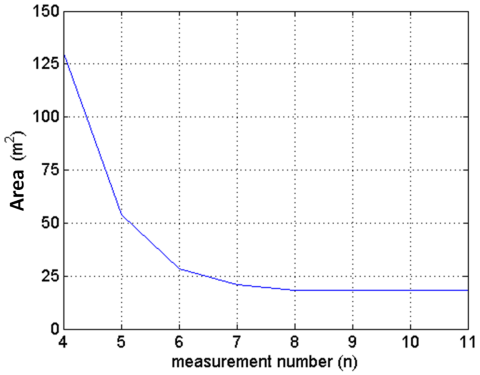
(c) t = 14 (d) t = 20

Figure 17. Error Ellipse of Straight Path

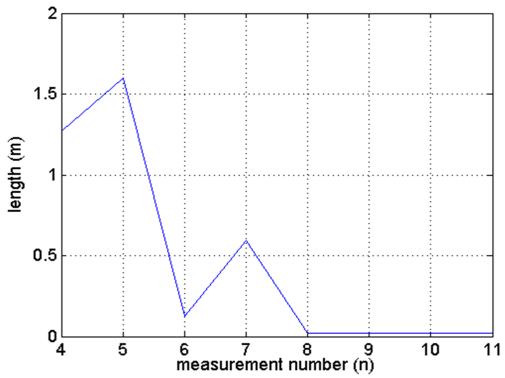
Figures 16 and Figures 17 show the true agent's straight path and the results. The initial ellipse shape shown in Figures 16a and 16b show that semi-major axis is oriented towards the agent's position but after the agent passes the same x value of the target position, there is not much change to the ellipse. In addition, it is impossible to see the effect on the shape of the ellipse after some amount of time as seen on Figure 17c and Figure 17d because the distance between the two lines of the same LOB is farther than the semi-major axis of the ellipse. There is no change after the 8th measurement as shown in Figure 18. As a result, the final semi-major axis is 3.15 m, the area of ellipse is 18.2 m^2 and distance between real target and the estimated



(a) Semi-major Axis of Ellipse



(b) Area of Ellipse

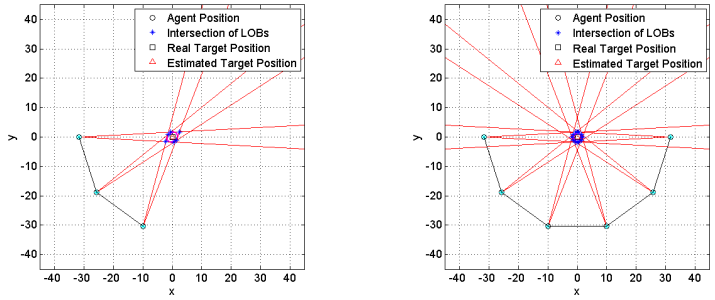


(c) Distance (Real - Estimated TGT)

Figure 18. Results of Using Straight Path

target is 0.02 m. The final result shows good targeting performance. However, the fact that it cannot minimize the semi-major axis after passing the same x value of the target position as in the example shows that it definitely has a disadvantage for use in targeting.

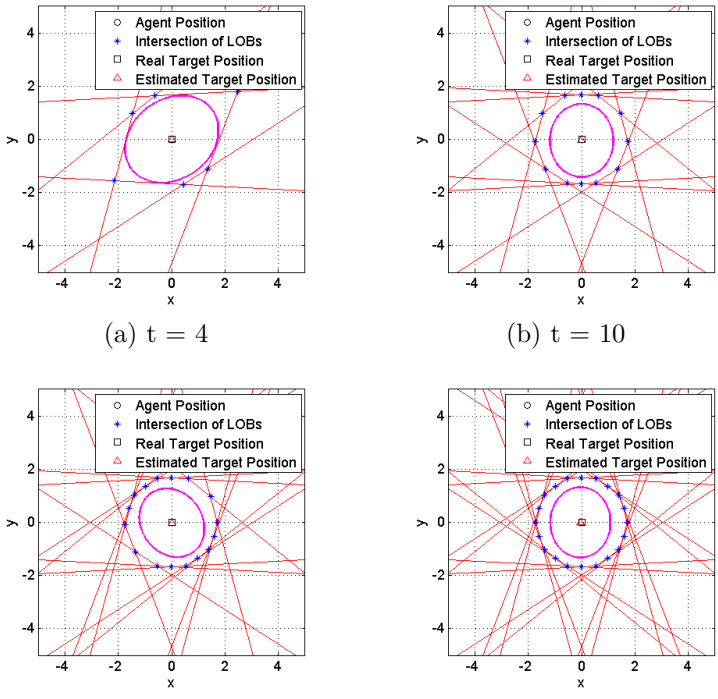
The second option is a circular flight path. A circular path is one path to represent a curved path solution. It maintains the same distance with respect to the target position but the angle changes between target and the x axis on the cartesian coordinate system. Figures 19 and 20 show an example flight path and the result of it. The velocity of the agent is 10 m/s, $\triangle t$ is 2 seconds, total time is 22 seconds and the max measurement error is 3°. The measurement LOB values are again assumed error free. Every other condition is the same as with the straight flight path case. The initial shape of



(a) t = 4 (b) t = 10

(c) t = 14 (d) t = 20

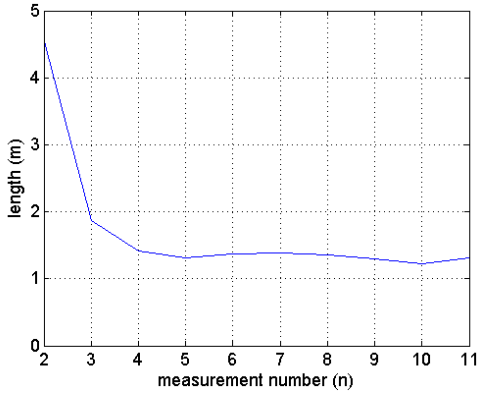
Figure 19. Circular Path & Error Ellipse



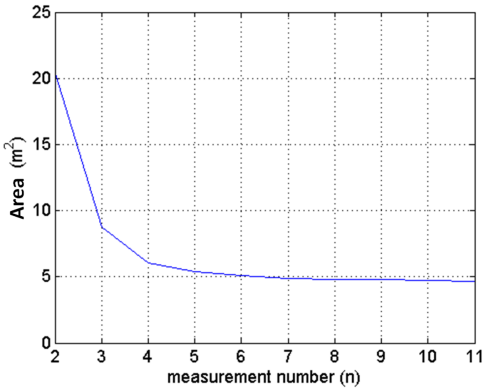
(a) t = 4 (b) t = 10

(c) t = 14 (d) t = 20

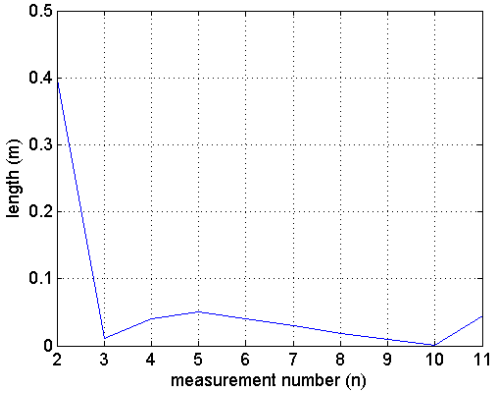
Figure 20. Error Ellipse of Circular Path



(a) Semi-major Axis of Ellipse



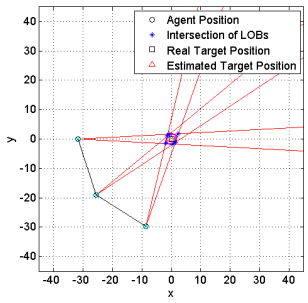
(b) Area of Ellipse



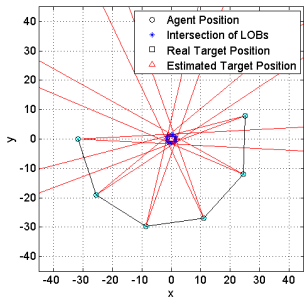
(c) Distance (Real - Estimated TGT)

Figure 21. Results of Using Circular Path

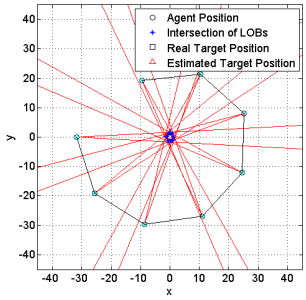
the error ellipse shows that its semi-major axis is oriented towards the agent as shown in Figure 20a like the initial shape from of straight path. However, Figure 20b shows that the error ellipse for the circular path is almost a circle. There is not much difference between the semi-major axis and semi-minor axis. This is because the LOBs of the agent creates an ellipse on the x-direction and y-direction evenly as shown in Figure 19b. Finally, Figures 19d and 20d show that the ellipse results in a circle after full measurements around the circlular path. This is reasonable because every measurement is taken in every direction on the circle path in even degrees. Every intersection on the circle shape is in even degrees, too. As a result, the final semi-major axis converges to 1.31 m/s, area of the ellipse is 4.66 m^2 and the distance between real target and estimated target is 0.04 m.



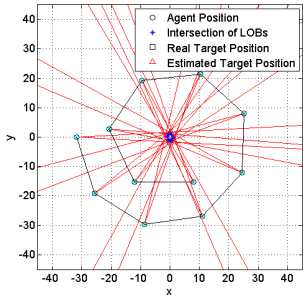
(a) t = 4



(b) t = 10

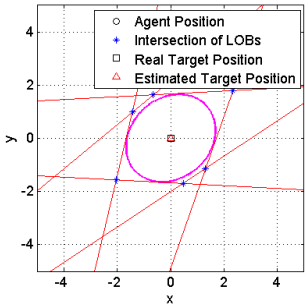


(c) t = 14

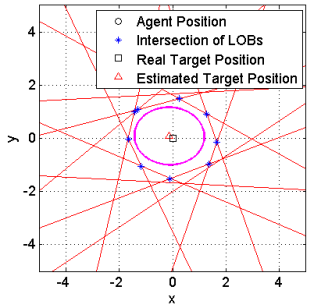


(d) t = 20

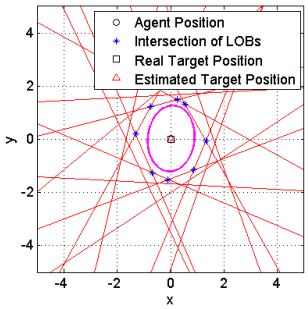
Figure 22. Spiral Path & Error Ellipse



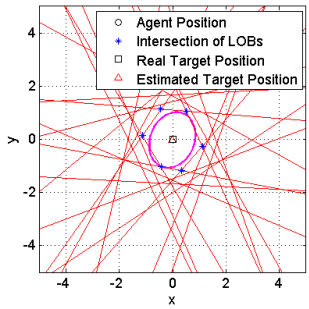
(a) t = 4



(b) t = 10

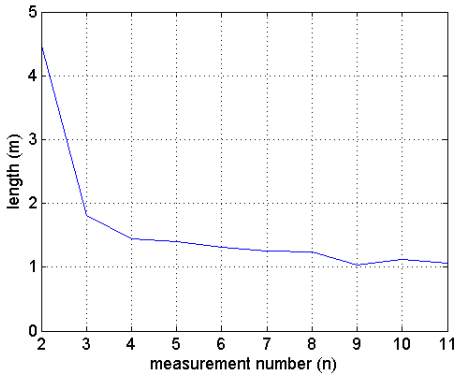


(c) t = 14

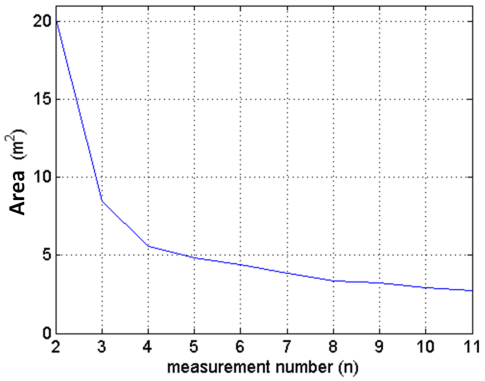


(d) t = 20

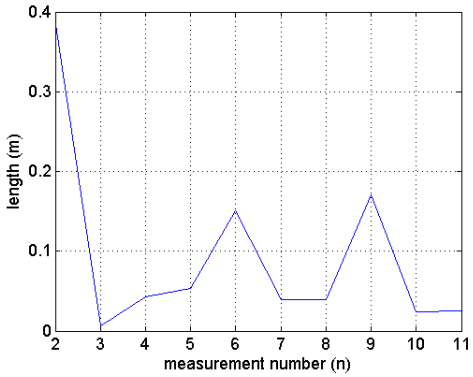
Figure 23. Error Ellipse of Spiral Path



(a) Semi-major Axis of Ellipse



(b) Area of Ellipse



(c) Distance (Real - Estimated TGT)

Figure 24. Results of Using Spiral Path

The third option explored was a spiral flight path. This path is another way to represent a curved path solution. The difference with the circular path is that distance to the target in the spiral path is being reduced while the agent is flying. Its flight path and results are shown in Figures 22 and 23. The agent's velocity is 10 m/s, $\triangle t$ is 2 seconds, LOBs are measured 11 times and the max measurement error is 3°. The measurement LOB values are again assumed error free for comparison with the other paths. Every other condition is the same as with the straight path and the circular path. The initial ellipse shape is very similar as with the circular path. The semi-major axis is oriented towards the agent position. However, it doesn't result in a circular shape and maintains typical ellipse shape even when it flies half of the total path as shown in Figure 23b. This tendency is not changed even after it passes 360° around the target. Its shape doesn't become circular because the distance is

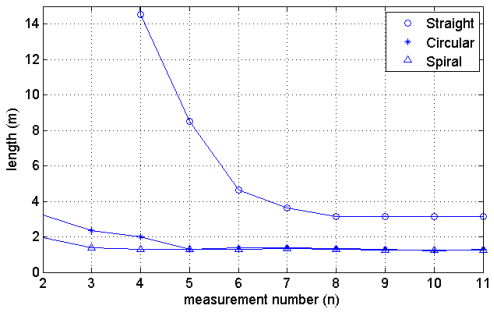
getting closer to target position. The LOB when taken near to the target position can shape the ellipse more than the LOB of the measurements taken farther from target position. Its result is better than the circular path with radius equal to the radius at time initial radius of the spiral path. The final semi-major axis converges to 1.07 m/s, area of ellipse is 2.76 m^2 and the distance between real target and estimated target is 0.03 m. These results show that the spiral path is more efficient than the circular path.

These three flight paths represent varied paths in real world. From the results, several characteristics can be seen on the error ellipse. First of all, each error ellipse is not plotted on the intersections. This is because the error ellipse is calculated using the error covariance matrix. If all intersections are on the same line, the error ellipse is expressed as a line on all intersections. However, all intersections for these example paths are put on the circular shape. If the axes of this circular shape are the same as the x-axis and y-axis, intersections on x-axis and y-axis have the maximum value with respect to axis. It is impossible for the error ellipse to pass through these intersections because there are other intersections have less values with respect to the same axis.

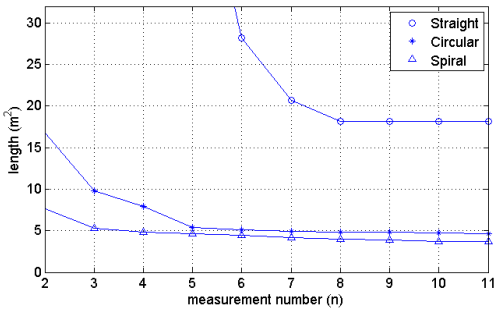
In addition, the distance graphs of the three paths are not asymptotic. This is because of the measurement positions. If the number of measurements near a certain axis is same as the number of measurements near the other axis, the distance value is decreased. Conversely, if the number of measurements near a certain axis is more than the number of measurements near the other axis, the distance value increased. This tendency can be shown in Figure 21b. The distance from the target to each measurement position is also a important factor. The measurement position nearer to the target effect stronger on this value. This effect is shown in Figure 24c.

Table 2. Example Results of Typical Flight Paths

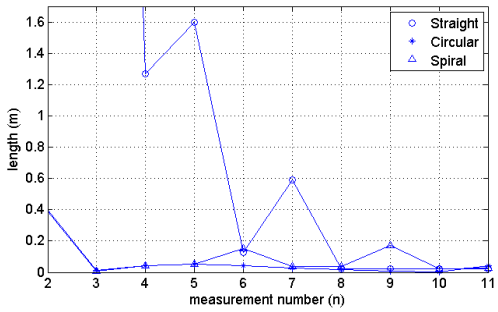
Path	Ellipse		Distance (m) (Real-Estimated)
	Semi-major Axis (m)	Size (m^2)	
Straight	3.15	18.2	0.02
Circular	1.31	4.66	0.04
Spiral	1.07	2.76	0.03



(a) Semi-major Axis of Ellipse



(b) Area of Ellipse



(c) Distance (Real - Estimated TGT)

Figure 25. Comparison of Typical Flight Paths

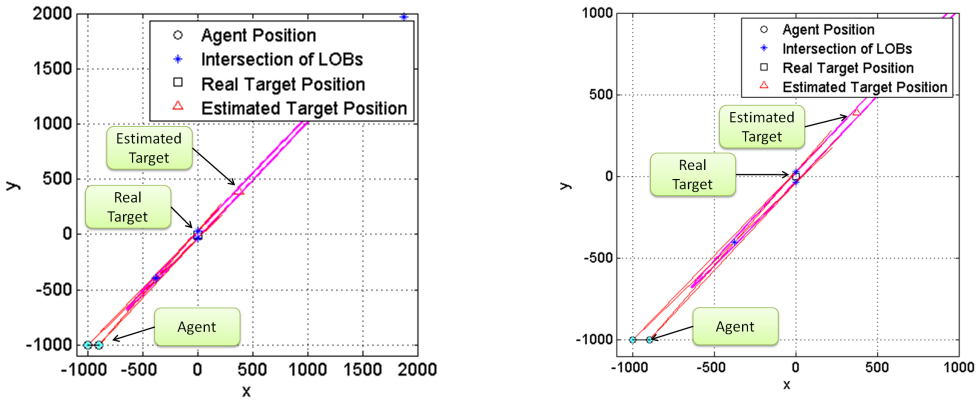
Table 2 shows the results of the three cases. The overall comparison of straight path and curved path including circular and spiral paths is impossible using only Table 2 because these value depend on the distance between agent and target. In these examples, the distance of the straight path and the curved path is different from each other. However, the straight path already shows its limitation in the example. It has an effect only on a single axis direction in the 2D space. A curved path can be assessed to be a more efficient path than the straight path. The comparison of

the circular path and the spiral path is possible. Their initial distance between the agent and the target is the same and the other conditions are also the same without the change in distance after departure. Every value of the spiral path distance to target is lower than for the circular path. The shape of the ellipse in the spiral path is more preferable than the shape in the circular path case and the distance between the real target and the estimated target is closer. As a result, the spiral path can be approximated as an optimal path in targeting problem using LOBs, or at least optimal from these three choices. Next, a true optimal solution will be computed for comparison.

3.2.4 Optimal Control Problem.

3.2.4.1 Initial Flight Path.

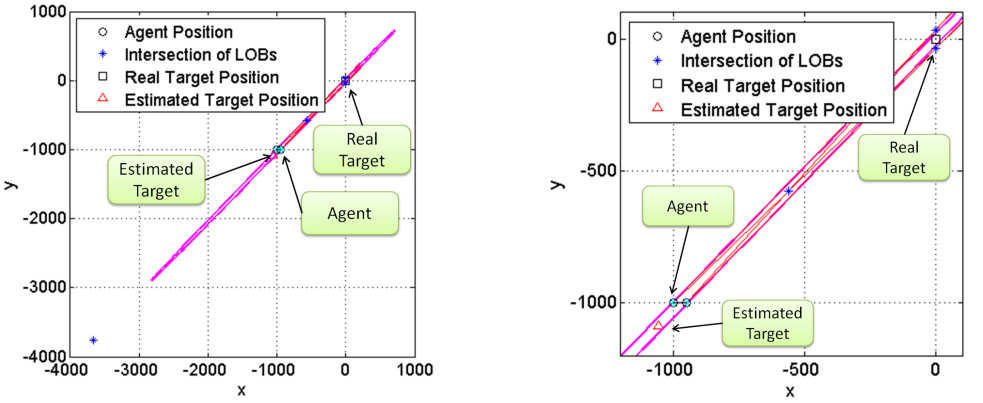
The error ellipse which was described before needs the intersections computed to form the ellipse. This requires that more than two LOBs are needed for the ellipse construction. Given an initial condition, the agents don't have enough information about the target position, so, they must maintain an initial direction until the ellipse is formed. As previously described, the minimum number of measurements to form the ellipse is two measurements. Figure 26 shows an example of the error ellipse formed with two measurements.



(a) Error Ellipse (b) Real Target & Estimated Target

Figure 26. Correct Ellipse Formation with 2 Measurements

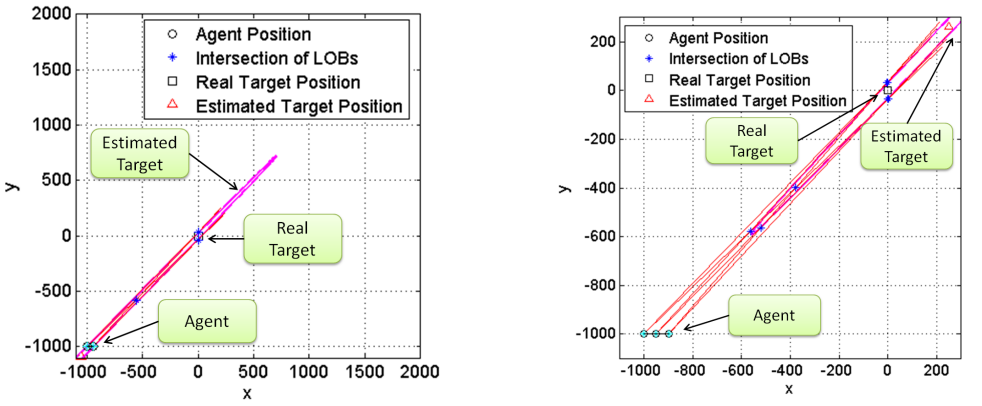
However, two measurements don't guarantee the correct ellipse formation. As shown in Figure 27, a incorrect ellipse is created because the estimated target position is in the wrong direction from the agent position. So, this ellipse cannot be used in the optimal control problem. It will move the agent in the reverse direction compared to real target position, and this case must be checked for and avoided.



(a) Error Ellipse (b) Real Target & Estimated Target

Figure 27. Incorrect Ellipse Formation with 2 Measurements

The difference between the two examples is the position where the agent measures the LOBs. The example shown in Figure 27 uses LOBs measured very near the position at the target. Actually, this problem is the result of several factors. First of all, maximum error value is one factor. A low value of maximum error makes the distance between the two lines per single LOB smaller. In this case, the small amount of change of the agent position brings an intersection behind the agent as compared with real target position shown in Figure 27a. In addition, the large value of the distance between the agent and the target is another factor. This causes a small amount of difference between two LOBs from each position. It results in identifying an incorrect intersection and it affects the estimated target position. The last factor is a small difference of the measurement position. This results in an incorrect intersection and an incorrect estimated target position, too. The measurement positions are decided by the velocity and measurement time step($\triangle t$). Confluence of these factors cause the singularity as shown in Figure 27.



(a) Error Ellipe (b) Real Target & Estimated Target

Figure 28. Ellipse Formation with 3 Measurements

To eliminate the singularity, the agent must check all intersections when the ellipse is generated. Every intersection needs to be put on each side of the estimated target position. If some of the intersections are on the reverse side with respect to agent position, it indicates that the error ellipse is not suitable to be used for the optimal control problem. In this case, it needs to gather more information of the target, LOBs. The agent should continue its heading and gather more LOBs. Figure 28 shows the case that one more measurement is taken than used in the Figure 27. As seen by Figure 28a, intersections are now on the correct side of the estimated position and the estimated position is in the correct direction to the real target position with respect to agent position. This technique can now be used to calculate the cost for use in the optimal control problem.

3.2.4.2 Cost Function & Constraints.

The answer to the optimal control problem depends primarily on the cost function used. In this research, the cost function is the semi-major axis of the error ellipse as described before.

$$J = max\,[\,\sqrt{\lambda\,(P)}\,] \tag{3.11}$$

where P is the error covariance matrix given in Equation 3.5. By using this cost function, the next measurement position which minimizes the semi-major axis of the

error ellipse can be calculated. The optimal flight path for getting a satisfactory semi-major axis value can be derived by iteration of this step.

The state vector for solving optimal problem is defined as

$$X = \begin{bmatrix} x \\ y \\ \psi \end{bmatrix} \tag{3.12}$$

where ψ indicates heading angle and the inequality constraint related to angular velocity effects on heading angle of the agent. In this research, angular velocity is assumed as 3°/sec. So, the inequality constraint on ψ is defined as

$$\dot{\psi} \leq 3°/sec \tag{3.13}$$

Additionally, x_i and y_i are affected by ψ. They function as the equality constraints to the optimal control problem. So, they are defined as

$$\begin{aligned} x &= x_0 + V \cdot \triangle t \cdot cos(\psi) \\ y &= y_0 + V \cdot \triangle t \cdot sin(\psi) \end{aligned} \tag{3.14}$$

where V is assumed the velocity of the agent and $\triangle t$ is assumed to be the measurement time step. As a result, the optimal control problem is described as

$$J = max\,[\,\sqrt{\lambda\,(P\,)}\,] \tag{3.15}$$

subject to

$$\begin{aligned} \dot{\psi} &\leq 3°/sec \\ x &= x_0 + V \cdot \triangle t \cdot cos(\psi) \\ y &= y_0 + V \cdot \triangle t \cdot sin(\psi) \end{aligned} \tag{3.16}$$

The optimal control problem with constraints can be solved by using the 'fmincon' function in MATLAB. 'fmincon' uses the Karush-Kuhn-Tucker (KKT) conditions for solving the optimal problem and the KKT conditions use the Lagrangian function defined as

$$L(x, \lambda) = f(x) + \sum_{i=1}^{n} \lambda_{g,i} g_i(x) + \sum_{i=1}^{n} \lambda_{h,i} h_i(x) \tag{3.17}$$

where $f(x)$ is a cost function, λ is a multiplier, $g(x)$ is the inequality constraint and $h(x)$ is the equality constraint. From this Lagrangian function, the KKT conditions are

$$\bigtriangledown_x L(x, \lambda) = 0 \tag{3.18}$$

$$\lambda_{g,i} g_i(x) = 0 \ \ \forall i \tag{3.19}$$

$$\begin{cases} g(x) \leq 0 \\ h(x) = 0 \\ \lambda_{g,i} \geq 0 \end{cases} \tag{3.20}$$

These KKT conditions are required for solving the optimal control problem.[3]

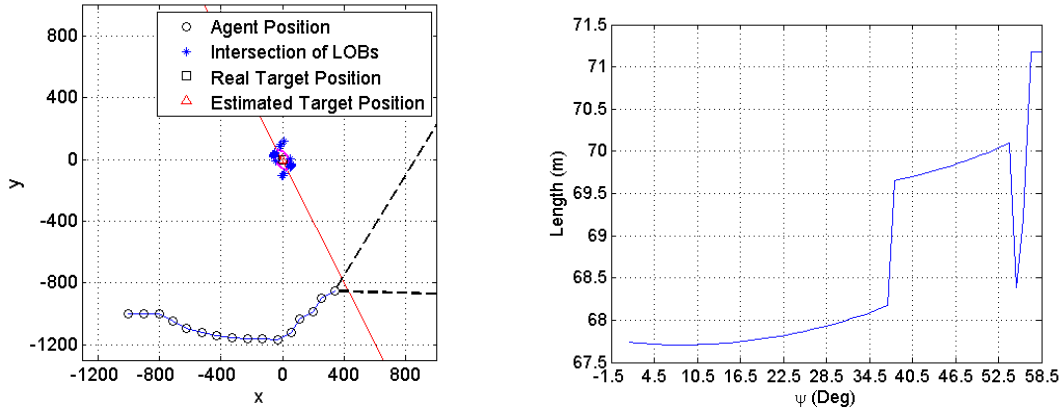
The 'fmincon' function finds a suitable state vector, x satisfying the KKT conditions. The initial value of x_0 is needed to be given. x_0 is applied to the the cost function and different x values are applied for comparison with the result of the cost function. Propagation of x is determined as

$$x_{(k+1,j)} = x_k + t_j \cdot d_k \tag{3.21}$$

where k is the iteration number and d_k is the search direction. The step size within each iteration, t_j, is defined as

$$t_j = \left(\frac{1}{2}\right)^j \quad j = 0, 1, 2, 3, 4, \cdots \tag{3.22}$$

With this propagating x values and the KKT conditions, 'fmincon' finds an optimal state vector. If the cost function is a convex function, it is simple to solve the optimal problem. This is because it has only one local minimum point and it is a global minimum point at the same time. However, if a cost function is non-convex, the problem becomes complicated. Normally, a nonlinear function has several minimum points. When some point is found as a local minimum point, it is hard to determine if this point is a global minimum point. The cost function used in this research is a non-convex function. As a result, it has the same problem mentioned before.



(a) Example Optimal Path (b) 14th Semi-Major Axis on Possible ψ

Figure 29. Multiple Local Minima Example

Figure 29 shows this case well. In this example, the agent starts from (-1000, -1000) and the target is at (0, 0). Its maximum measurement error is 3° and the agent speed is 10 m/s. The agent is flying to the 14th optimal path point in Figure 29a. Its possible heading angle at the 14th point is -1.5° $\sim$ 58.5°. This heading angle decides the next semi-major axis of the ellipse as shown in Figure 29b. In this graph, there are two local minimum points when ψ is 10.46° and 55.46°. So, the local minimum value of ψ depends on which value is used as ψ_0.

Elimination of the local minima ambiguity can be done by using an approximated flight path as was previously shown. The optimal flight contour is approximated as a spiral path as before. There are several ways to eliminate local minima ambiguity using an approximated contour. First of all, ψ_0 needs to be directed towards the

estimated target point. It will help to find the local minimum point to move the agent to the target position as near as possible. Furthermore, moving the agent to the nearest position to the estimated target is helpful when an ambiguity is encountered. The first solution is to move the agent to the position nearer to the target. However, it doesn't guarantee to move the agent closer to the target compared with former agent's position. Sometimes, a local minimum ψ value results in the agent moving farther from the target. Even though it can temporarily reduce the semi-major axis, it will adversely effect the future semi-major axis value. So, moving to the farther position from the target position needs to be avoided for the future semi-major axis. As a result, ψ_0 is needed to be set to the direction closer to the estimated target position, but if the agent moves to a farther position from the estimated target position as compared to the former position, the agent is required to move to the nearest position to the estimated target position. The decision logic for this method is shown in a flow chart in the next section.

3.2.4.3 Optimal Flight Path Algorithm.

An optimal flight path is generated by the iteration of solving the optimal control problem. The cost function and constraints is as suggested in Section 3.2.4.2. The solution of the optimal control problem supplies the next point to move for the agent. The repetition of these steps produces the sequence of points to defining the path. The line connecting each point is the optimal flight path.[1] However, it needs to create an error ellipse before solving the optimal control problem and all intersections are on the estimated target's side preventing the agent from going in the wrong direction. In the optimal control problem, the local minima ambiguity can generate ambiguity to the flightpath. So, a filter preventing local minima ambiguity is needed and an assumption of the flightpath is referenced to do it.

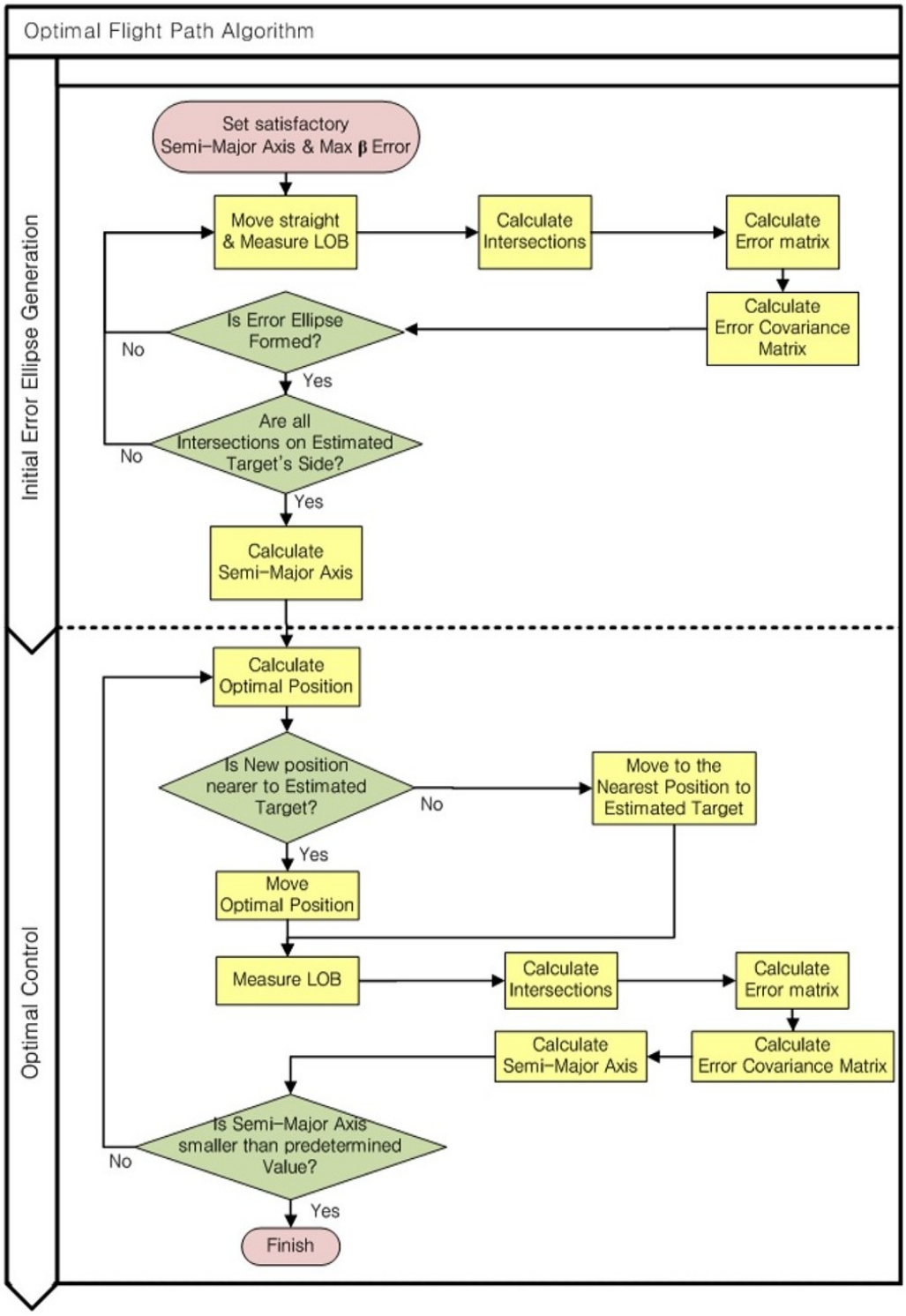


Figure 30. Optimal Flight Path Algorithm

The optimal flight path is propagated using the process as described in Section 3.2. First of all, satisfactory semi-major axis(user requirement) and ε error information(hard specification) are needed before applying the algorithm. The whole process is divided in two parts.

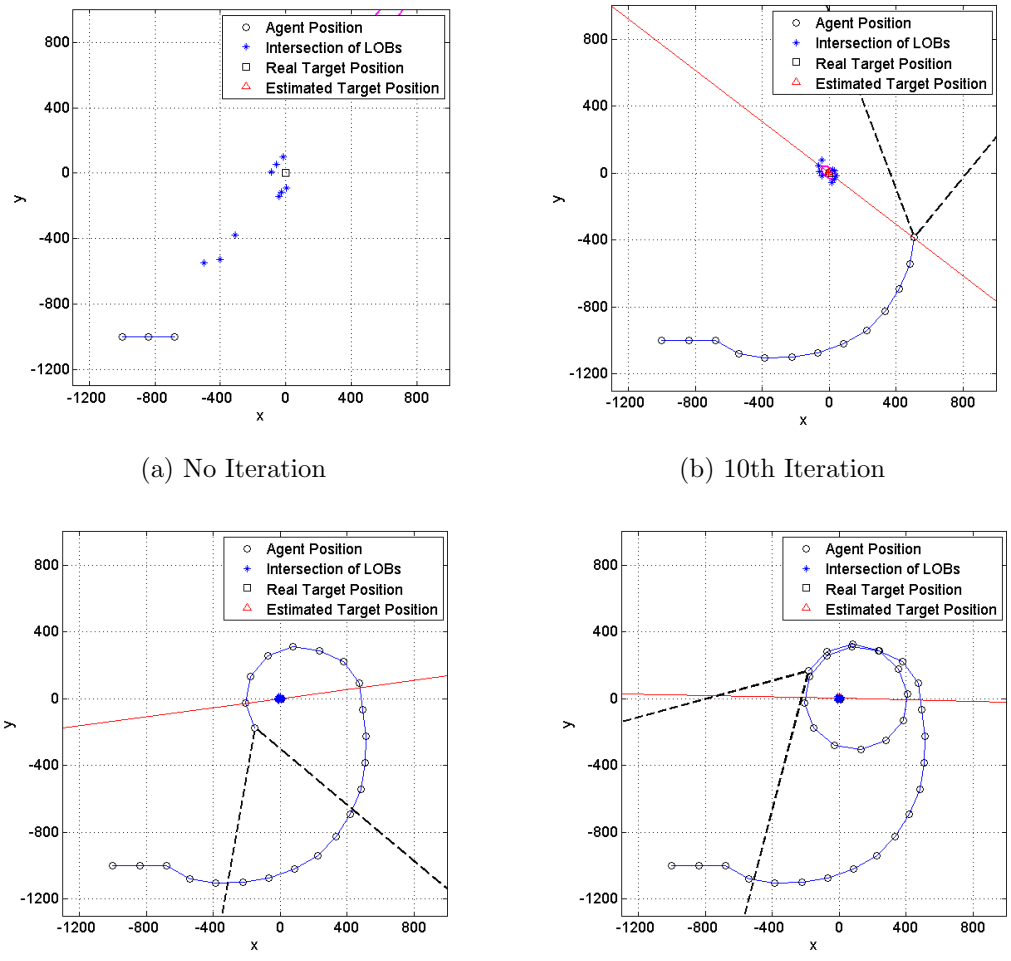
The first part is an initial error ellipse generation. This process is the preparation for running the optimal control of the agent. The objective of part one is to gather enough information for solving the optimal control problem. In this process, the agent flies a straight path until it makes a proper error ellipse. First the agent flies straight and measures the LOB information. Based on the position information and the measured LOB value, the agent calculates intersections of lines described in Section 3.2.1. Error matrix and error covariance matrix can be calculated using intersections. If the error ellipse can be formed with this error covariance matrix and all intersections are on the estimated target's side, the agent calculates the semi-major axis. If it is not, it moves straight and measures more LOBs.

The second part is the optimal control process. This process makes the agent move to the best position for reducing the semi-major axis of the error ellipse. However, it has a filter in place to prevent the local minima ambiguity as described before. The result of the optimal position needs to be checked whether it is a closer position to the estimated target point as compared with the former position. After moving to the next point, the agent measures the LOB and calculates intersections, the error matrix, the error covariance matrix and the semi-major axis. While doing this optimal process, its result needs to be checked whether its semi-major axis is smaller than the predetermined satisfactory semi-major axis value. If it is smaller, the algorithm is finished. The entire algorithm is shown in Figure 30. The next section will implement this algorithm.

3.3 Sample Result for Models

Figure 31 shows the sample result when using this algorithm. For doing this, V is set to 16 m/s, $\triangle t$ is 10 seconds and the max measurement error is assumed

as 3°. Velocity of the agent is based on the real air frame intended for testing, the SigRascal 110. Its cruise velocity is $28 \sim 40$ $knots(14.40 \sim 20.58$ $m/s)$[4]. So, 16 m/s is determined as the velocity of agent. Figure 31a shows the initial error ellipse generation step. Every intersection is on the side of the estimated target position. It is ready for part two, the optimal control step.



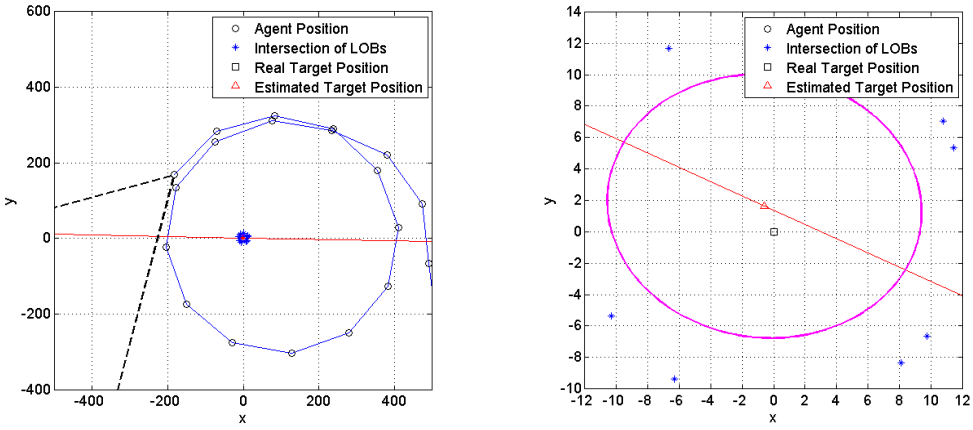
(a) No Iteration

(b) 10th Iteration

(c) 20th Iteration

(d) 30th Iteration

Figure 31. Sample Optimal Flight Path



(a) Loop Shape of Flight Path (b) Error Ellipse

Figure 32. Last Part of Optimal Flight Path

For the optimal control solution, the agent as it progresses, the path has one tendency. The error ellipse on every figure has a red line which shows the semi-major axis direction. The semi-minor axis is normal to this red line. The agent in Figure 31 flies to the semi-minor axis. It is very reasonable because turning on the side of the semi-minor axis is more efficient. Taking the path on semi-minor axis can shape the direction of the semi-major axis. Conversely, taking measurements on the semi-major axis side, it is possible to shape the direction of the semi-minor axis.

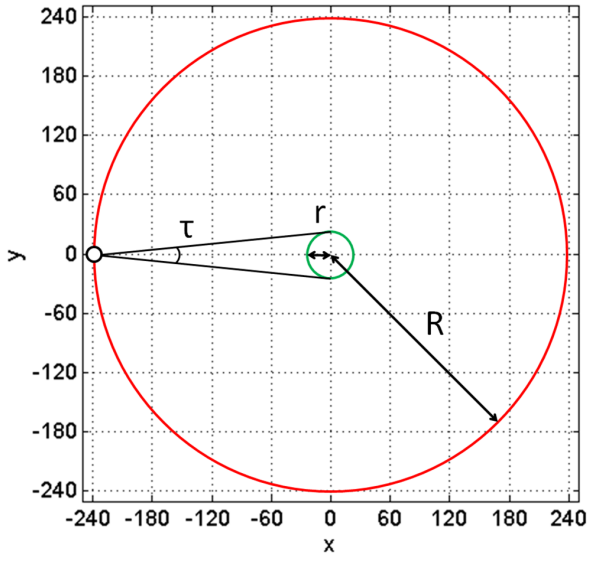


Figure 33. Predetermined Semi-Major Axis

Another trait of the optimal flight path is that it shows a loop shape in the final part with a tight turn radius. This is shown well in Figure 32. This loop shape can give a clue to limit the predefined semi-major axis value. In Figure 33, the large circle indicates a loop shape path and a small shape indicates an example of the error ellipse. In Chapter II, a circular path results in a circular shape of the error ellipse, so, the final shape of the error ellipse is circular. The radius of the large circle, R, is decided by the angular velocity, ω. R is defined as

$$R = \frac{v}{\omega} \tag{3.23}$$

for constant velocity and τ represents the possible LOB value. So, its value is the same with two times of the maximum measurement error value. Using τ and R values, r is defined as

$$r = R \cdot \frac{\tau}{2} \tag{3.24}$$

In this sample problem, V is set to 16 m/s and ω is $3°/s$. R is calculated as 305.58 m and r is determined as 16 m geometrically. These values are very similar with the simulation. In simulation, R is 309.09 m and r is 10.01 m. The reason why r is different for both cases is that the error ellipse is not on the center of the loop shape of the path in the simulation. As shown in Figure 32a, the error ellipse is canted to the left side of the circular path. However, this low value is generated accidentally. It is hard to anticipate a tilt of the error ellipse inside of the circular path. As a result, the predefined semi-major axis value needs to be over 16 m in this case and it is helpful to use this to escape an infinite loop in the algorithm.

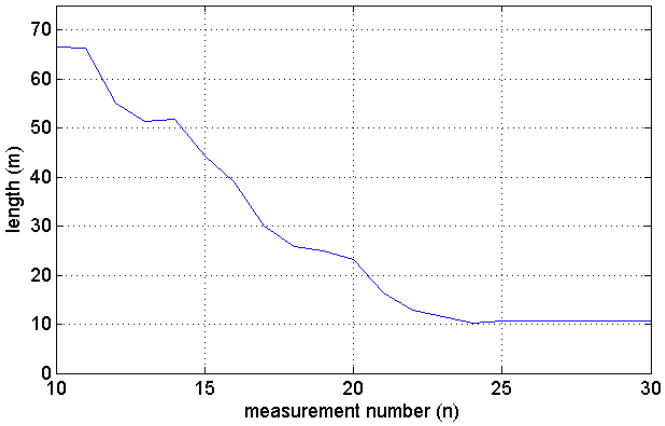


Figure 34. Semi-Major Axis of Ellipse

The semi-major axis is reduced continuously while the number of measurements is increased as in Figure 34. Sometimes, however, it is increased by a small value but it is decreased again with the following measurement. At last, its reduction is stopped on the 25th measurement because of the geometric reason described previously. From the 25th measurement onward, the agent starts a loop shape flying as shown in Figure 31d. However, the change of the semi-major axis for the loop shape the agent is flying is very small.

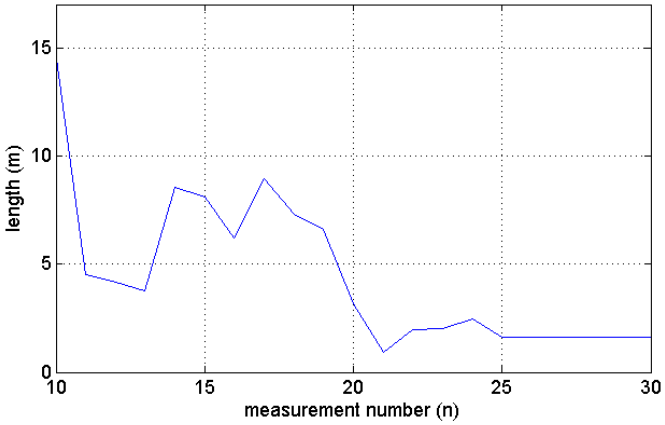


Figure 35. Distance (Real - Estimated TGT)

Often, the distance between the real target and the estimated target fluctuates in the beginning as shown in Figure 35. However, the most important thing is that it is still inside of the error ellipse and the distance is still lower than the semi-major axis so it is decreasing along with reduction of error ellipse at last. Its reduction is almost stopped on the 25th measurement, like the semi-major and there is not much change while flying the loop path. Next, the sensitivity to the different parameters used in the simulation is explored.

3.4 Sensitivity

The targeting algorithm developed provides varied results for several different factors. Especially, the first step of the algorithm is to set a satisfactory semi-major axis and max ρ error value. The results obtained from the algorithm is very dependent on these two factors. For finding sensitivity of the algorithm, the same assumption is used with the Section 3.3.

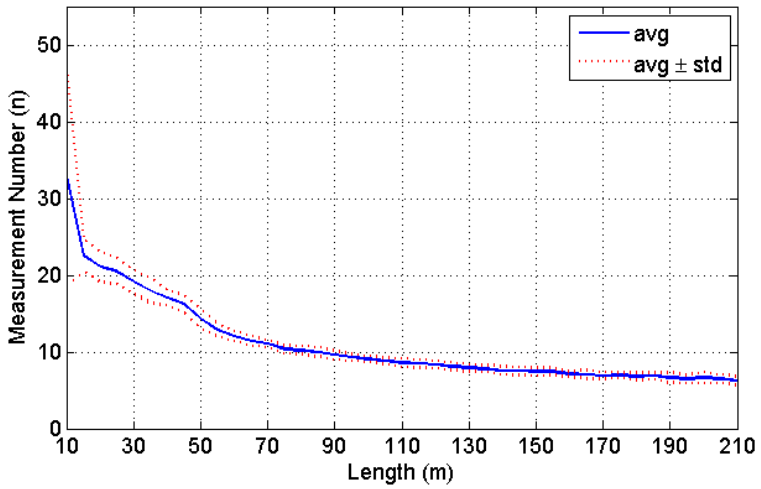


Figure 36. Measurement Number vs. Predefined Semi-Major Axis($\varepsilon = 3°$)

A satisfactory semi-major axis results are tied directly to the time to get to the predefined semi-major axis value. A small value of the semi-major axis requires a longer time to get to it. Because the small semi-major axis needs more measurements

and a closer measurement position to the target. These results are shown in Figure 36 while varying the predefined semi-major axis from 10 m to 210 m. Average values and standard deviation values are calculated using the results of 100 simulations. This result uses the same conditions as with Section 3.2.5. It shows that the measurement number is inversely proportional to the predefined semi-major axis value. However, its slope is smaller as the predefined semi-major axis is increased. It means that the measurements in the beginning can affect the semi-major axis value more and its effect diminishes with an increasing number of measurements.

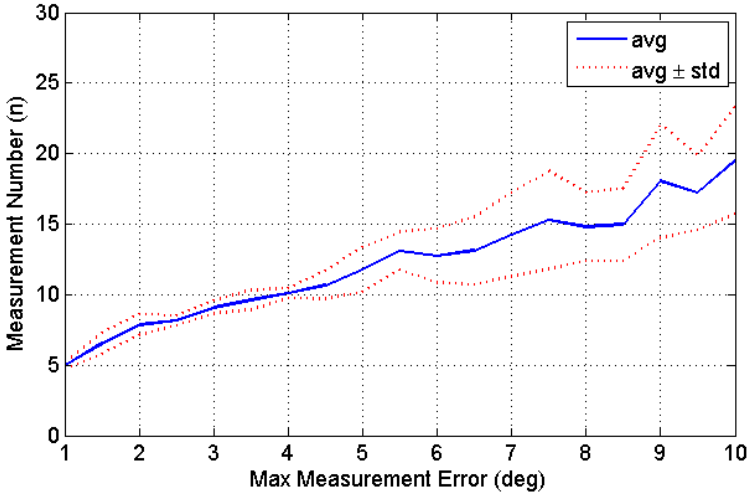


Figure 37. Measurement Number vs. Max Measurement Error($\rho_d = 100m$)

The measurement error bound, ε, is another factor which effects the required number of measurements. The semi-major axis of the ellipse is directly connected with ε because it decides the range of $\tilde{\beta}$. Small ε is more advantageous to shape the ellipse. The relationship between ε and the required measurement number is shown in Figure 37. As described before, a low value of maximum measurement error requires fewer measurements. However, it is not valid in all the cases. Even though the overall tendency shows that the measurement number is increased with a larger maximum measurement error, some cases show more measurements with less maximum measurement number. This is because moving to the nearest position in

the optimal control process doesn't always guarantee a smaller semi-major axis of the ellipse. The error ellipse is decided by the intersections of LOBs. So, intersections created by a new LOB can create a larger ellipse than the former one. However, this case is caused randomly and it is impossible to control because of the measurement error in the real world. As a result, the tendency of measurement number versus max measurement error is considered correct although there are individual cases where this trend may not hold.

3.5 Scenario

3.5.1 Overview.

As specific scenario was devised and is divided into two cases. The first one is a single target localization. It is similar to the sample result taken before but the initial distance from the agent to the target is farther than the distance of shown previously. This makes the agent fly longer and show the flightpath in detail. Changes to the length of the semi-major axis and the distance between real target and estimated target is checked in detail.

In addition, localization for multiple targets is described. The LOBs from different targets are taken at the same time while the agent flies. So, the agents sort these LOBs and make two different error ellipse. The algorithm for multiple targets are explained at first and its flightpath is explained. Additionally, the change of length of semi-major axis and distance between real target and estimated target is measured.

3.5.2 Single Target.

This section describes the specific result of a single target in a certain condition. For doing this, V is set to 16 m/s, $\triangle t$ is 10 seconds and 3° is used as max measurement error, ε. The single target is at (0, 0) and initially the agent is at (-2000, -2000). A satisfactory semi-major axis is set to 10 m.

The optimal flight path for single target is shown in Figure 38 based on using the algorithm in Figure 30. Its shape has the common traits with the sample result shown in Figure 31. The initial flightpath maintains its heading until the ellipse is shaped and overall flightpath has a spiral shape for targeting. Finally, the flightpath for the single target shows a loop shape at the end. In Figure 40, the change in heading angle is shown and it shows the same derivative in the end. This shows that the agent is looping around a target.

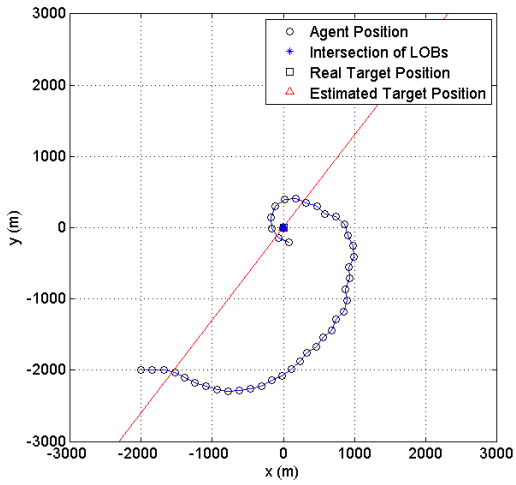
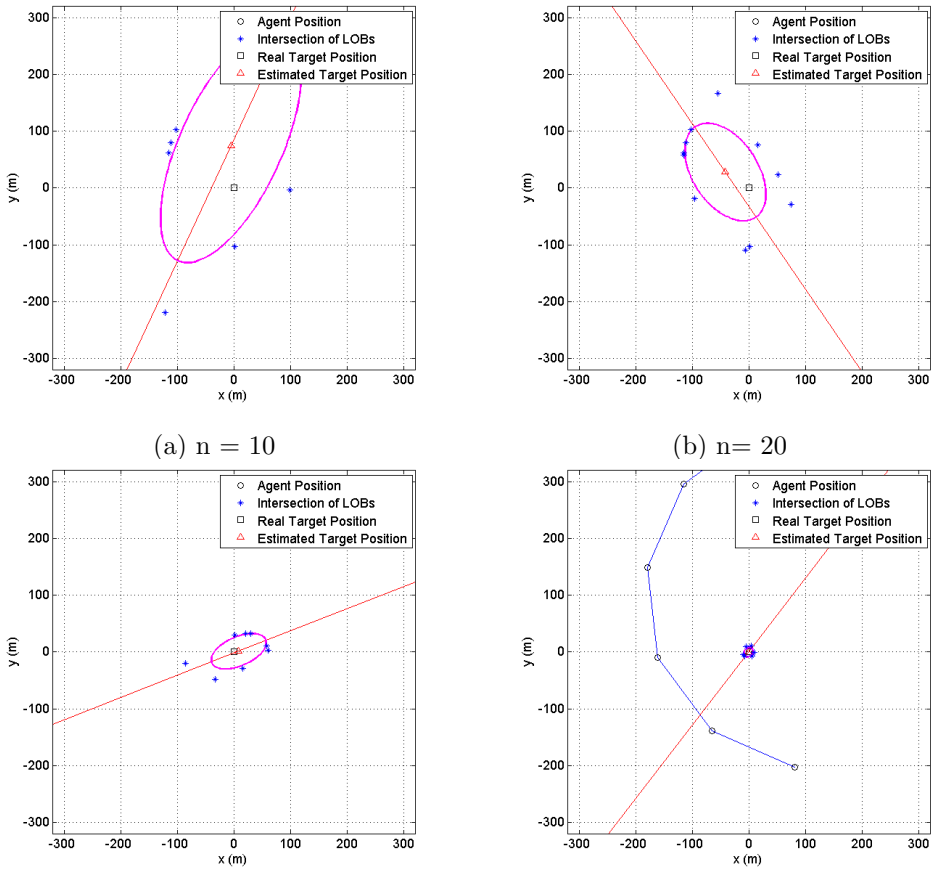


Figure 38. Optimal Flight Path for Single Target



(a) n = 10

(b) n= 20

(c) n = 30

(d) n = 41 (Final Ellipse)

Figure 39. Error Ellipse for single Target

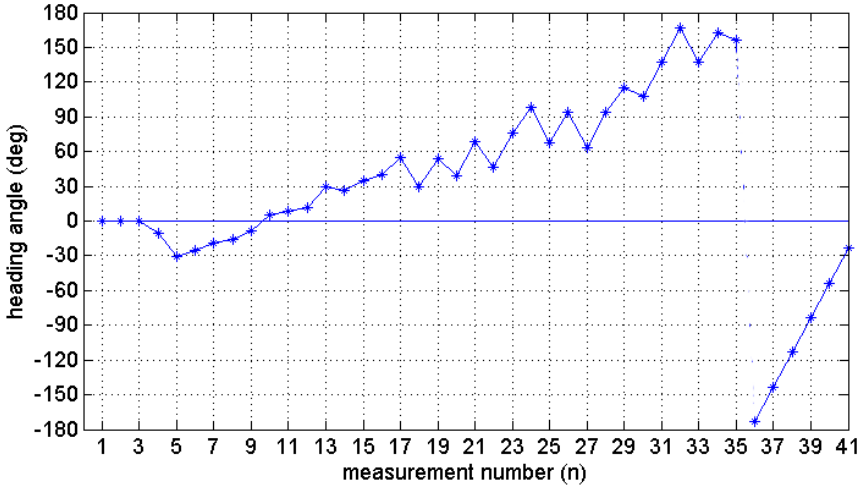


Figure 40. Heading Angle for Single Target

The shape of the error ellipse is shown in Figure 39. The semi-major axis is oriented towards the agent like the sample results. In addition, its area decreases while the agent flies the optimal path. Finally, its semi-major axis satisfies its predefined value on the 41th measurement position, $t = 410$ sec.

The semi-major axis value is drastically reduced in the beginning of the flight path as shown in Figure 41. While time is increasing, the rate of the semi-major axis reduction is decreased. However, the semi-major axis is continuously decreased by adding measurements until it satisfies the predefined value. As a result, its final value is 7.61 m. The distance between the real target and the estimated target position is decreasing though the graph and is fluctuating as shown in Figure 42. However, it is clear that the decreasing distance is its tendency and the amplitude of the fluctuation is also decreasing. Surely, the distance value for every step is lower than the semi-major axis value. So, every time when the agent increases its number of LOBs, the tendency is to get closer and closer to the target. In this scenario, the final distance value is 0.70 m.

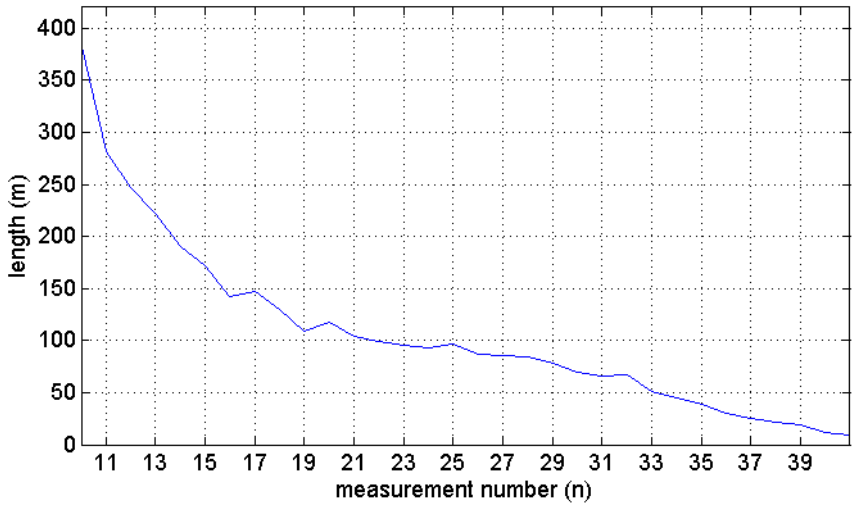


Figure 41. Semi-Major Axis of Ellipse for Single Target

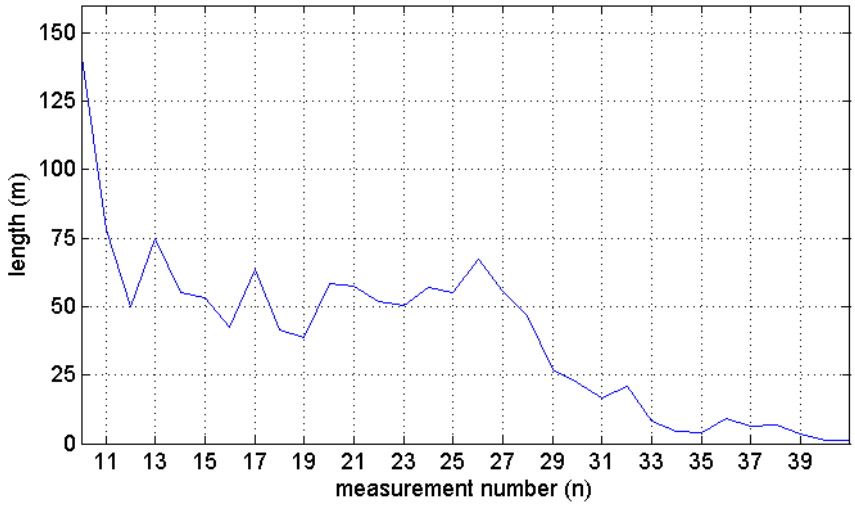


Figure 42. Distance (Real - Estimated TGT) for Single Target

3.5.3 Multiple Targets.

This section shows the results where there are multiple targets. Its assumptions are the same as that of a single target. In addition, the agent is assumed to be capable of measuring multiple LOBs separately and distinguish each other. This can be possible by using a rotating directional antenna [10]. A rotating directional antenna can make time series in LOB measurements data. So, it can distinguish each from the other when it receives multiple signals that have the same frequency. Besides, when the agent knows the channel of every target's radio signal, each LOBs measurement data from the different channels can be distinguished by measuring every channel in each time step with a rotating directional antenna. In this section, multiple targets are at (0, 0) and (0, 2000).

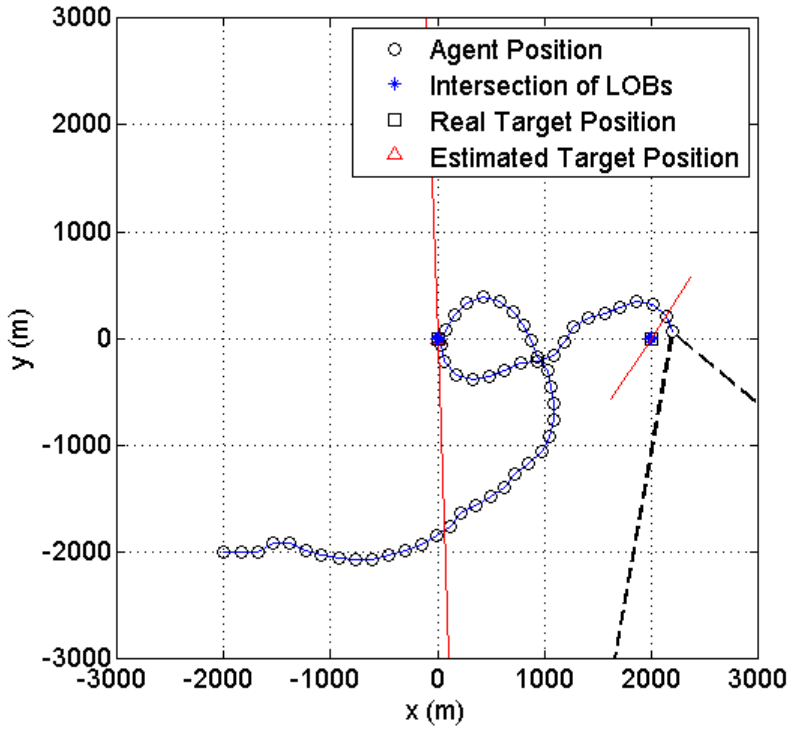
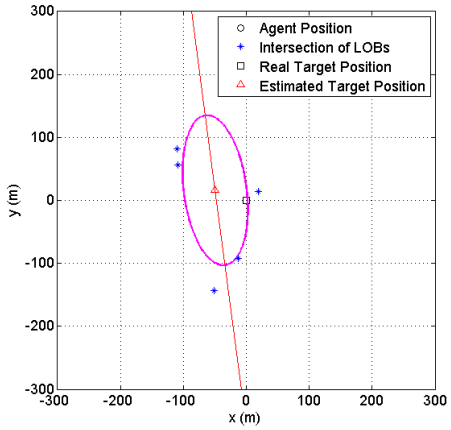


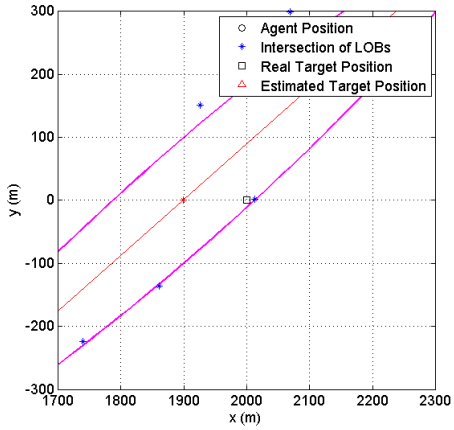
Figure 43. Optimal Flight Path for Multiple Targets

The algorithm for multiple targets are a composition of the algorithm for the single target. First of all, the agent decides which target is nearer from the agent position by using the estimated targets' positions. It follows the optimal path for the nearer target while measuring LOBs for multiple targets. After getting satisfactory semi-major axis of the nearer target, it follows the optimal path for the farther target. The optimal flight path looks like a composition of optimal paths for two single targets.

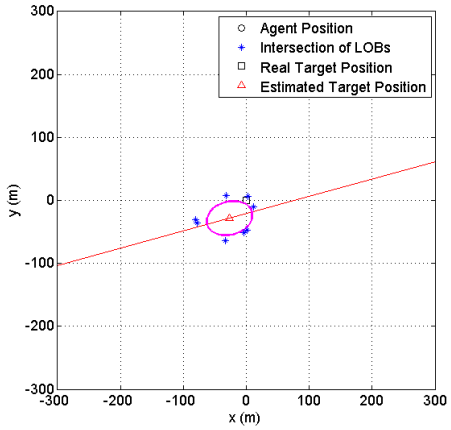
Figure 43 shows the optimal flight path for these multiple targets. This optimal path has several characteristics. First of all, the ellipse error of the farther target is bigger than that of the nearer target. This is because of the effect of the LOBs measured from the further distance target is lower than that of the nearer target and the initial optimal path is for the nearer target.



(a) n = 15 / TGT ♯1

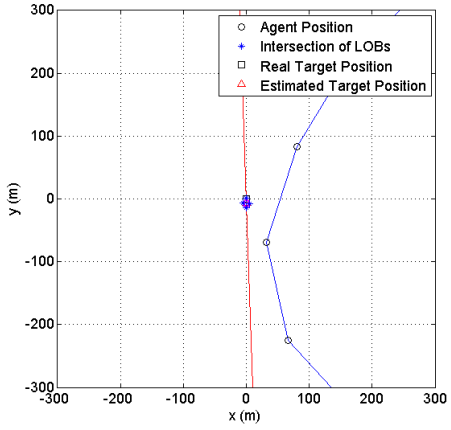


(b) n= 15 / TGT ♯2



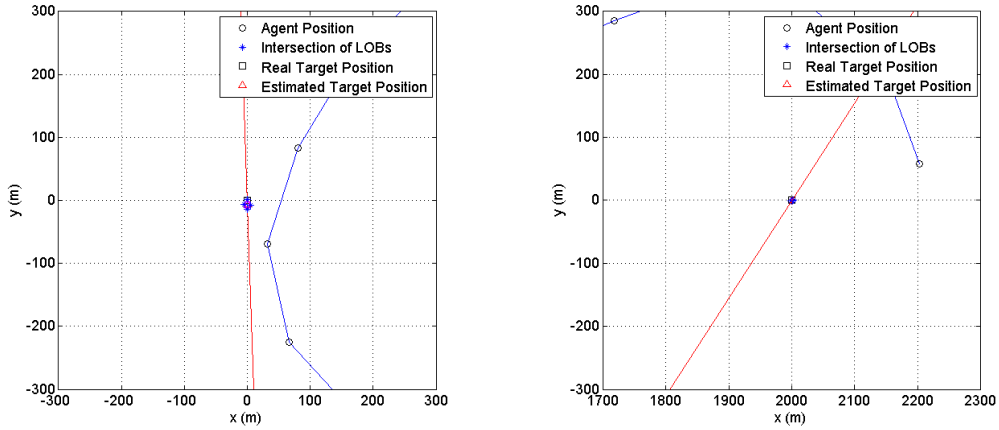
(c) n = 30 / TGT ♯1

(d) n= 30 / TGT ♯2



(e) n = 45 / TGT ♯1

(f) n= 45 / TGT ♯2



(g) n = 51 (Final) / TGT ♯1 (h) n= 51 (Final) / TGT ♯2

Figure 44. Error Ellipse for Multiple Targets

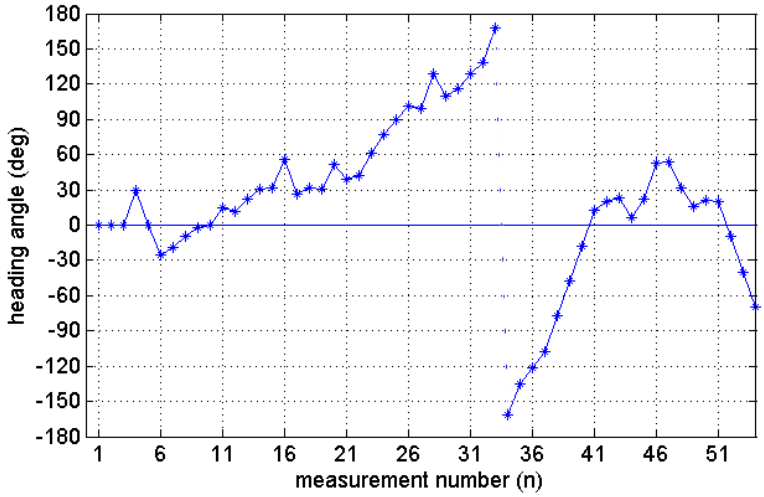


Figure 45. Heading Angle for Multiple Target

Secondly, the optimal path shows the loop shape on Figure 44e but it doesn't loop around the nearer target. This is because the satisfactory semi-major axis is achieved when the agent arrives right under the target position, (-5.5514, -35.6846). So, it can fly to the second target from that position.

The long radii of the ellipses are shown in Figure 46. Because of the algorithm for multiple target as described before, the overall semi-major axis for the farther target is larger than it is for nearer target. For obtaining a satisfactory semi-major axis for

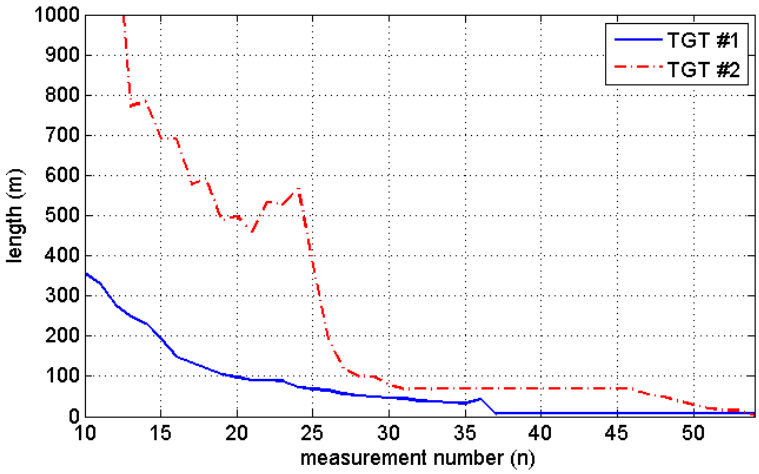


Figure 46. Semi-Major Axis of Ellipse for Multiple Target

the nearer target, the agent needs 37 LOBs and it takes only 17 more LOBs for the farther target. This is because the agent continuously measures LOBs for both of them from the beginning. By doing this, the ellipse for the farther target becomes smaller continuously even though the agents flies to minimize the nearer target error ellipse.

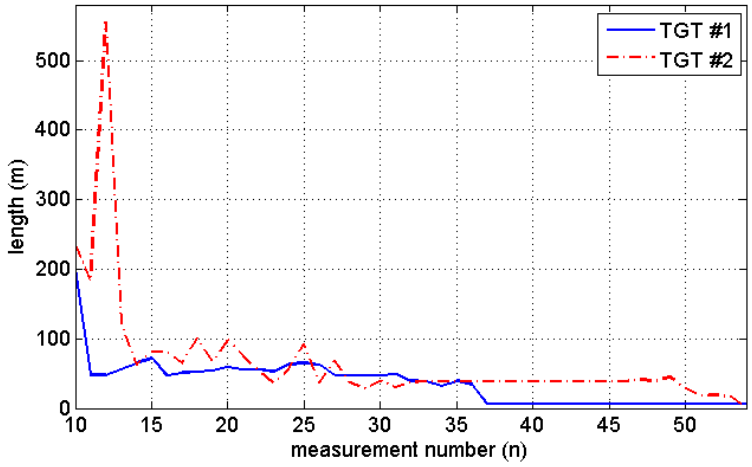


Figure 47. Distance (Real - Estimated TGT) for Multiple Target

Figure 47 shows the distance between the real targets and the estimated targets. The final distance between real target and estimated target for the target #1 is 1.5 m

and for target #2 is 0.3 m. The contours of Figure 46 and Figure 47 are very similar. Normally, the values for the nearer target are lower than for the farther target but this relation is reversed in some ranges as shown in Figure 47. Even though there are several exceptions, it can be assumed that the semi-major axis is proportional to the distance between the real target and the estimated target.

3.6 Summary

An algorithm based on a geometric approach is described in this chapter. The significant characteristic of this algorithm is that it uses two lines by adding and subtracting maximum error, ε, from the measured LOBs. These two lines provide a possible region and LOBs from varied positions creates intersection regions. The intersection of these regions provide an estimated target position. The intersection of this region is called an error ellipse because its shape is an ellipse made by measurement error. The error ellipse area is determined by taking measurements at different positions on the flightpath. Among the varied flightpaths, the spiral path is assumed as an optimal path contour. This contour supports the method to eliminate singularities in the optimal control problem.

The results for a single target scenario was satisfactory. By doing optimal control, this algorithm supplies an optimal path for minimizing the error ellipses. Using semi-major axis as the metric to determine the area of the ellipse can meet the predefined value. In addition, the result of using multiple target paths can be derived from the result of single target scenario. Its shape is the composition of two single target optimal paths. It shows that the composition of multiple single target optimal paths can be used to geolocate more targets.

IV. Test & Analysis

4.1 Overview

A real-world test is done to verify the algorithm described in Chapter III. Even though it worked well in simulation, it can have problems and be useless in the real world. So, what are the pitfalls for this algorithm and how can they be resolved is the objective for this test. This thesis is motivated for use with UAVs but a remote controlled truck is used for experimental test instead of real UAV. Using a truck saves time for flight preparation like assembling the airplane and safety consideration. This real test was conducted on 22-23 Dec 2015 around the National Museum of the United States Air Force. To conduct the test, a large empty space was needed for maneuvering the truck around the target. The parking lot of National Museum of the United States Air Force was a good place to do it with an RC truck.

Chapter IV describes the simulation result based on geographical information of the parking lot. Next, equipment for the experiment is described in detail. Finally, the analysis and comparison between simulation and real test is presented.

4.2 Simulation

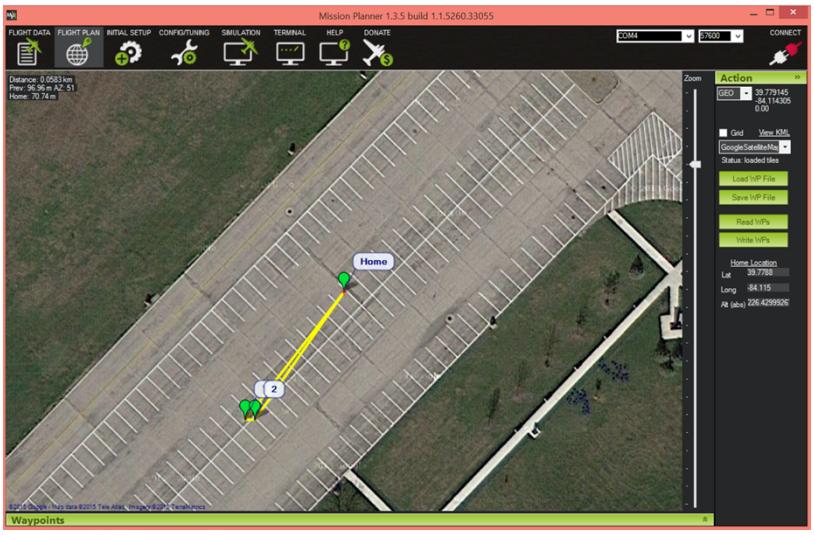
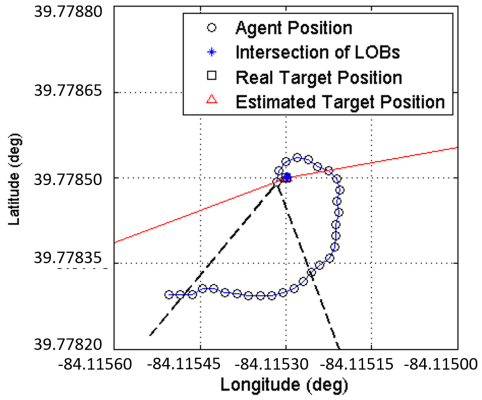
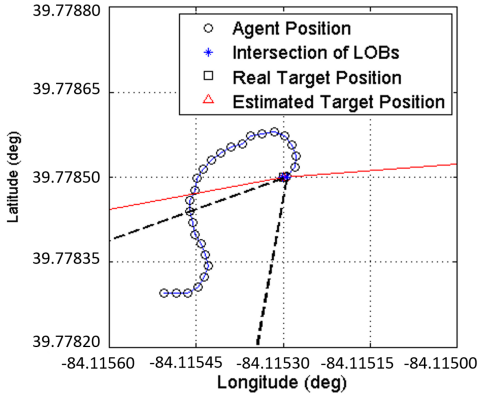


Figure 48. Arrangement of Target & Agent

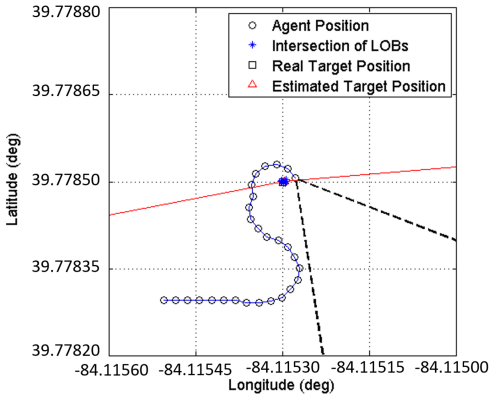
In the real test, the target is in the position where the longitude is -84.1150000° and the latitude is 39.7788000°. Considering the agent's movement, the truck is in the position where the longitude is -84.1152050° and the latitude is 39.7785950° and it moves in longitude direction. It is shown in Figure 48. ε is set to 10° and satisfactory semi-major axis of ellipse is $40m$. The speed of the agent is assumed as 0.66 ft and $\triangle t$ is 10 sec. The speed of the agent is constrained by the small space of the parking lot. If the real speed of the air frame was used, it is hard to control the movement of the agent. So, this speed is determined as a good value for the real test. In practice, it would have been better to apply scaling laws to ensure similarity between ground vehicle and flight vehicle testing.



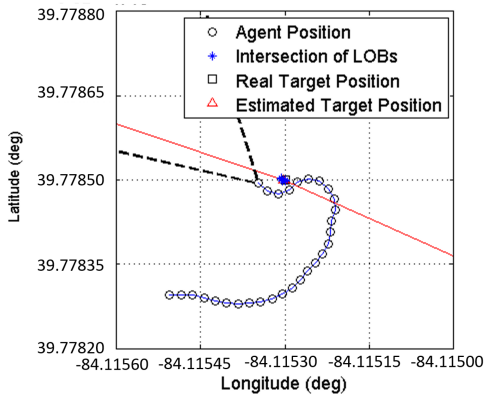
(a) 1st Type



(b) 2nd Type



(c) 3rd Type



(d) 4th Type

Figure 49. Optimal Flight Path & Error Ellipse

The second position is the position where longitude is -84.1151845° and latitude is 39.7785950° for making intersections. The agent can get its initial correct ellipse on the third position where longitude is -84.1151640° and latitude is 39.7785950°. The measurement error can effect the flight path propagation. So, several different types of flight paths are generated and they are shown in Figure 49. Every type of path is around the target position and it shows roughly a spiral shape.

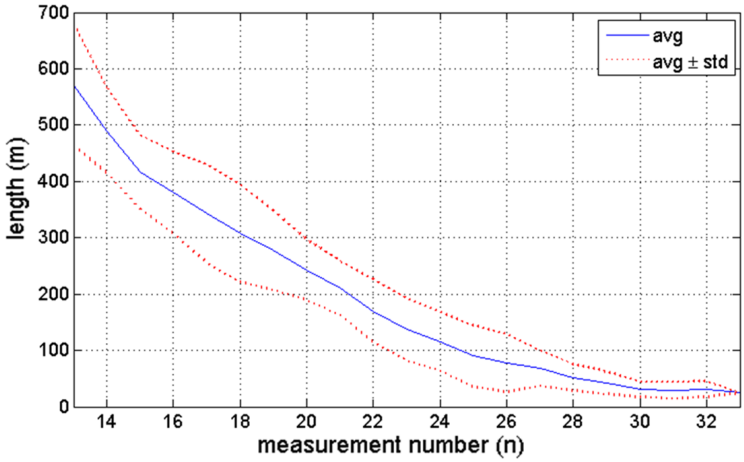


Figure 50. Semi-Major Axis of Ellipse for Simulation

The change of semi-major axis on flight path is shown in Figure 50. This result is taken by 100 different simulations. In the graph, the blue line means the average value of semi-major axis on each measurement number. The fact that the semi-major axis is reduced with time is very definitely shown in Figure 50.

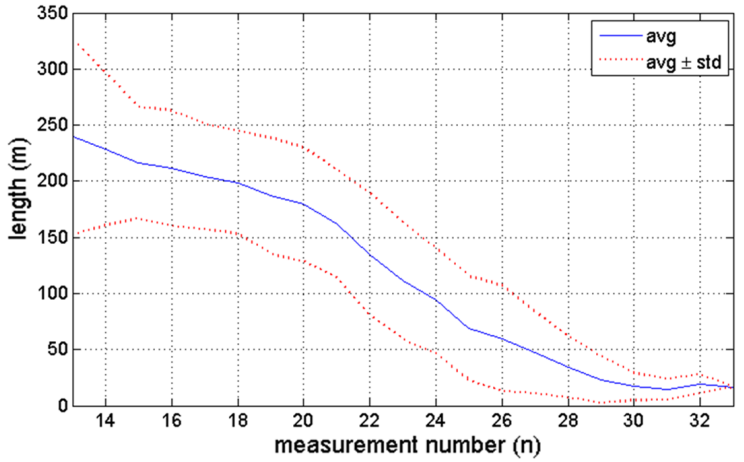


Figure 51. Distance (Real - Estimated TGT) for Simulation

Figure 51 expresses the distance between the real target and the estimated target. This graph is taken the same way as Figure 50 and its trend is very similar to the semi-major axis of ellipse.

Table 3. Percentage of Measurement (100 simulations)

Measurement (n)	Percentage (%)	Measurement (n)	Percentage (%)
22	1	28	13
23	7	29	9
24	4	30	8
25	21	31	2
26	22	32	1
27	11	33	1

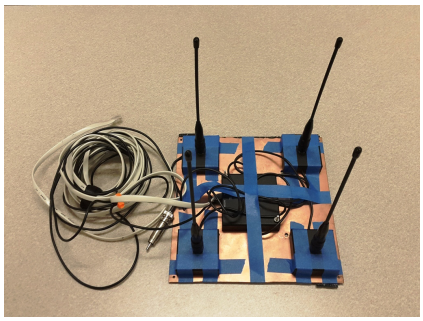
As described before, the simulation is run 100 times. Every simulation has the different measurement number for getting satisfactory semi-major axis. Table 3 shows how many times a certain measurement number is needed for getting the predefined semi-major axis among 100 runs simulation. So, the different measurement errors bring up the different shape of flight path and measurement number.

4.3 Implementation

The LOBs are measured by radio signal. The radio emitter and radio direction finder take significant roles in the real test. The radio emitter is assumed to be a target and the radio direction finder is mounted on the remotely controlled truck as in Figure 52d. A radio direction finder measures the LOBs and transfers data to the computer.

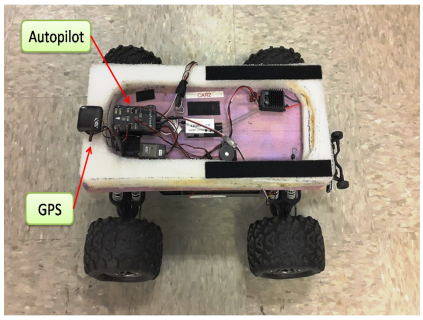
A remote controlled truck is needed for moving to the correct position in latitude and longitude, automatically. So, it uses an autopilot and GPS for doing this function. The autopilot makes it move to the point where the program commanded and the GPS supplies the current position information to the autopilot.

MATLAB calculates the next optimal position with the LOBs and the optimal position information is input to the 'Mission Planner' software. The software interface



(a) Radio Direction Finder

(b) Radio Emitter



(c) Remote Controlled Truck

(d) Truck Assembly

Figure 52. Real Test Equipment

is shown in Figure 48. The 'Mission Planner' communicates with the truck and controls the truck position using the GPS and autopilot.

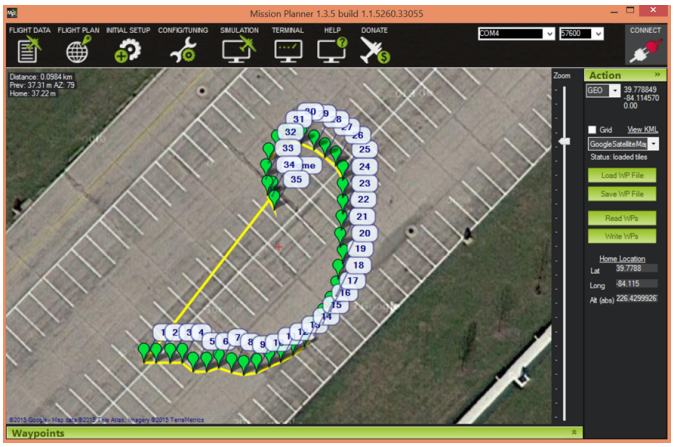
The truck assembly is provided by the ANT lab at AFIT and had been used for other research. Mounting the radio direction finder and adjusting the proper speed of truck were needed for the original configuration. It doesn't take much time to change its configuration for this real test.

As described before, the radio emitter serves the role of the target. So, radio emitter is put in the position where longitude is -84.1151640° and latitude is 39.7785950°. The truck moves in a straight path until a proper ellipse is formed from the position where -84.1152050° and latitude is 39.7785950°. The speed of agent is assumed as 0.66 $ft/$s and $\triangle t$ is 10 sec. After forming a proper ellipse, it moves to an optimal position and measures the LOB using radio direction finder. With these measurements, MATLAB calculates error ellipse and the next optimal position. When the semi-major axis of error ellipse meets the predefined number, the truck stops its movement.

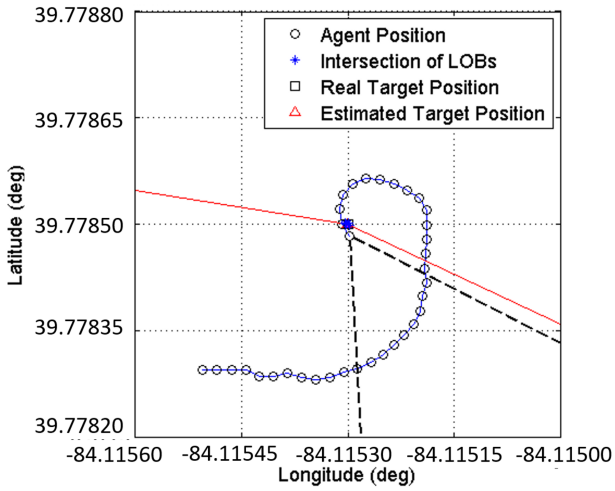
4.4 Test Result & Analysis

4.4.1 Flight Path & Semi-Major Axis.

The result of real test produced an optimized path as shown in Figure 53. Its contour is similar with Figure 49a. This type of contour is very general for a single target problem. The curve of the path is very smooth and truck shows a spiral movement to the radio emitter as expected.

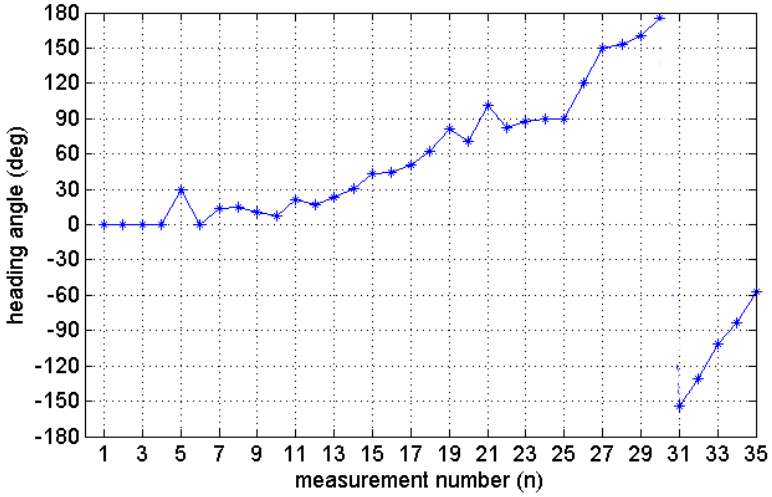


(a) Mission Planner Interface

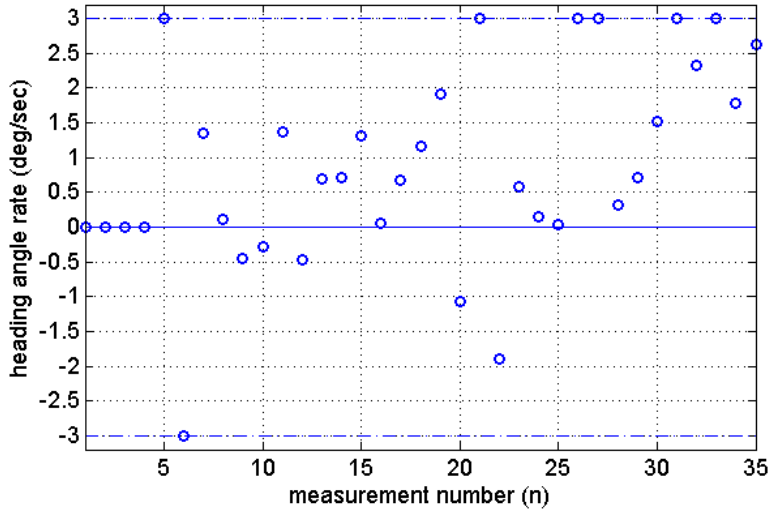


(b) MATLAB Graph

Figure 53. Optimized Flight Path



(a) Heading Angle



(b) Heading Angle Rate

Figure 54. Heading Angle and Rate for Real-World Test

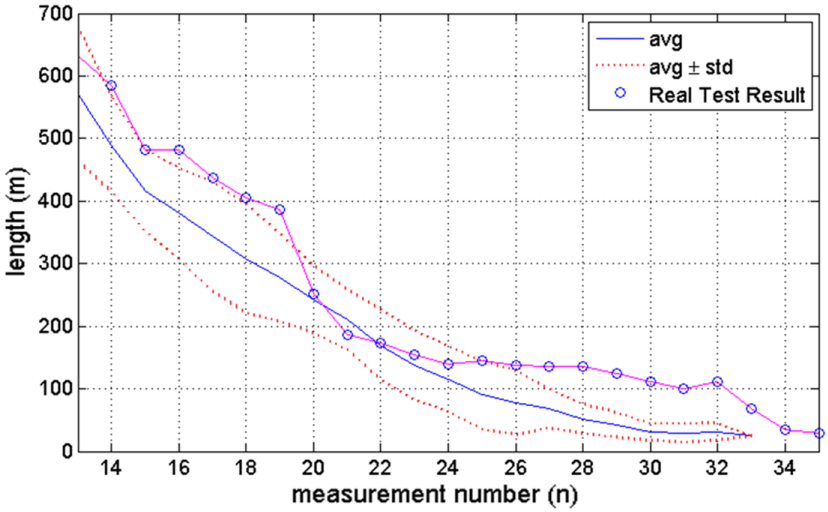


Figure 55. Semi-Major Axis of Ellipse for Real-World Test

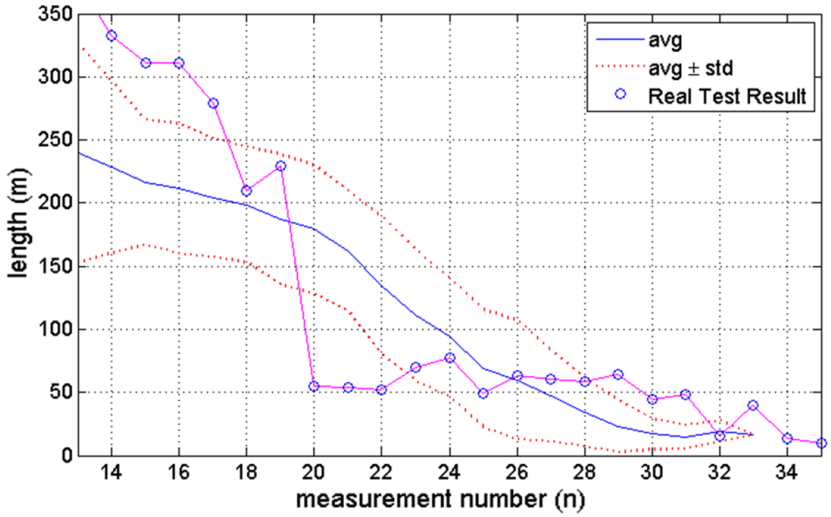


Figure 56. Distance (Real - Estimated TGT) for Real-World Test

The semi-major axis values of the real test are very similar with the simulation results. In Figure 55, the line of test is almost inside of simulation result range even though the last part is not. In addition, the distance between real target and estimated target is normally inside of simulation range or lower in value as in Figure

56. Similarly with the semi-major axis tendency, the graph of distance is extended further than simulation result.

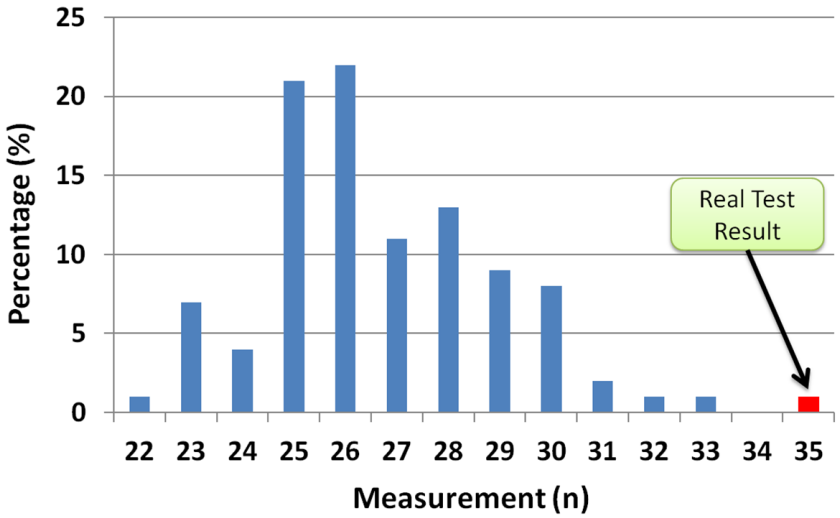


Figure 57. Percentage of Measurement Time (100 simulations)

This extended graph means that the real test takes more time to meet the predefined semi-major axis value. Figure 57 shows how many measurements are needed to meet the predefined number. To get the percentage value, the simulation was run 100 times. The minimum measurement number is 22 and the maximum measurement number is 33. The most frequent measurement number is 25 and 26. So, the result of real test, 35 measurements, is a quite a bit larger number than the statistical result. However, this can be due to the assumed vs actual value of the measurement error.

4.4.2 Scenario Improvement.

As demonstrated in simulation and in a real test, the agent flies into a real target position to minimize the error ellipse. However, it can be dangerous considering enemy's attack systems in real warfare. Normally, a radio emitter is used in RADAR or Missile defense facilities. So, if the agent approaches the real target position, it is threatened by the enemy's attack weapon. As a result, a flight prohibited area around an estimated target is needed to enhance its survivability.[12] To achieve this,

a constraint in the optimal control problem is required to keep the agent out of the threat zone and is defined as

$$\mid P_A - \hat{P}_T \mid \geq \ R_{TGT} \tag{4.1}$$

where R_{TGT} is the radius of the flight prohibited area.

Furthermore, the agent may consider obstacles like mountains, hills, trees or buildings. Increasing the altitude of the agent is the easiest way to escape these obstacles. However, increasing the altitude effects the accuracy of the radio measurement. So, the agent needs an avoidance process for evading the danger of crashing into these obstacles. This avoidance process can be helpful to evade the identified enemy's weapon facility around a target. Information about geographical obstacles can be achieved by using a satellite map. So, each identified obstacle is considered in this section for the avoidance process.

The avoidance process can be achieved by adding the inequality constraint to the optimization problem as previously defined. This inequality constraint is defined as

$$\delta - \mid \psi - \phi \mid \leq \ \triangle t \cdot \dot{\psi}_{max} \tag{4.2}$$

where R_{obs} is a radius of flight prohibited area around a obstacle, ρ_{obs} represents the distance between the agent and the obstacle and δ is the avoidance angle as shown in Figure 58. The avoidance angle can be calculated by using R_{obs} and ρ_{obs}. If the sum of the maximum change in heading angle and the absolute value of the heading angle subject to ρ_{obs} is bigger than the avoidance angle, the agent can evade the obstacle.

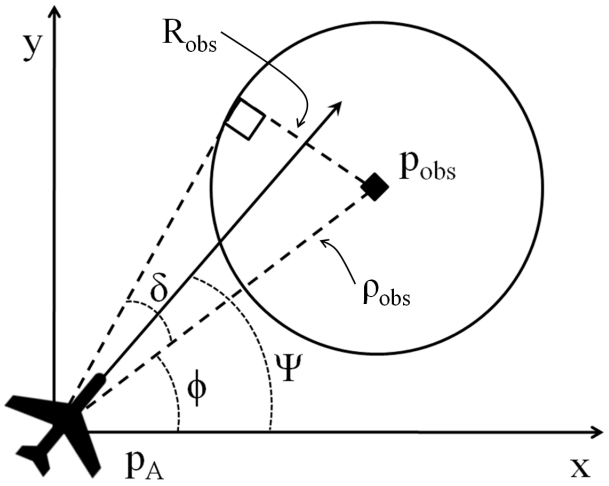


Figure 58. Obstacle Avoidance

As a result, the optimal control problem is described as

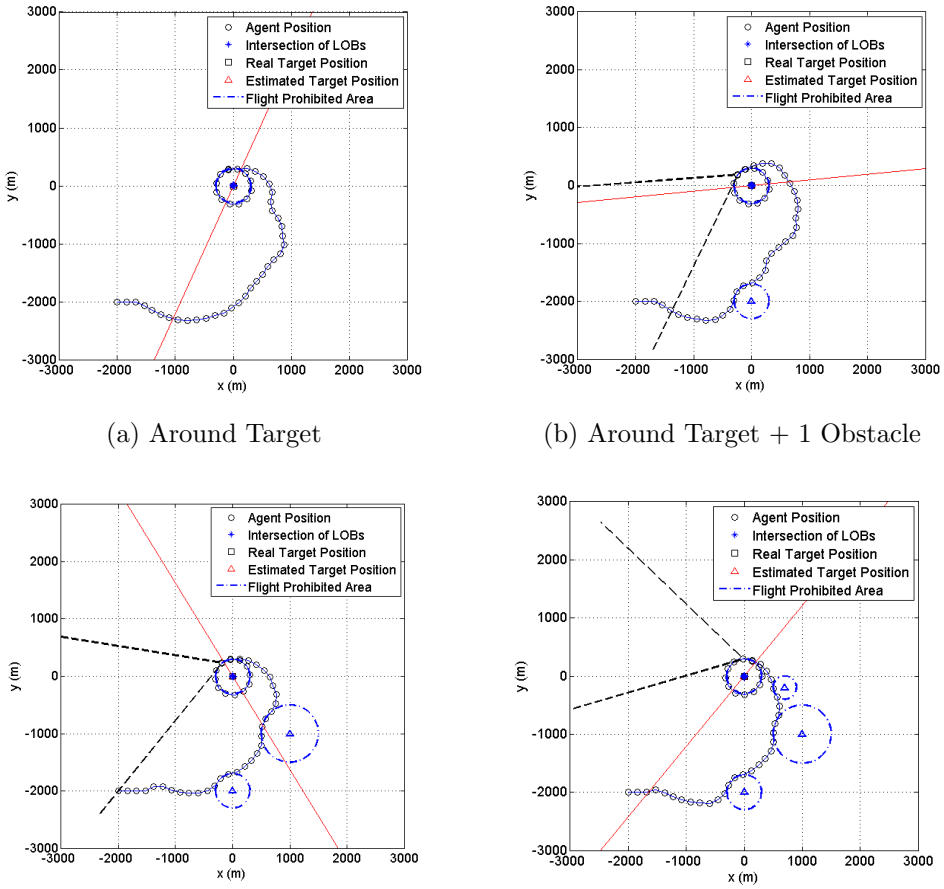
$$J = max\,[\,\sqrt{\lambda\,(P)}\,] \tag{4.3}$$

subject to

$$\begin{aligned} x &= x_0 + V \cdot \triangle t \cdot cos(\psi) \\ y &= y_0 + V \cdot \triangle t \cdot sin(\psi) \\ R_{TGT} &\leq |\, P_A - \hat{P}_T \,| \\ \dot{\psi} &\leq 3^\circ/sec \\ \delta - |\, \psi - \phi \,| &\leq \triangle t \cdot \dot{\psi}_{max} \end{aligned} \tag{4.4}$$

The result of the simulation including the new constraints is shown in Figure 59a. Its R_{TGT} value is 300 m and the agent doesn't fly inside of flight prohibited area. The results of setting flight prohibited area around obstacles are shown in Figure 59b, 59c and 59d. The first obstacle is on (0, -2000) and its R_{TGT} is 300 m. The

second obstacle is on (1000, -1000) and its R_{TGT} is 500 m. The last one is on (700, -200) and its R_{TGT} is 200 m.



(a) Around Target

(b) Around Target + 1 Obstacle

(c) Around Target + 2 Obstacles

(d) Around Target + 3 Obstacles

Figure 59. Flight Prohibited Area

The significant feature of the optimal path including a flight prohibited area is that the agent choose the path near the target but around the flight prohibited area. When the agent approaches the flight prohibited area, it has two ways to avoid flight into the prohibited area. One option is nearer to the target and the other is farther away. However, it chooses the way nearer to the target. It is very advantageous with respect to radio signal receiver. If the agent flies farther way, the obstacle hinders the radio reception. This result is considered very natural because the nearer position to the target is more advantageous to minimize error ellipse.

4.5 Summary

In this chapter, simulation based on situation based on real-world test was conducted and real-world test was conducted using RC truck. The results of simulation and real-world test were compared each other.

The results of real-world test are very similar with them of simulation. The semi-major axis and distance between real target and estimated target of real-world test are almost inside of simulation result range. However, more measurements were needed for the predefined semi-major axis. In addition, the algorithm for evading the flight prohibited area is added using inequality constraints. With this algorithm, the agent approaches nearer way to the target around the flight prohibited area. This can be advantageous with respect to radio transmittnace.

V. Conclusions and Recommendations

5.1 Conclusions

The goal of this research was to minimize uncertainty in the target position in minimal time. Minimizing uncertainty is achieved by using an optimal problem solution. Uncertainty of the target position was assumed as the shape of the ellipse that is generated by LOBs measurement error. LOB measurement values and a prescribed error of the equipment produces an area which suggest a possible region of target. More than 2 LOBs make intersection and can be reduced adding LOBs information from different places. So, LOBs adding error information can shape the surface of a possible target position, geometrically. For doing this, a semi-major axis of ellipse is used as a cost function and it provides an a optimal position for minimizing the semi-major axis of the error ellipse.

The algorithm suggested in this research propagates this sequence until the semi-major axis meets a satisfactory value. Finally, it can reduce the error ellipse and estimate the target position precisely. This algorithm was successfully demonstrated in Chapter IV. The semi-major axis value is continuously decreased with flightpath and the distance between the real target and the estimated target is reduced. Eventually, the decrease of the semi-major axis is bounded because of measurement error. The flightpath shows the spiral path to the real target position. This result corresponds with the approximation in the beginning of this research.

An algorithm for multiple targets is presented, briefly. Further research is needed for multiple targets. In addition, implementation of this algorithm onto a UAV should be demonstrated. Disturbances associated with real UAV flight conditions and their effects on the algorithm's performance should be investigated. So, additional research is needed for adoption to real world applications.

5.2 Recommendations for Future Research

5.2.1 Real Flight Test.

A real flight test is required to fully validate the algorithm. The result of an application to a truck is very restricted. Specifically, the truck moves in 2D space, whereas the airplane moves in 3D space. It affects the overall algorithm profoundly. The 3D circumstance needs to address LOBs measurement and flightpath including altitude of aircraft. As a result, incorporation of 3D is needed for the overall algorithm. This will require more steps in the algorithm to calculate the optimal path.

For the real flight test, more stable equipment is needed. The UAV creates vibration and effects the error on the devices. These conditions adversely effect the device's performance. Sometimes, the devices used in this research showed several malfunctions. A real flight test with stable devices could validate the targeting algorithm.

Furthermore, a transition to real software for the aircraft(not MATLAB) is required when this algorithm is used for a real UAV. By doing that, it can reduce the required hardware processor and make the system more efficient.

5.2.2 Multiple Target Algorithm.

The algorithm used in this research for multiple target minimized the error ellipse of the nearer target and then transited to minimizing for the next target. However, this algorithm doesn't guarantee minimal time for targeting. The optimal direction to the first target can adversely effect the next target position. The initial flightpath can create a long distance to the second target. So, the direction of flightpath needs to consider the next target position. As a result, the optimal decision of flightpath direction has to be done for minimal time.

In addition, this algorithm is not guaranteed as the best solution for minimizing time for targeting. So, research into for minimizing the error ellipse summation or any other cost function is needed for validation. Even though the algorithm using

transition to error ellipse used in this research works well, the comparison with other algorithms using other cost function is needed for validation and comparison to true optimal.

5.2.3 Global minimum Solution.

Finding the global minimum condition requires computation time. So, it is not suitalbe for real-time control problem, especially where aircraft cannot wait for the next position to move because it threatens the safety of the aircraft. This is the reason why this thesis used a local minimum. However, the simulation result shows that the semi-major axis is increased sometimes. Even though it is decreased after all, this needs to be prevented.

There are deterministic methods and stochastic methods for global optimization. So, there are several ways to find a global minimum. But, the important thing is that it must be suitable for real-time control.

5.2.4 Radio Direction Finding Device Validation.

This research required a radio direction finding device that the measurement error is less than some certain value. This error is directly related with accuracy and time to achieve a satisfactory semi-major axis. So, the correct maximum error associated with the hardware used needs to be known for this algorithm.

This value can be measured by real field test. Many experiments with radio emitter and radio direction finder can give approximated value. However, the barriers like buildings and trees effect the experiment results. So, many variables needs to be considered for measurement.

For the purpose of the real flight test experiment in an open field, the radio direction finding measurement experiment for multiple tests in an open field can give maximum error value and this value can be validated. However, the usage of this algorithm in real-world needs higher accuracy device than the device used for the

experiments herein. Additional low cost radio direction finding equipment suitable for use on a UAV should be explored and tested.

Appendix A. Data Tables

Table 4. Key Flight Parameters of SigRascal110 [4]

Parameter	Unit	Value
Wing Span	feet	8.8
Length	feet	6.1
Propulsion	feet	Electric
MTOW	pounds	16
Payload Weight	pounds	11
Payload Volume	cu-in	756
Criuse Velocity	knots	28-40

Table 5. Real-world Test Result

No.	Lat(°)	Lon(°)	ψ(°)	β(°)	No.	Lat(°)	Lon(°)	ψ(°)	β(°)
1	3.9778595	-8.4115205	0.0	53	19	3.9778698	-8.4114895	81.8	128
2	3.9778595	-8.4115185	0.0	56	20	3.9778717	-8.4114888	71.1	131
3	3.9778595	-8.4115164	0.0	52	21	3.9778738	-8.4114892	101.1	145
4	3.9778595	-8.4115143	0.0	55	22	3.9778758	-8.4114890	82.3	154
5	3.9778585	-8.4115126	30.3	61	23	3.9778778	-8.4114889	88.1	162
6	3.9778585	-8.4115105	0.0	65	24	3.9778799	-8.4114889	89.6	163
7	3.9778590	-8.4115085	13.6	70	25	3.9778819	-8.4114889	90.0	180
8	3.9778584	-8.4115066	14.8	71	26	3.9778837	-8.4114899	120.0	-167
9	3.9778581	-8.4115045	10.3	82	27	3.9778847	-8.4114917	150.0	-158
10	3.9778583	-8.4115025	7.4	87	28	3.9778857	-8.4114935	153.2	-150
11	3.9778590	-8.4115006	21.1	83	29	3.9778864	-8.4114954	160.3	-140
12	3.9778597	-8.4114986	16.5	95	30	3.9778865	-8.4114975	175.4	-128
13	3.9778605	-8.4114967	23.4	102	31	3.9778856	-8.4114993	-154.6	-106
14	3.9778615	-8.4114950	30.5	100	32	3.9778840	-8.4115007	-131.2	-91
15	3.9778629	-8.4114935	43.6	111	33	3.9778820	-8.4115011	-101.2	-86
16	3.9778644	-8.4114920	44.2	114	34	3.9778800	-8.4115008	-83.4	1
17	3.9778660	-8.4114907	50.9	120	35	3.9778783	-8.4114997	-57.0	99.4
18	3.9778678	-8.4114898	62.7	128					

Appendix B. MATLAB simulation code

2.1 Sample_Result.m

```
1  % ------------------------------------------------------------
2  % Sample_Result
3  % ------------------------------------------------------------
4  clear all;close all;clc;
5
6  sigma_beta=deg2rad(1);
7
8  %% Max error & Predefined semi-major axis
9  err_const = 3 ;
10 radius_limit = 5;
11
12 %target position:
13 tx_true=0;
14 ty_true=0;
15
16 % Initial path
17 x_ini=[-1000;-840];
18 y_ini=[-1000;-1000];
19
20 X=x_ini;
21 Y=y_ini;
22 pos_X= X;
23 pos_Y=Y;
24
25 % LOB Measurement
26 beta = atan2(ty_true-Y,tx_true-X)+sigma_beta*(rand(1));
```

```
27
28 psi = [0;atan2(Y(2)-Y(1),X(2)-X(1))];
29 delpsi=[0;0];
30
31 err=sigma_beta*err_const;
32 [a1,b1] = size(beta);
33 for loop1=1:a1
34     if beta(loop1,1) < err
35         beta(loop1,1)= beta(loop1,1)+2*pi;
36     else
37     end
38 end
39
40 beta_err(:,1) = [beta-err];
41 beta_err(:,2) = [beta+err];
42
43 line = sqrt((X(1)-tx_true)^2+(Y(1)-ty_true)^2)*1.2;
44
45 figure(1)
46 [X,Y,beta,beta_err,inter,TGT,LongRadius,EllipseSize] =
     pathmaker(X,Y,beta,beta_err,err,tx_true,ty_true,sigma_beta
     );
47 hold on
48 plot(X,Y,X,Y,'ko')
49 hold on
50 h(1)=plot(X,Y,'ko');
51 TGTn=TGT(length(TGT),:);
52 h(2)=plot(TGTn(1,1),TGTn(1,2), 'r^');
53 h(4)=plot(tx_true,ty_true, 'ks');
```

```
54 h(5)=plot(inter(:,1),inter(:,2),'*')    ;
55
56 legend(h([1 5 4 2]),'Agent Position', 'Intersection of LOBs',
       'Real Target Position','Estimated Target Position');
57 ax = gca;
58 XTick = [-1200:400:1000];
59 YTick =  XTick;
60 set(gca,'XTick',XTick,'YTick',YTick,'fontsize',13)
61 xlabel('x');ylabel('y');
62 axis([-1300,1000,-1300,1000])
63 axis square
64 hold off;
65
66 inter=[];
67
68 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
69 %% Iteration Option
70 tf=10;
71 V=16;
72 N=1;
73
74 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
75 %% Iteration Start
76 len=length(X);
77 x_update = X(len);
78 y_update = Y(len);
79 psi_update =0;
80 x_path(:,1)=X;
81 x_path(:,2)=Y;
```

```
x_path(:,3)=0;
itn = 1;

while LongRadius(length(LongRadius)) > radius_limit
psi_init = psi_update;
x_init = x_update+(tf/N)*V*cos(deg2rad(psi_init+30));
y_init = y_update+(tf/N)*V*sin(deg2rad(psi_init+30));
psi_prev = 0;
x0=[x_init y_init psi_init+30];
s0=[x_update y_update psi_update];

A=[];
b=[];
Aeq=[];
beq=[];
lb=[];
ub=[];

options=optimset('Algorithm','sqp','Display','Iter','
    MaxFunEvals',1e10,'MaxIter',1e2);
% ------------------------------------------------------
% Call fmincon using the following syntax
% [x, fval, exitflag, output, lambda, grad, hessian] =fmincon
    (@myfun, x0, [], [], [], [], lb, ub, @mynonlcon, options)
% ------------------------------------------------------
[x_star,fval,exitflag,output,lambda] = fmincon(@(x0)Eval_th(
    x0,sigma_beta,beta,err,tx_true,ty_true,N,len,x_path),x0,A,
    b,Aeq,beq,lb,ub,@(x0)cons_th(x0,s0,N,tf,V),options);
```

```
106 [f,beta,err,beta_err,TGTn,ell_x,ell_y,inter_sel]=Eval_th(
      x_star,sigma_beta,beta,err,tx_true,ty_true,N,len,x_path);
107 [~,~,psi_dot_star]=cons_th(x_star,s0,N,tf,V);
108
109 TGTa=TGT(len+itn-1,:);
110
111     if (s0(1)-TGTa(1))^2+(s0(2)-TGTa(2))^2 < (x_star(1)-TGTa
          (1))^2+(x_star(2)-TGTa(2))^2
112         options=optimset('Algorithm','sqp','Display','Iter','
              MaxFunEvals',1e10,'MaxIter',1e2);
113         [x_star,fval,exitflag,output,lambda] = fmincon(@(x0)
              Eval_th2(x0,N,TGTa),x0,A,b,Aeq,beq,lb,ub,@(x0)
              cons_th(x0,s0,N,tf,V),options);
114         [dis]=Eval_th2(x_star,N,TGTa);
115         [~,~,psi_dot_star]=cons_th(x_star,s0,N,tf,V);
116
117         X=[x_path(:,1);x_star(1)];
118         Y=[x_path(:,2);x_star(2)];
119     else
120     end
121
122 x_update = x_star(1);
123 y_update = x_star(2);
124 psi_update = x_star(3);
125
126 % LOB Measurement
127 [x_update,y_update]
128 beta(len+itn) = atan2(ty_true-y_update,tx_true-x_update)+
      sigma_beta*(rand(1));  % Real angle + ERROR
```

```
129
130 [a1,b1] = size(beta);
131     for loop1=1:a1
132         if beta(loop1,1) < err
133             beta(loop1,1)= beta(loop1,1)+2*pi;
134         else
135         end
136     end
137 beta_err(:,1) = [beta-err];
138 beta_err(:,2) = [beta+err];
139 X=[x_path(:,1);x_star(1)];
140 Y=[x_path(:,2);x_star(2)];
141 [inter_sel,ell_x,ell_y,TGTn,LongRadiusn] = Ellipse(X,Y,
      beta_err,err);
142
143 LongRadius(len+itn)=LongRadiusn;
144 TGT(len+itn,:)=TGTn;
145 x_path(len+itn,1:3) = x_star;
146 objective_func(itn)=fval;
147
148 figure(1);
149 plot(x_path(:,1),x_path(:,2),x_path(:,1),x_path(:,2),'ko')
150 hold on
151 h(1)=plot(x_path(1,1),x_path(1,2),'ko');
152 h(2)=plot(TGTn(1,1),TGTn(1,2), 'r^');
153 h(3)=plot(TGTn(1)+ell_x,TGTn(2)+ell_y,'m','linewidth',2);
154 h(4)=plot(tx_true,ty_true, 'ks');
155 h(5)=plot(inter_sel(:,1),inter_sel(:,2),'*');
156 p=sqrt((ell_x).^2+(ell_y).^2);
```

```
157 pp=find(p==max(p));
158 ppp= [TGTn(1,1)+1000*(ell_x(pp)),TGT(1,2)+1000*(ell_y(pp))];
159 h(6)=plot(TGTn(1,1),TGTn(1,2), 'r*');
160 hold on
161 grid on;
162 h(6)=plot([-ppp(1,1),TGTn(1,1),ppp(1,1)],[-ppp(1,2),TGTn(1,2)
      ,ppp(1,2)], 'r');
163 h(7)=plot(TGTn(1)+ell_x,TGTn(2)+ell_y,'m','linewidth',2);
164 h(8)=plot(inter_sel(:,1),inter_sel(:,2),'*');
165 aa=length(X);
166 h(9)=plot([X(aa), X(aa)+line*cos(deg2rad(x_star(1,3)+30))],[Y
      (aa), Y(aa)+line*sin(deg2rad(x_star(1,3)+30))],'--k','
      linewidth',1.5);
167 h(10)=plot([X(aa), X(aa)+line*cos(deg2rad(x_star(1,3)-30))],[
      Y(aa), Y(aa)+line*sin(deg2rad(x_star(1,3)-30))],'--k','
      linewidth',1.5);
168 legend(h([1 5 4 2]),'Agent Position', 'Intersection of LOBs',
      'Real Target Position','Estimated Target Position');
169 ax = gca;
170 XTick = [-1200:400:1000];
171 YTick = XTick;
172 set(gca,'XTick',XTick,'YTick',YTick,'fontsize',13)
173 xlabel('x');ylabel('y');
174 axis([-1300,1000,-1300,1000])
175 axis square
176 hold off;
177
178 %% Drawing Ellipse
179 figure(2)
```

```
plot(x_path(:,1),x_path(:,2),x_path(:,1),x_path(:,2),'ko')
hold on
h(1)=plot(x_path(1,1),x_path(1,2),'ko');
h(2)=plot(TGTn(1,1),TGTn(1,2), 'r^');
h(3)=plot(TGTn(1)+ell_x,TGTn(2)+ell_y,'m','linewidth',2);
h(4)=plot(tx_true,ty_true, 'ks');
h(5)=plot(inter_sel(:,1),inter_sel(:,2),'*')    ;
p=sqrt((TGTn(1,1)-ell_x).^2+(TGTn(1,2)-ell_y).^2);
pp=find(p==max(p));
ppp= [TGTn(1,1)+1000*(ell_x(pp)-TGTn(1,1)),TGT(1,2)+1000*(
    ell_y(pp)-TGTn(1,1))];
grid on;
h(6)=plot([-ppp(1,1),TGTn(1,1),ppp(1,1)],[-ppp(1,2),TGTn(1,2)
    ,ppp(1,2)],'r');
h(7)=plot(TGTn(1)+ell_x,TGTn(2)+ell_y,'m','linewidth',2);
h(8)=plot(inter_sel(:,1),inter_sel(:,2),'*')    ;

aa=length(X);
legend(h([1 5 4 2]),'Agent Position', 'Intersection of LOBs',
    'Real Target Position','Estimated Target Position');
ax = gca;
XTick = [-10:2:10];
YTick = XTick;
set(gca,'XTick',XTick,'YTick',YTick,'fontsize',13)
xlabel('x');ylabel('y');
axis([-10,10,-8,12])
axis square
hold off;

```

```
206 itn= itn+1;
207 end
208
209 distance = sqrt((TGT(:,1)-tx_true).^2+ (TGT(:,2)-ty_true).^2)
210
211 figure(3)
212 plot(LongRadius); grid on;
213 axis([10,45,0,105])
214 ax = gca;
215 XTick = [10:5:45];
216 YTick = [0:10:100];
217 set(gca,'XTick',XTick,'YTick',YTick,'fontsize',13)
218 xlabel('measurement number (n)');ylabel('length (m)');
219
220 figure(4)
221 plot(distance); grid on;
222 axis([10,45,0,22])
223 ax = gca;
224 XTick = [10:5:45];
225 YTick = [0:5:30];
226 set(gca,'XTick',XTick,'YTick',YTick,'fontsize',13)
227 xlabel('measurement number (n)');ylabel('length (m)');
```

2.2 Ellipse.m

```
1 function [inter_sel,ell_x,ell_y,TGT,LongRadius] = Ellipse(x,y
    ,beta_err,err)
2
3 len = length(x);
```

```
4  inter=[];
5  [row1,col1]=size(inter);
6
7  for m = 1:(len-1)
8          for n = 1:(len-m)
9          for n2 = 1:2
10             if  abs(tan(beta_err(m,1))-tan(beta_err(len+1-n,
                   n2))) < 10^(-2)
11                 inter(row1+1,1) = ((y(len+1-n,1)-y(m,1)) + (x
                       (m,1)*tan(beta_err(m,2))-x(len+1-n,1)*tan(
                       beta_err(len+1-n,n2))))/...
12                                 (tan(beta_err(m,2))-tan(
                                       beta_err(len+1-n,n2)));
13                 inter(row1+1,2) = (inter(row1+1,1)-x(m,1))*
                       tan(beta_err(m,2))+y(m,1);
14                 if n==1
15                     inter(row1+1,4) = 1;
16                         inter(row1+1,5) = beta_err(m,2);
17                         inter(row1+1,6) = beta_err(len,n2);
18                 else
19                 end
20                 [row1,col1]=size(inter);
21
22         elseif abs(tan(beta_err(m,2))-tan(beta_err(len+1-n,n2
               ))) < 10^(-2)
23                 inter(row1+1,1) = ((y(len+1-n,1)-y(m,1)) + (x
                       (m,1)*tan(beta_err(m,1))-x(len+1-n,1)*tan(
                       beta_err(len+1-n,n2))))/...
```

```
24                                  (tan(beta_err(m,1))-tan(
                                        beta_err(len+1-n,n2)));
25              inter(row1+1,2) = (inter(row1+1,1)-x(m,1))*
                    tan(beta_err(m,1))+y(m,1);
26              if n==1
27                  inter(row1+1,4) = 1;
28                  inter(row1+1,5) = beta_err(m,1);
29                  inter(row1+1,6) = beta_err(len,n2);
30              else
31              end
32          [row1,col]=size(inter);
33          else
34              inter(row1+1,1) = ((y(len+1-n,1)-y(m,1)) + (x
                    (m,1)*tan(beta_err(m,1))-x(len+1-n,1)*tan(
                    beta_err(len+1-n,n2)))) /...
35                                  (tan(beta_err(m,1))-tan(
                                        beta_err(len+1-n,n2)))
                                        ;
36              inter(row1+1,2) = (inter(row1+1,1)-x(m,1))*
                    tan(beta_err(m,1))+y(m,1);
37              if n==1
38                      inter(row1+1,4) = 1;
39                  inter(row1+1,5) = beta_err(m,1);
40                      inter(row1+1,6) = beta_err(len,n2);
41              else
42              end
43              [row1,col1]=size(inter);
```

```
44                  inter(row1+1,1) = ((y(len+1-n,1)-y(m,1)) + (x
                        (m,1)*tan(beta_err(m,2))-x(len+1-n,1)*tan(
                        beta_err(len+1-n,n2)))) /...
45                                      (tan(beta_err(m,2))-tan(
                                            beta_err(len+1-n,n2)));
46                  inter(row1+1,2) = (inter(row1+1,1)-x(m,1))*
                        tan(beta_err(m,2))+y(m,1);
47                  if n==1
48                          inter(row1+1,4) = 1;
49                          inter(row1+1,5) = beta_err(m,2);
50                       inter(row1+1,6) = beta_err(len,n2);
51                  else
52                  end
53                  [row1,col1]=size(inter);
54              end
55          end
56      end
57  end
58
59  %% intersection selection
60  [r,c]=size(inter);
61  for   q1=1:r
62          for q2 = 1:len
63          slide = atan2(inter(q1,2)-y(q2), inter(q1,1)-x(q2)) ;
64
65          if (slide-beta_err(q2,1))*(slide-beta_err(q2,2)) <=
                0.0000001
66                  inter(q1,3)=inter(q1,3)+1;
67          else
```

```
68          end
69          if slide <= 2*err
70              slide=slide+2*pi;
71                  if (slide-beta_err(q2,1))*(slide-beta_err(q2
                        ,2)) <= 0.0000001
72                  inter(q1,3)=inter(q1,3)+1;
73              else
74              end
75          else
76          end
77          end
78  end
79
80  [a1,b1] = size(inter);
81  inter_length=a1;
82  inter_sel = [];
83  [r1,c1]=size(inter_sel);
84  for u=1:a1
85          if inter(u,3) == len
86          inter_data(r1+1,:)=inter(u,:);
87          inter_sel(r1+1,:)=inter(u,1:2);
88      else
89      end
90          [r1,c1]=size(inter_sel);
91  end
92
93  if length(inter_sel) ==0
94      inter_sel=[];
95      avg=[0,0];
```

```
96        ell_x=[];
97        ell_y=[];
98       TGT=[];
99        LongRadius=10000;
100  else
101
102  %% TGT position Acquisition
103  avg = [mean(inter_sel(:,1)),mean(inter_sel(:,2))];
104  TGT = avg;
105
106  %% Error Matrix (TGT - Intersection)
107  error=[];
108  for l = 1:r1
109           error(l,:)=[inter_sel(l,1)-avg(1,1),inter_sel(l,2)-
                avg(1,2)];
110  end
111
112  %% Ellipse Size & Long Radius
113  covariance=cov(error);
114  [eigvec,eigval] = eig(covariance);
115  if length(eigval)==1
116       LongRadius=10000;
117  else
118  a=sqrt(eigval(1,1));
119  b=sqrt(eigval(2,2));
120  ang=dcm2angle([eigvec [0;0];0 0 0]);
121  EllipseSize = pi*sqrt(eigval(1,1))*sqrt(eigval(2,2));
122
123  LongRadius=sqrt(max(max(eigval)));
```

```
124 end
125
126 %% Drawing Ellipse
127 ell_n=100;
128 ell_p=0:pi/ell_n:2*pi;
129
130 ell = [cos(ell_p'),sin(ell_p')] * sqrt(eigval) * eigvec';
131 ell_x = ell(:,1);
132 ell_y = ell(:,2);
133
134 end
```

2.3 Pathmaker.m

```
1 function [X,Y,beta,beta_err,inter,TGT,LongRadius,EllipseSize]
      = pathmaker(X,Y,beta,beta_err,err,tx_true,ty_true,
    sigma_beta);
2
3 [a2,b2] = size(X);
4 for z=1:a2
5 %% ---------------------        1st        ---------------------
6     if z==1
7         figure(1)
8         plot(tx_true,ty_true, 'go'); grid on; hold on
9         plot([X(z), X(z)+line*cos(beta_err(z,1))],[Y(z), X(z)
              +line*sin(beta_err(z,1))],'r-')
10        plot([X(z), X(z)+line*cos(beta_err(z,2))],[Y(z), Y(z)
              +line*sin(beta_err(z,2))],'r-')
11        axis square
```

```
12          axis equal
13          hold off
14
15          TGT(1,:) = [X(1),Y(1)];
16          LongRadius(1,1) = [0];
17          EllipseSize(1,1)= [0];
18
19  %% ---------------------          2nd          ---------------------
20      elseif z==2
21          quad_check=0;
22       while quad_check == 0
23          len = length(X);
24          inter=[];
25          [ra,ca]=size(inter);
26          for m = 1:(len-1)
27              for n = 1:(len-m)
28              for n2 = 1:2
29              if  abs(tan(beta_err(m,1))-tan(beta_err(len+1-n,
                    n2))) < 10^(-3)
30                  inter(ra+1,1) = ((Y(len+1-n,1)-Y(m,1)) + (X(m
                        ,1)*tan(beta_err(m,2))-X(len+1-n,1)*tan(
                        beta_err(len+1-n,n2))))/...
31                       (tan(beta_err(m,2))-tan(beta_err(len+1-n,
                            n2)));
32                  inter(ra+1,2) = (inter(ra+1,1)-X(m,1))*tan(
                        beta_err(m,2))+Y(m,1);
33
34                  inter(ra+1,4) = 0;
```

```
35                  if atan2(inter(ra+1,2)-ty_true,inter(ra+1,1)-
                        tx_true) > pi/10;
36                      if atan2(inter(ra+1,2)-ty_true,inter(ra
                            +1,1)-tx_true) < pi/2.2
37                          inter(ra+1,4) = 1;
38                      else
39                      end
40                  else
41                  end
42                  [ra,ca]=size(inter);
43                  elseif abs(tan(beta_err(m,2))-tan(beta_err(
                        len+1-n,n2))) < 10^(-3)
44                  inter(ra+1,1) = ((Y(len+1-n,1)-Y(m,1)) + (X(m
                        ,1)*tan(beta_err(m,1))-X(len+1-n,1)*tan(
                        beta_err(len+1-n,n2)))) /...
45                      (tan(beta_err(m,1))-tan(beta_err(len+1-n,
                            n2)));
46
47                  inter(ra+1,2) = (inter(ra+1,1)-X(m,1))*tan(
                        beta_err(m,1))+Y(m,1);
48
49                  inter(ra+1,4) = 0;
50                  if atan2(inter(ra+1,2)-ty_true,inter(ra+1,1)-
                        tx_true) > pi/10;
51                      if atan2(inter(ra+1,2)-ty_true,inter(ra
                            +1,1)-tx_true) < pi/2.2
52                          inter(ra+1,4) = 1;
53                      else
54                      end
```

```
55                  else
56                  end
57                  [ra,ca]=size(inter);
58              else
59                  inter(ra+1,1) = ((Y(len+1-n,1)-Y(m,1)) + (X(m
                        ,1)*tan(beta_err(m,1))-X(len+1-n,1)*tan(
                        beta_err(len+1-n,n2)))) /...
60                  (tan(beta_err(m,1))-tan(beta_err(len+1-n,n2))
                        );
61                  inter(ra+1,2) = (inter(ra+1,1)-X(m,1))*tan(
                        beta_err(m,1))+Y(m,1);
62                  inter(ra+1,4) = 0;
63                  if atan2(inter(ra+1,2)-ty_true,inter(ra+1,1)-
                        tx_true) > pi/10;
64                   if atan2(inter(ra+1,2)-ty_true,inter(ra
                            +1,1)-tx_true) < pi/2.2
65                    inter(ra+1,4) = 1;
66                   else
67                   end
68                  else
69                  end
70                  [ra,ca]=size(inter);
71                  inter(ra+1,1) = ((Y(len+1-n,1)-Y(m,1)) + (X(m
                        ,1)*tan(beta_err(m,2))-X(len+1-n,1)*tan(
                        beta_err(len+1-n,n2)))) /...
72                   (tan(beta_err(m,2))-tan(beta_err(len+1-n,
                            n2)));
73                  inter(ra+1,2) = (inter(ra+1,1)-X(m,1))*tan(
                        beta_err(m,2))+Y(m,1);
```

```
74                      inter(ra+1,4) = 0;
75                      if atan2(inter(ra+1,2)-ty_true,inter(ra+1,1)-
                            tx_true) > pi/10;
76                          if atan2(inter(ra+1,2)-ty_true,inter(ra
                                +1,1)-tx_true) < pi/2.2
77                              inter(ra+1,4) = 1;
78                          else
79                          end
80                      else
81                      end
82                      [ra,ca]=size(inter);
83                  end
84                      end
85                  end
86              end
87
88  %% inter selection
89  [rf,cf]=size(inter);
90  for   qa=1:rf
91              for qb = 1:len
92              slide = atan2(inter(qa,2)-Y(qb), inter(qa,1)-X(qb)) ;
93
94              if (slide-beta_err(qb,1))*(slide-beta_err(qb,2)) <=
                    0.0000001
95                  inter(qa,3)=inter(qa,3)+1;
96              else
97              end
98              if slide <= 2*err
99                  slide=slide+2*pi;
```

```
100                 if (slide-beta_err(qb,1))*(slide-beta_err(qb,2))
                        <= 0.0000001
101                     inter(qa,3)=inter(qa,3)+1;
102                 else
103                 end
104             else
105             end
106             end
107 end
108
109 [rg,cg] = size(inter);
110 inter_length=rg;
111 inter_sel = [];
112 [rh,ch]=size(inter_sel);
113 for pd=1:rg
114             if inter(pd,3) == len
115             inter_data(rh+1,:)=inter(pd,:);
116             inter_sel(rh+1,:)=inter(pd,1:2);
117         else
118         end
119             [rh,ch]=size(inter_sel);
120 end
121
122 %% TGT position Acquisition
123    [rg,cg]=size(X);
124
125       avg = [mean(inter(:,1)),mean(inter(:,2))];
126       TGT(rg,:) = avg;
127
```

```
128 %% Error Matrix (TGT - Intersection)
129 for pc = 1:rh
130     error(pc,:)=[inter_sel(pc,1)-avg(1,1),inter_sel(pc,2)-avg
        (1,2)];
131 end
132
133 %% Ellipse Size & Long Radius
134         covariance=cov(error);
135         [eigvec,eigval] = eig(covariance);
136         a=sqrt(eigval(1,1));
137         b=sqrt(eigval(2,2));
138         ang=dcm2angle([eigvec [0;0];0 0 0]);
139         [rf,cf]=size(X);
140         LongRadius(rf,1)=sqrt(max(max(eigval)));
141         EllipseSize(rf,1) = pi*sqrt(eigval(1,1))*sqrt(eigval
            (2,2));
142
143
144 %% Drawing Ellipse
145         ell_n=100;
146         ell_p=0:pi/ell_n:2*pi;
147
148         ell = [cos(ell_p'),sin(ell_p')] * sqrt(eigval) *
            eigvec';
149         ell_x = ell(:,1);
150         ell_y = ell(:,2);
151
152         figure(1)
153         plot(avg(1)+ell_x,avg(2)+ell_y,'m','linewidth',2);
```

```
154         grid on; hold on
155         plot(tx_true,ty_true, 'ks');
156         plot(avg(1,1),avg(1,2), 'r*')
157
158         plot(inter(:,1),inter(:,2),'*')
159         axn=min(min(ell_x+avg(1,1)),min(ell_y+avg(1,2)));
160         axp=max(max(ell_x+avg(1,1)),max(ell_y+avg(1,2)));
161         hold off
162             quad_sum = sum(inter(:,4));
163
164         if quad_sum >= 1
165             quad_check = 1;
166         else
167             X(len+1) = 2*X(len)-X(len-1);
168             Y(len+1) = 2*Y(len)-Y(len-1);
169             [X(len+1), Y(len+1)]
170             beta(len+1) = atan2(ty_true-Y(len+1),tx_true-X(
                    len+1))+sigma_beta*(rand(1));
171
172             if beta(len+1) < err;
173                 beta(len+1)= beta(len+1)+2*pi;
174             else
175             end
176
177             beta_err(len+1,1) = beta(len+1,1)-err;
178             beta_err(len+1,2) = beta(len+1,1)+err;
179
180             psi(len+1) = atan2(Y(len+1)-Y(len),X(len+1)-X(len
                    ));
```

```
181                 delpsi(len+1) = 0;
182             end
183         end
184     end
185 end
```

2.4 Eval_th.m

```
1  function [f,beta,err,beta_err,TGT,ell_x,ell_y,inter_sel]=
       Eval_th(x,sigma_beta,beta,err,tx_true,ty_true,N,len,x_path
       )
2
3  AA(:,1) = [tan(beta)];
4  AA(:,2) = -ones(length(beta),1);
5  BB = tan(beta).*x_path(:,1)-x_path(:,2);
6
7  aft = (inv(AA'*AA)*AA'*BB)';
8  X=[x_path(:,1);x(N)];
9  Y=[x_path(:,2);x(2*N)];
10
11 beta_new=atan2(aft(2)-x(2*N),aft(1)-x(N));
12 beta = [beta;beta_new];
13
14 [a1,b1] = size(beta);
15 for loop1=1:a1
16     if beta(loop1,1) < err
17         beta(loop1,1)= beta(loop1,1)+2*pi;
18     else
19     end
```

```
20 end
21
22 beta_err(:,1) = [beta-err];
23 beta_err(:,2) = [beta+err];
24
25 [inter_sel,ell_x,ell_y,TGT,LongRadius] = Ellipse(X,Y,beta_err
       ,err);
26
27 f=LongRadius;
```

2.5 Eval_th2.m

```
1 function [dis]=Eval_th2(x,N,TGTa)
2
3 dis=(x(N)-TGTa(1))^2+(x(2*N)-TGTa(2))^2;
```

2.6 Cons_th.m

```
1 function [g,h,psi_dot]=cons_th(x,s0,N,tf,V)
2
3 dt=tf/N;
4 % calculate psi
5 psi_dot = abs(s0(3)-x(3))/dt;
6 g = psi_dot - 3;
7
8 h(1)      = x(1)-s0(1)-dt*V*cos(deg2rad(x(3)));
9 h(N+1)    = x(N+1)-s0(N+1)-dt*V*sin(deg2rad(x(3)));
```

Bibliography

[1] Amato, N.M. and Y. Wu. "A Randomized Roadmap Method for Path and Manipulation Planning", *IEEE Int. Conf. Robot. Autom.*, 1996.

[2] Arena, Mark V., John C. Graser, and Paul DeLuca. *Implications of an Air Force Budget Downturn on the Aircraft Industrial Base*. Technical report, RAND Corporation, 2013.

[3] Arora, Jasbir. *Introduction to Optimum Design*. 2011.

[4] Chang, Justin, Elioth Fraijo, Zack Grannan, Steven Kestler, Eric Lo, Bryan Ritoper, and Jordan Sendar. *Technical Paper*. Technical report, University of California, San Diego, 2012.

[5] Deghat, Mohammad, Iman Shames, Brian D. O. Anderson, and Changbin Yu. "Target Localization and Circumnavigation Using Bearing Measurements in 2D". *49th IEEE Conference on Decision and Control*. 2010.

[6] Grabbe, Michael T. and Brandon M. Hamschin. "Geo-Location Using Direction Finding Angles", *Johns Hopkins APL Technical Digest*, 2013.

[7] Headquarters, United States Air Force. *United States Air Force RPA Vector*. 2014.

[8] Koks, Don. *Numerical Calculations for Passive Geolocation Scenarios*. Technical report, Defence Science and Technology Organization, 2007.

[9] LaValle, Steven M. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*. Technical report, Department of Computer Science, Iowa State Universsity, 1998.

[10] Moell, Joseph D. and N.Thomas Curlee. *Transmitter Hunting*. 1987.

[11] Ross, Steven M., Richard G. Cobb, and William P. Baker. "Stochastic Real-Time Optimal Control for Bearing-Only Trajectory Planning", *International Journal of Micro Air Vehicles*, 2014.

[12] Shames, Iman, Baris Fidan, and Brian D. O. Anderson. "Close Target Reconnaissance Using Autonomous UAV Formation". *47th IEEE Conference on Decision and Control*. 2008.

[13] Torrieri, J. Don. "Statistical Theory of Passive Location Systems", *IEEE Trans. Aero. Elect. Syst.*, 1984.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
26–03–2015	Master's Thesis	Aug 2013 – Mar 2015

4. TITLE AND SUBTITLE

Optimized Flight Path for
Localization using Line of Bearing

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Namkyu Kim, ROKAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENY-MS-15-M-246

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

AFRL/RQQA
2210 8th Street
Bldg 146, Room 300, Wright-Patterson AFB, OH 45433
COMM 937–713–7038
e-mail : derek.kingston@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Distribution Statement A :
Approved for Public Release; Distribution Unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT This research develops optimized flight paths for localization of a target using LOB measurements. The target area is expressed as an error ellipse using the measurement errors of the LOBs. The optimization approach is focused on minimizing the size of the error ellipse. The algorithm for the optimized path is generated and compared with typical flight paths. The optimization routine is based on the results revised from previous similar research in the literature.

A geometrical method to estimate the error ellipse is combined with optimal control in this research. Each LOB gives a possible target area and this target area can be reduced by overlapping areas developed from multiple LOBs. The algorithm based on this method is tested with a single target and with multiple targets in simulation. In addition to analytical simulations of the proposed method, a real-world test is conducted using a remotely controlled truck. From the simulation and a real-world test, the change of the semi-major axis of the error ellipse with increasing number of measurements and the total number of measurements needed for to achieved predefined semi-major axis are verified.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Richard G. Cobb, PhD
U	U	U	UU	123	19b. TELEPHONE NUMBER *(include area code)* (937) 785–3636 ext 4559; Richard.Cobb@afit.edu