The Pennsylvania State University
The Graduate School

REAL-TIME TRAJECTORY GENERATION FOR TARGET LOCALIZATION USING MICRO AIR VEHICLES

A Thesis in
Aerospace Engineering
by
Jeffrey B. Corbets

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Master of Science

August 2008

The thesis of Jeffrey B. Corbets was reviewed and approved* by the following:

Jacob W. Langelaan
Assistant Professor of Aerospace Engineering
Thesis Advisor

Joseph F. Horn
Associate Professor of Aerospace Engineering

George A. Lesieutre
Professor of Aerospace Engineering
Department Head

*Signatures are on file in the Graduate School.

Abstract

This thesis presents an approach to near-optimal target localization for small and micro unmanned aerial vehicles using a family of pre-computed parameterized trajectories. These trajectories are pre-computed for a set of nominal target locations uniformly distributed over the sensor field of view and stored offline in a non-dimensionalized form. In the first part of this research, the trajectories are parameterized and stored as a sequence of turn-rate commands. In the second part of this research, the trajectories are parameterized and stored as a sequence of non-dimensional waypoints. Upon target detection, a trajectory corresponding to the nearest nominal target location is selected and dimensionalized. An onboard navigation controller follows the dimensionalized trajectory. Thus, trajectory generation occurs in near-constant time, which allows for fast adaptation as the target state estimate is refined. Non-dimensionalization of the trajectories with respect to relative vehicle speed, sensor range, and sensor update rate allows the same table to be used for various combinations of sensor package and vehicle or vehicle operating conditions.

Table of Contents

List of Figures

List of Tables

List of Symbols

B	Vehicle body reference frame, page 12
CN	"Characteristic number" defining the vehicle and sensor package properties used in the trajectory table, page 36
J	Cost function to be optimized, page 21
J_{info}	Information term in the cost function, page 32
J_{safe}	Safety zone term in the cost function, page 32
N	Duration of the planning horizon, page 30
O	Inertial reference frame, page 12
P	Covariance matrix of the state estimate of the target, page 18
R	Maximum sensor range, page 15
T_c	Duration of the control horizon, page 36
T_f	Sensor frame sample time, page 26
T_s	Vehicle kinematics sample time, page 26
Y	Fisher Information Matrix, page 18
$\Delta\mathbf{Y}_k$	Increase in Fisher Information at discrete time k, page 19
$\Delta\tilde{\mathbf{Y}}_k$	Increase in non-dimensional Fisher Information at discrete time k, page 27
Σ_ν	Covariance of the Gaussian random noise associated with the bearing measurement, page 15

β Optimization parameter controlling step size adjustment, page 34

$\dot{\psi}_v$ Time-based derivative of observer vehicle heading, page 16

$\dot{\tilde{\psi}}$ Non-dimensional change in heading of the observer vehicle, page 26

$\dot{\tilde{x}}$ Non-dimensional velocity of the observer vehicle in the x-dimension, page 26

$\dot{\tilde{y}}$ Non-dimensional velocity of the observer vehicle in the y-dimension, page 26

$\dot{x}_v$ Time-based derivative of observer vehicle position in the x-direction, page 16

$\dot{y}_v$ Time-based derivative of observer vehicle position in the y-direction, page 16

γ Bearing between the observer and target vehicles, page 12

γ_{max} Sensor field of view limit, page 15

γ_{opt} Optimal angle between two successive measurements for the second Fisher Information Matrix example, page 22

$\hat{\mathbf{x}}_i$ Estimated state vector of the target at the end of a single run within a Monte Carlo simulation, page 42

$\hat{\mathbf{x}}_t$ Estimate of the state vector of the target, page 16

$\mathbf{H}_k$ Jacobian of the measurement model used in the calculation of the Fisher Information Matrix, page 18

$\mathbf{P}_0$ Initial covariance matrix, page 19

$\mathbf{Y}_0$ Initial Fisher Information Matrix, page 19

$\mathbf{Y}_k$ Fisher Information Matrix at the end of the trajectory, page 23

$\mathbf{Y}_k$ Fisher Information Matrix calculated at a discrete time k, page 18

$\mathbf{e}$ Tracking error associated with the estimate of the state vector of the target, page 18

$\mathbf{u}_{mn}$ Sequence of turn rate commands parameterizing trajectory mn, page 30

$\mathbf{x}_1 \dots \mathbf{x}_n$	State vectors of the targets in a sequential target localization task, page 12
$\mathbf{x}_v$	State vector of the observer vehicle, page 12
$\mathbf{x}_t$	State vector of the target, page 14
ν	Gaussian random noise associated with uncertainty in the sensor measurement of the bearing to the target, page 15
ν_v	Gaussian random noise associated with uncertainty in the velocity of the kinematic model of the target, page 16
ν_ψ	Gaussian random noise associated with uncertainty in the heading of the kinematic model of the target, page 16
ψ_t	Heading of the target, page 14
ψ_v	Heading of the observer vehicle, page 15
θ	Angle from the nominal target location to the waypoint, page 30
$\tilde{\Delta}t$	Integration time step in the non-dimensinal discrete time kinematics model, page 26
$\tilde{\kappa}_k$	Non-dimensional curvature of a trajectory at discrete time k, page 32
$\tilde{\mathbf{H}}_k$	Non-dimensional Jacobian of the sensor measurement model, page 27
$\tilde{\mathbf{X}}_{mn}$	Sequence of non-dimensional waypoints parameterizing trajectory mn, page 30
$\tilde{\mathbf{Y}}_k$	Non-dimensional Fisher Information Matrix at discrete time k, page 27
$\tilde{\mathbf{x}}_k$	Non-dimensional state vector of the observer vehicle at discrete time k, page 26
e_i	Location tracking error associated with the target position estimate at the end of a single run within a Monte Carlo simulation, page 42
f	Interpolating function used in the waypoint-based optimization problem, page 32
i	Index of an individual run within a Monte Carlo simulation, page 42
k	A discrete step in time, page 16

r	Distance from the nominal target location to the waypoint, page 30
r_k	Range to the target at discrete time k, page 18
r_{safe}	Radius of the safety zone around the target, page 34
t	Time, page 30
t_0	Initial time of the planned trajectory, page 30
u	Input turn-rate command, page 16
v	Observer vehicle speed, page 36
v_t	Speed of the target, page 14
w_{fov}	Weight factor associated with the field of view term in the cost function, page 32
w_{info}	Weight factor associated with the information term in the cost function, page 32
w_{safe}	Weight factor associated with the safety zone term in the cost function, page 32
x_n	Non-dimensional x-position of waypoint n, page 31
x_t	Position of the target in the x-dimension, page 14
x_v	Position of the observer vehicle in the x-dimension, page 15
y_n	Non-dimensional y-position of waypoint n, page 31
y_t	Position of the target in the y-dimension, page 14
y_v	Position of the observer vehicle in the y-dimension, page 15

Acknowledgments

This work would not be possible without the support of a great many people. First off, I would like to thank my entire family for their support and encouragement throughout my life. My advisor, Prof. Jack Langelaan at Penn State, kept this work moving in a forward direction. The entire Penn State Aerospace Department deserves mention for keeping me around (or even just for putting up with me) for graduate school after four years of undergraduate education. Finally, the people at the American Society of Engineering Education who administer the National Defense Science and Engineering Graduate (NDSEG) Fellowship and the Air Force Office of Scientific Research (AFOSR) deserve special thanks for providing me with graduate funding.

Chapter 1

Introduction

This thesis describes the development of a technique for fast, adaptive trajectory planning suitable for deployment on micro air vehicles (μAVs) or autonomous submunitions. Missions envisioned for these vehicles typically include surveillance and target tracking. The research was motivated by a combination of the limited computing power typically available on these vehicles and the limited sensing which can be carried. The sensing limitations complicates the problem of target tracking due to the limited information which can be obtained about the target. Generally, only the availability of a bearing sensor, such as a monocular camera, is assumed. The target tracking problem is further complicated by the non-linearity of the measurement model.

The combination of limited information, a bearing to the target provides no information about the range to the target, and the non-linearity of the measurement model leads to a problem of dynamic observability

This thesis: (a) describes a framework for adaptive trajectory generation based on a non-dimensionalized table of optimal trajectories; (b) describes the process of generating the trajectory table; (c) presents simulation results demonstrating the performance of this table-based approach to trajectory planning.

1.1 Motivation

Unmanned aerial vehicles are not a new idea in the world of aerospace engineering. Originally known as "drones," unmanned aerial vehicles, or UAVs, were initially

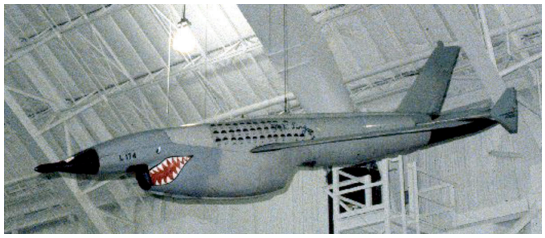
used as targets for air combat practice. In the late 1960s, a new mission for unmanned aerial vehicles was developed: surveillance. Surveillance missions often meet the so-called "three Ds" criteria, missions that are dull, dangerous, or dirty. Long duration surveillance missions are often boring as the search or surveillance pattern is flown. If the surveillance mission is flown in a hostile environment, the mission is inherently dangerous. Finally, unmanned vehicles can be used to enter areas contaminated by radiological, biological, or chemical weapons - dirty areas - that cannot be safely surveyed by humans.

An example of early use of UAVs for reconnaissance is the Teledyne Ryan AQM-34 Firebee, derived from the BQM-34 family of target drones, shown in Figure 1.1. It was used extensively during the Vietnam War in the 1960s and early 1970s. Versions of unmanned aerial vehicles developed through the early-to-mid 1990s still relied on a pilot guiding the vehicle from a ground station. The pinnacle of unmanned aerial vehicles are autonomous aerial vehicles. These vehicles use onboard computers and special software to complete a pre-defined task. With the computational power available today, autonomous aerial vehicles are generally capable of performing one or more of the following tasks - transport, scientific research, remote sensing or surveillance, and precision bombing. As computers become both more powerful and require less resources including space and electrical power, the size of the autonomous aerial vehicle can decrease. From 1992 to 2008, the size of unmanned aerial vehicles designed primarily for surveillance missions has decreased significantly and can also be seen in Figure 1.1.

Actually being able to complete a surveillance or target tracking mission using a μAV requires advances in several fields, including though not limited to - flight control, sensing systems, obstacle avoidance, state estimation, and trajectory planning. The small size of μAVs complicates the problem because of the limited power, sensing, and computation which can be carried on board. Managing the trade-offs requires careful system design.

This thesis is concerned with target tracking and state estimation, specifically trajectory design to maximize the information gained about the target. It is assumed that observer vehicle state is known precisely (for example, using GPS) and that the only sensing available is a monocular camera fixed to the observer vehicle.

A monocular camera provides bearings to targets (or features) in the environ-



(a) Firebee UAV

(b) Predator UAV

(c) Mosquito μAV

(d) Wasp μAV

Figure 1.1. Representative UAVs designed for surveillance missions.

ment. A single bearing provided by a monocular camera does not provide enough information to localize a target. Fusing bearings from multiple vantage points, however, allows target localization by triangulating measurements. This problem of triangulation can readily be cast as a non-linear estimation problem and a recursive estimator such as an Extended Kalman Filter (EKF) or Unscented Kalman Filter (UKF) can be implemented.

When an estimator is implemented to solve the bearings-only target localization problem, the lack of range information results in *dynamic observability*: multiple measurements, collected over time, from varying vantage points, allow estimation of target state. Because of sensor noise, the geometry of observer positions and target position greatly affects the accuracy of the target state estimate. This leads to the subject of this thesis: What is the optimal trajectory which will localize a target

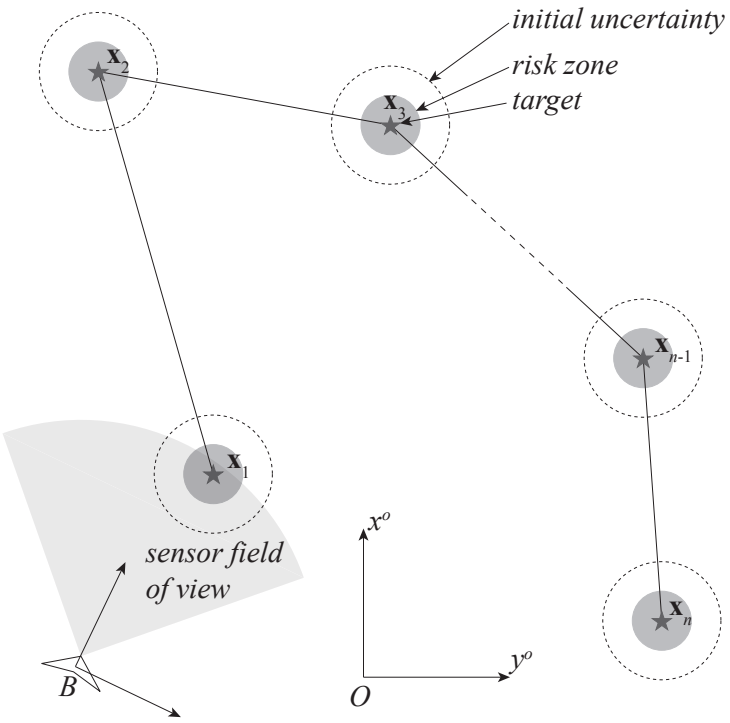


Figure 1.2. Schematic of target localization task showing a sequence of targets to be tracked.

with the smallest degree of uncertainty? Furthermore, how can this trajectory be computed in real-time on computing hardware likely to be available on a μAV or autonomous sub-munition?

While the technology is general to many trajectory generation problems, the motivating mission is target state estimation for a μAV. A schematic of a sequential target localization task (where the sequence of targets to be visited is determined a priori) is shown in Figure 1.2.

A human operator provides a sequence of targets and initial (possibly highly uncertain) estimates of target positions. Each target has an associated risk zone which must be avoided to reduce the likelihood of detection and possible loss of the observer vehicle. The vehicle has a limited field of view sensor, and must plan a sequence of trajectories to minimize the uncertainty in the final state estimate of each target.

1.1.1 Some Important Terms

Throughout the remainder of this thesis, a few important terms are used that should be defined.

For the purposes of this work, an unmanned aerial vehicle, or UAV, shall refer to an autonomous unmanned aerial vehicle. The ground station for this vehicle is used only to send high-level commands rather than low-level control inputs. For example, the human operator could send the command "Fly the pattern defined by the following set of way points." The vehicle has enough autonomy that it can fulfill this high-level command. This research specifically focuses on trajectory generation for μAVs, or micro air vehicles. These vehicles are a subset of UAVs that have wingspans of approximately six (6) inches.

An observer refers to the unmanned aerial vehicle coupled with the on-board sensor package. The observer follows the trajectory specified by the trajectory generator. Because the on-board sensor package is rigidly attached to the flight vehicle, the trajectory followed by the observer is the same as the trajectory of the sensor. Note that this is not the same as the usual controls-theoretic definition of observer.

The target is the object of interest in the surveillance mission. The target's state (i.e. its position and velocity) is estimated by the observer.

Trajectory generation refers to the optimization of a path to be followed by the observer. This path may also be called a trajectory. In this work, trajectory generation occurs before the vehicle is launched.

Finally, monocular vision is the process of using a single digital camera and computer system to determine information about the world around the camera from a series of pictures taken over time.

1.2 Fast, Adaptive Trajectory Selection

A schematic of a system for target tracking using a μAV is shown in Figure 1.3. It consists of 5 parts: (a) an aircraft, which is acted upon by external disturbances and has as input control commands; (b) a flight control system which enables controlled flight and has as input a desired trajectory; (c) a trajectory generator;

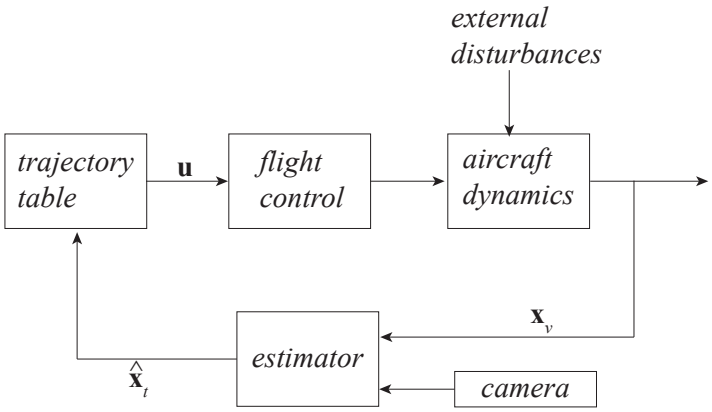


Figure 1.3. Block diagram of the system.

(d) an estimator which combines vehicle state information with measurements from a vision system to compute an estimate of target state; (e) a camera, which provides bearing measurements to the target.

The problem of state estimation for autonomous vehicles such as μAVs has been studied in great detail and seems to be well understood for these systems: indeed, several textbooks dealing with state estimation and target tracking have been published [1, 2, 3]. Low-dimensional state estimation problems (such as target tracking) can easily be implemented on relatively simple computers; planning trajectories which maximizes information gain is, however, still a difficult problem.

This thesis proposes a method for fast, adaptive trajectory selection based on a table of trajectories for a set of *nominal* target locations. Upon receipt of an initial (highly uncertain) target location, the trajectory corresponding to the nearest nominal target location is selected and the observer vehicle begins to fly the trajectory. Bearing measurements to the target are obtained and an estimator updates the target state estimate in real time. As the target state estimate improves, a new trajectory can be selected from the table.

1.2.1 The Trajectory Generation Problem

The critical technology described is the development of a table of trajectories for a set of nominal target locations. Several questions are addressed:

- What is the appropriate parameterization of the trajectory?
  Two parameterizations are discussed: in the first, the trajectory is represented as a sequence of turn-rate commands. In the second, the trajectory is represented as a sequence of waypoints, and it is assumed that the vehicle follows a spline path between waypoints.

- Can the trajectory table be generalized to different vehicle and sensor combinations?
  The planning problem is first non-dimensionalized using sensor range, vehicle speed and sensor frame rate as parameters. The resulting table of trajectories is thus general to variations in vehicle speed, sensor range and sensor frame rate.

1.3 Related Work

The main focus of this thesis is trajectory generation. However, state estimation is a key contributing technology, thus a brief (and necessarily incomplete) overview of the state of the art in target state estimation is provided. Here the focus is on state estimation using vision and on planning algorithms which attempt to maximize information gain.

1.3.1 Estimation

Estimation is the problem of transforming noisy measurements into information about the variables of interest (states). In the context of autonomous vehicles and mobile robots, estimation problems can be loosely divided into three categories: localization, where vehicle position, orientation and velocity are the states of interest (direct information concerning the environment is unnecessary); mapping, where information about the environment is desired (typically vehicle state is assumed

known); and simultaneous localization and mapping (SLAM), where the vehicle must obtain information about its own state and about the environment.

Localization using vision has been used in many contexts including approach and landing for UAVs [4], as an aid for inertial navigation [5, 6], aircraft state estimation [7, 8], navigation for autonomous underwater vehicles [9, 10, 11], and for wheeled vehicles such as the Mars Exploration Rovers [12].

Mapping applications include terrain reconstruction for landing autonomous helicopters [13], obstacle avoidance in urban environments [14], relative position sensing for underwater vehicles [15], reconstruction of archaeological sites [16], and stereo vision to create digital reconstructions of three-dimensional scenes and collision avoidance systems [17, 18, 19]

SLAM has become a very important field of research, described in several textbooks (e.g. Thrun et al. [20] and hundreds of conference and journal publications. Vision-based methods have been applied to problems in wearable computing [21, 22], wheeled ground robots [23] and UAVs [24, 25].

The problems of target localization (this assumes a stationary target) and target tracking (which allows for a moving target) can be considered examples of a mapping problem, since the position of the observer vehicle is assumed known.

1.3.2 Trajectory Design

The field of robot motion planning is similarly broad, with a long history of study. Two notable textbooks are Latombe [26] and LaValle [27].

Because of the dynamic observability caused by the bearings-only sensor, the trajectory followed by the observer vehicle has an enormous effect on the quality of state estimates[28] and optimal trajectory generation for target localization or tracking has become an active area of research[29, 30, 31].

Computing the optimal trajectory for a realistic vehicle model and realistic sensor models can become computationally prohibitive, and simplified models are generally used. For example, vehicle dynamics have been modeled as a point mass with velocity and acceleration constraints[31] and sensor models have been linearized[32]. Solution methods including dynamic programming[33] and direct collocation[34] have been used to generate the optimal trajectories. These tech-

niques still require fairly powerful computers and depending on the complexity of the model (e.g. field of view constraints also increase complexity) may not be suitable for real-time operation on the processors likely to be available on a μAV.

1.4 Summary of Contributions

The main contributions of this thesis are:

- **A method for fast, adaptive trajectory selection.**
  A method of trajectory selection based on a table of pre-computed trajectories for a set of nominal target locations is proposed. When a target is detected or specified by a human operator, the vehicle selects the trajectory corresponding to the nearest nominal target location and follows the trajectory. Adaptation to improvements in the target state estimate is enabled by selecting a new trajectory when it is determined that sufficient change in target state estimate has occurred.

- **Non-dimensionalization of the table of trajectories.**
  This non-dimensionalized table of trajectories is general to variations in vehicle speed, sensor range, and sensor frame rate, and is thus applicable to different vehicle and sensor combinations.

- **Performance verification through simulations.**
  Results of Monte Carlo simulations show that the information gained about the target using the trajectory table is almost the same as that gained by direct computation of optimal trajectories (90% of the information is gained when following trajectories obtained from the table) at vastly reduced computational overhead. Using the waypoint parameterization, a trajectory can be selected and dimensionalized 130 times faster than a trajectory optimized online.

1.5 Reader's Guide

The remainder of this thesis is organized as follows:

- **Chapter 2: Theory** begins with a formulation of the trajectory generation problem. It defines models for the observer vehicle kinematics, vision measurements and target model, and then describes the solution of the state estimation problem. Finally it describes the use of the Fisher Information Matrix in trajectory generation.

- **Chapter 3: A Table of Optimized Trajectories** describes the table of trajectories which is used for trajectory selection. The chapter discusses non-dimensionalizing the trajectory generation problem and two methods of trajectory parameterization (turn-rate and waypoint). It describes the process for generating non-dimensional optimal trajectories and how the trajectories are stored for use by the observer vehicle.

- **Chapter 4: Simulation Results** presents results of simulations showing the performance of the proposed system.

- **Chapter 5: Conclusion** summarizes results of this research and discusses areas for future work.

Chapter 2

Theory

2.1 Introduction

This chapter will provide an overview of the problem formulation as well as provide some background on the relevant theories and ideas used throughout the research. First, the problem formulation is presented in Section 2.2. After the problem formulation is presented, the kinematics and sensor models of the observer vehicle will be given in Section 2.3. In Section 2.4 a brief statement about target state estimation is made and the algorithm for the Unscented Kalman Filter (UKF) is presented. This is followed by an introduction to information theory and the Fisher Information Matrix in Section 2.5. To illustrate the use of the Fisher Information Matrix in trajectory planning two sample problems are described and solved analytically in Section 2.6.

2.2 Problem Formulation

The problem of a μAV or autonomous submunition performing a surveillance and target tracking task is considered in this thesis. An on-board vision system (e.g. a monocular camera) obtains bearing measurements to the target. The observer vehicle position is assumed to be known precisely. The speed of the target vehicle is assumed to be constant. A schematic of a sequential target localization task (where the sequence of targets to be visited is determined a priori) is shown in

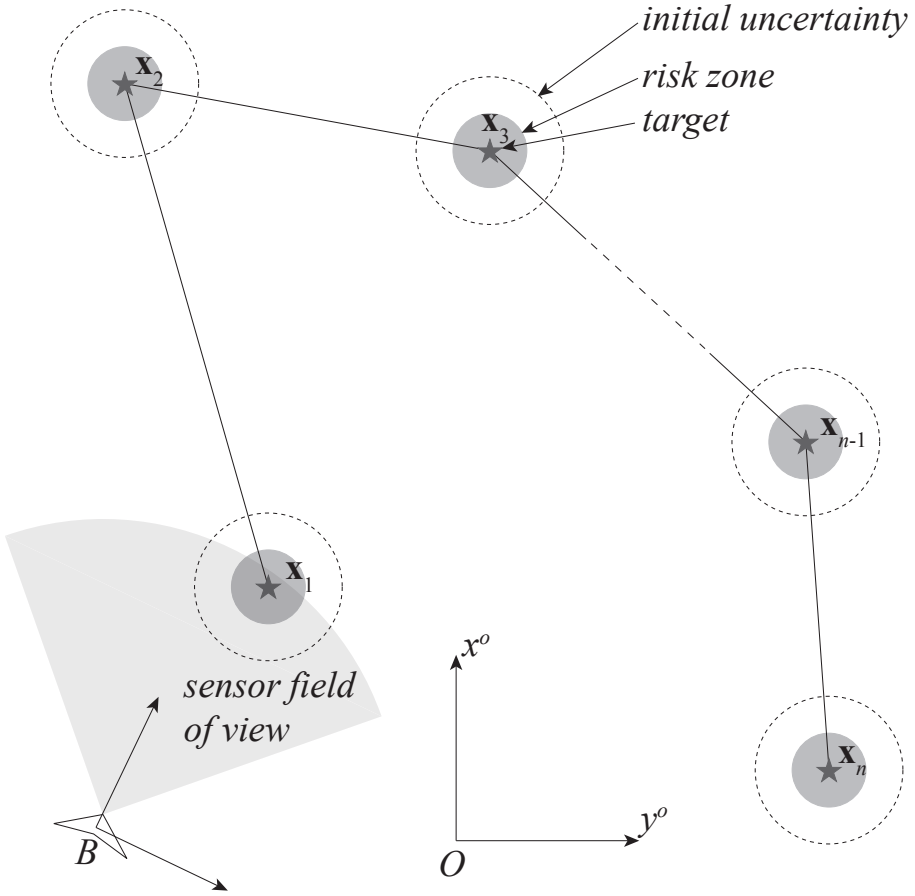


Figure 2.1. Schematic of target localization task showing a sequence of targets to be tracked.

Figure 2.1.

Vehicle and target positions are denoted $\mathbf{x}_v$ and $\mathbf{x}_1 \ldots \mathbf{x}_n$, respectively, in an inertial frame O. The vision system obtains a bearing γ to the target (in the vehicle body frame B). An estimation algorithm (implemented using a Sigma Point Kalman Filter[35, 36]) uses knowledge of vehicle position and the bearing measurements to compute an estimate of target position. A block diagram showing the flow of the system is given in Figure 2.2.

With only a monocular camera aboard to gain information about a target, the path flown by the observer vehicle greatly affects the amount of uncertainty in the estimation of the target position. This is because a two-dimensional picture provides no information on the third dimension: depth or range to the target.

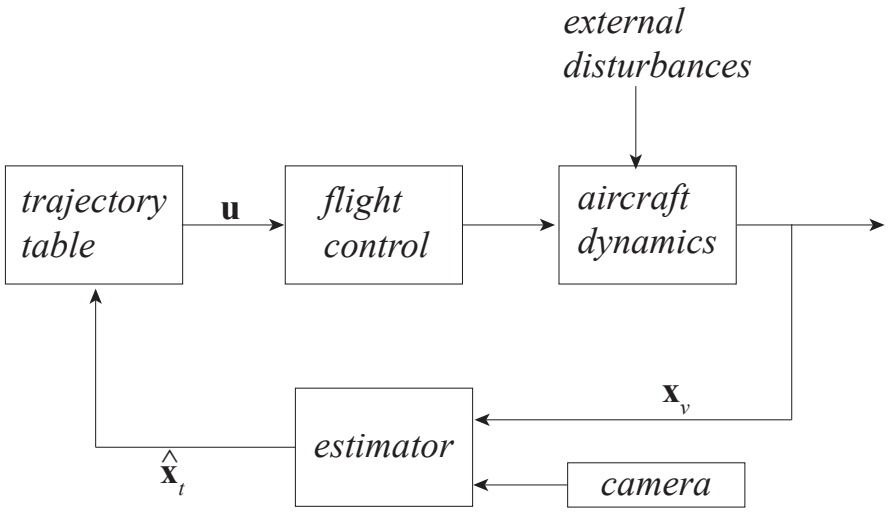


Figure 2.2. Block diagram of the system.

However, by fusing data from pictures taken at different vantage points, target position can be triangulated. Because of sensor noise the geometry of the observer and target positions affects the accuracy of the range calculation. This work focuses on computing a trajectory for a small UAV that minimizes the uncertainty in the estimated position of a target given a bearings-only sensor with a limited field of view, i.e. a non-gimbaled monocular vision system.

A schematic of a target localization task is shown in Figure 2.3. Observer vehicle and target positions are denoted $\mathbf{x}_v$ and $\mathbf{x}_t$, respectively, in an inertial frame O. A highly uncertain initial target position is assumed to be given and shown in the figure by the dashed line surrounding the target position. The target has a safety zone surrounding it, denoted in the figure by the gray circle surrounding the target position. The safety zone ensures the observer vehicle does not fly too close to the target, putting the observer vehicle at risk from target defense systems or collision. The vision system obtains a bearing γ to the target in the vehicle body frame B. The vision system has a limited field of view, defined by $\pm\gamma_{max}$. The maximum range of the sensor system is shown in the figure by the top arc at range R. An estimation algorithm uses knowledge of vehicle position and the bearing

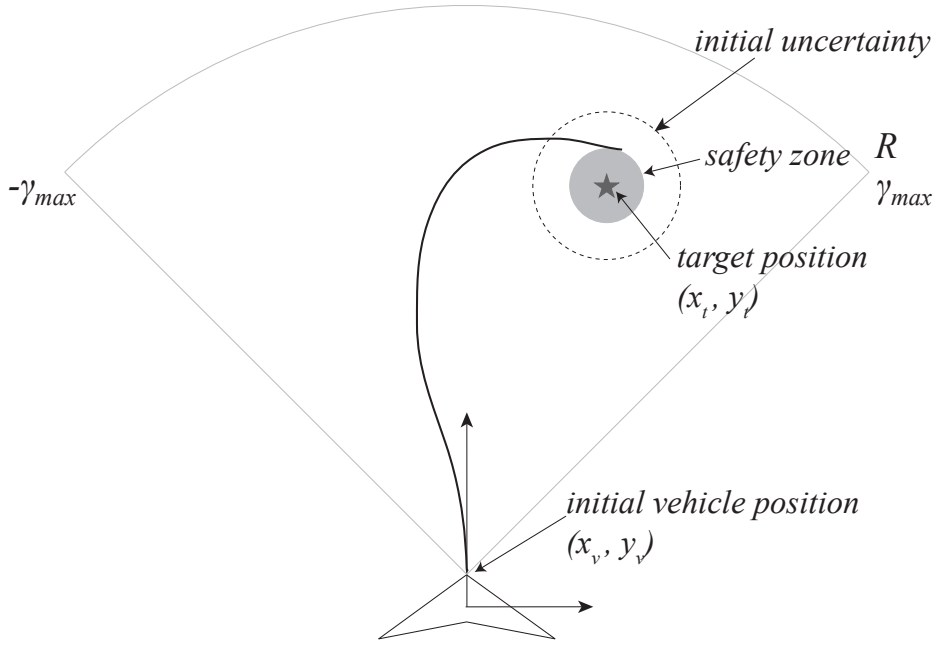


Figure 2.3. Schematic of target localization task.

measurements to compute an estimate of target position. Two implementations of this problem were used over the course of the research and will be discussed further in Chapter 3.

In determining the position of the target, the following states are estimated:

$$\mathbf{x}_t = \begin{bmatrix} x_t \\ y_t \\ v_t \\ \psi_t \end{bmatrix} \tag{2.1}$$

where x_t and y_t are the two-dimensional position of the target, v_t is the speed of the target, and ψ_t is the heading of the target, or direction in which the target is moving.

For this problem, the initial uncertainty of the target position is assumed to be unbiased in any direction and greater in area than the area determined by the safety zone around the target. A safety zone is included in this problem formulation as

oftentimes targets for UAVs may be hostile or located in an environment dangerous for a UAV, e.g. close to trees or other structures. As such the safety zone defines the area with a large amount of risk for the UAV. In the case of an autonomous munition, the edge of the safety zone is where a terminal guidance algorithm would take over control of the vehicle.

The problem of target state estimation is clearly critical in the target tracking problem. Solutions to this problem have been well-represented in the literature, and the focus here is on planning trajectories to maximize information gained about the target. Because of the non-linearities in the system models, estimation and planning are tightly coupled, and a discussion of this coupling follows.

2.3 Sensor and Vehicle Models

2.3.1 Monocular Vision System Model

The vision system obtains a bearing to the target:

$$\gamma = \arctan\left(\frac{y_t - y_v}{x_t - x_v}\right) - \psi_v + \nu \tag{2.2}$$

where $x_t,\ y_t$ represent the location of the stationary target in the 2D plane; $x_v,\ y_v,\ \psi_v$ represent the vehicle position and heading; and ν is uncorrelated zero-mean Gaussian random noise with covariance Σ_ν. Maximum sensor range is R and the sensor field of view is limited to $-\gamma_{max} \leq \gamma \leq \gamma_{max}$. The sensor sample period, which is the inverse of the frame rate for the vision system, is T_f.

2.3.2 Observer and Target Motion Model

Onboard the μAV, a guidance controller provides velocity and turn rate commands, leading to a kinematic vehicle model. The velocity of the observer vehicle is assumed to be a constant, v:

$$\dot{x}_v = v\cos\psi_v \tag{2.3}$$

$$\dot{y}_v = v\sin\psi_v \tag{2.4}$$

$$\dot{\psi}_v = u \tag{2.5}$$

where u is a commanded turn rate from the waypoint-following controller.

The target motion model is also that of a non-holonomic vehicle, such as a car, ship, or fixed-wing airplane. Again, the velocity of the target vehicle is assume to be a constant, v_t:

$$x_{t,k} = x_{t,k-1} + v_{t,k-1}\cos\psi_{t,k-1} \quad (2.6)$$

$$y_{t,k} = y_{t,k-1} + v_{t,k-1}\sin\psi_{t,k-1} \quad (2.7)$$

$$v_{t,k} = v_{t,k-1} + \nu_v \quad (2.8)$$

$$\psi_{t,k} = \psi_{t,k-1} + \nu_\psi \quad (2.9)$$

where ν_v and ν_ψ are uncorrelated zero-mean Gaussian random noise terms associated with uncertainty in the velocity and heading, respectively.

2.4 Target State Estimation

For the results presented in this thesis, the target is assumed to be stationary, hence the target state vector can be reduced to $\mathbf{x}_{t,k} = [x_t y_t]^T$ and $\mathbf{x}_{t,k+1} = \mathbf{x}_{t,k}$. The bearing model given in Equation 2.2 results in a non-linear estimation problem, and the algorithm for a Sigma Point Kalman Filter (i.e. an Unscented Kalman Filter) given in van der Merwe and Wan[35] is used to compute the target state estimate, $\hat{\mathbf{x}}_t$. The algorithm used in the Sigma Point Kalman Filter is given in Figure 2.4.

The purpose of the estimator is to compute an estimate, $\hat{\mathbf{x}}_t$, of the state $\mathbf{x}_t$ and an estimate, $\mathbf{P}$, of the covariance of the estimation error. The trajectory planning algorithm will find a path which minimizes the uncertainty $\mathbf{P}$ of the target state estimate.

2.5 The Fisher Information Matrix (FIM)

The idea of information was developed first in the research of thermodynamics and exists as a way to measure the amount of "information" a known variable contains about a second unknown variable. In this research, the accuracy of the

initialize with $\hat{\mathbf{x}}_0$ and $\mathbf{P}_{0|0}$.

For t_k, $k \in (1, \ldots, \infty)$ compute sigma points:

$$\mathbf{X}_{k-1|k-1} = \left[\begin{array}{ccc} \hat{\mathbf{x}}_{k-1|k-1} & \hat{\mathbf{x}}_{k-1|k-1} + \eta\sqrt{\mathbf{P_{k-1|k-1}}} & \hat{\mathbf{x}}_{k-1|k-1} + \eta\sqrt{\mathbf{P_{k-1|k-1}}} \end{array} \right] \tag{2.10}$$

Time update (prediction):

$$\mathbf{X}_{k|k-1} = f(\mathbf{X}_{k-1|k-1}, \mathbf{u}_{k-1}) \tag{2.11}$$

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{X}_{k|k-1}\mathbf{w}_m \tag{2.12}$$

$$\mathbf{P}_{k|k-1} = \left[\mathbf{X}_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\mathbf{1}\right]^T \mathbf{W}_c \left[\mathbf{X}_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\mathbf{1}\right] + \mathbf{Q} \tag{2.13}$$

Measurement update (correction):

$$\mathbf{Z}_{k|k-1} = h(\mathbf{X}_{k|k-1}) \tag{2.14}$$

$$\hat{\mathbf{z}}_{k|k-1} = \mathbf{Z}_{k|k-1}\mathbf{w}_m \tag{2.15}$$

$$\mathbf{P}_{zz} = \left[\mathbf{Z}_{k|k-1} - \hat{\mathbf{z}}_{k|k-1}\mathbf{1}\right]^T \mathbf{W}_c \left[\mathbf{Z}_{k|k-1} - \hat{\mathbf{z}}_{k|k-1}\mathbf{1}\right] + \mathbf{R} \tag{2.16}$$

$$\mathbf{P}_{xz} = \left[\mathbf{X}_{k|k-1} - \hat{\mathbf{x}}_{k|k-1}\mathbf{1}\right]^T \mathbf{W}_c \left[\mathbf{Z}_{k|k-1} - \hat{\mathbf{z}}_{k|k-1}\mathbf{1}\right] \tag{2.17}$$

$$\mathbf{K} = \mathbf{P}_{xz}\mathbf{P}_{zz}^{-1} \tag{2.18}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}(\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}) \tag{2.19}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}\mathbf{P}_{zz}\mathbf{K}^T \tag{2.20}$$

There are $2N + 1$ sigma points, where N is the dimension of the state vector. In this algorithm η is a weight factor, $\mathbf{w}_m$ is a vector of weights, $\mathbf{W}_c$ is a diagonal matrix of weights, $\mathbf{1}$ is a $(1 \times 2N + 1)$ matrix of ones, $\mathbf{Q}$ is process noise and $\mathbf{R}$ is measurement noise. The weight factors are calculated as

$$\eta = \alpha\sqrt{N} \tag{2.21}$$

The constant α is a parameter which determines the spread of the sigma points. Typically $10^{-4} \leq \alpha \leq 1$. The weight vector $\mathbf{w}_m$ and weight matrix $\mathbf{W}_c$ are

$$\mathbf{w}_{m,1} = \tfrac{\alpha^2-1}{\alpha^2} \qquad \mathbf{w}_{m,i} = \tfrac{1}{2N\alpha^2} \tag{2.22}$$

$$\mathbf{W}_{c,1} = \tfrac{\alpha^2-1}{\alpha^2} + (1 - \alpha^2 + \beta) \qquad \mathbf{W}_{c,ii} = \tfrac{1}{2N\alpha^2} \tag{2.23}$$

where $i = 2, \ldots, (2N + 1)$. The parameter β incorporates prior knowledge of the distribution of the state vector. For Gaussian distributions $\beta = 2$ is optimal [35].

Figure 2.4. Algorithm for Unscented Kalman Filter.

state estimate is measured using the tracking error $\mathbf{e} = \mathbf{x}_t - \hat{\mathbf{x}}_t$. Minimizing the uncertainty $\mathbf{P}$ in this error is equivalent to maximizing the information $\mathbf{Y}$ as the two are inverses of each other, $\mathbf{Y} = \mathbf{P}^{-1}$.

To illustrate the use of Fisher Information in a target tracking application, consider a discrete time system with trivial dynamics and non-linear measurement model

$$\mathbf{x}_{k+1} = \mathbf{x}_k \tag{2.24}$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \tag{2.25}$$

where $\mathbf{v}_k$ is uncorrelated zero-mean Gaussian random noise.

As shown by Ousingsawat and Campbell[31], the FIM for the estimation problem associated with this system can be computed recursively

$$\mathbf{Y}_k = \mathbf{Y}_{k-1} + \mathbf{H}_k^T \Sigma_v^{-1} \mathbf{H}_k \tag{2.26}$$

where $\mathbf{H}_k$ is the Jacobian of the measurement model evaluated at time k, i.e.

$$\mathbf{H}_k = \frac{\delta}{\delta \mathbf{x}} h(\mathbf{x}_k) \tag{2.27}$$

For the vision model given by Equation 2.28 the Jacobian of the sensor model with respect to the estimate of the target is

$$\mathbf{H}_k = \begin{bmatrix} -\frac{\sin\gamma_k}{r_k} & \frac{\cos\gamma_k}{r_k} \end{bmatrix} \tag{2.28}$$

where $r_k = \sqrt{(x_t - x_{v,k})^2 + (y_t - y_{v,k})^2}$. The information gained about the target from a single measurement can now be expressed as

$$\mathbf{Y}_k = \mathbf{Y}_{k-1} + \mathbf{H}_k^T \Sigma_\nu^{-1} \mathbf{H}_k \tag{2.29}$$

For a single bearing measurement to a single target, $\Sigma_\nu = \sigma_\nu^2$. Expanding gives

$$\mathbf{Y}_k = \mathbf{Y}_{k-1} + \frac{1}{r_k^2 \sigma_\nu^2} \begin{bmatrix} \sin^2\gamma_k & -\sin\gamma_k\cos\gamma_k \\ -\sin\gamma_k\cos\gamma_k & \cos^2\gamma_k \end{bmatrix} \tag{2.30}$$

Writing Equation 2.30 as $\mathbf{Y}_k = \mathbf{Y}_{k-1} + \Delta\mathbf{Y}_k$, the information gained about a target over a trajectory can be expressed as

$$\mathbf{Y} = \mathbf{Y}_0 + \sum_{k=1}^{K} \Delta\mathbf{Y}_k \tag{2.31}$$

This presentation of the Fisher Information Matrix is applicable when the target is stationary and the FIM itself is representative of the information gained about the position of the target. In the moving target case, the target motion causes a loss of information that can be accounted for in the FIM by including terms related to the uncertainty in the target motion.

2.6 Uncertainty Minimization Using the Fisher Information Matrix

To see how how the uncertainty associated with a target position estimation can be minimized using the Fisher Information Matrix, consider the following two simplified examples. In the first example, the case where the observer vehicle is flying a constant radius circle around the target is analyzed. The vehicle is initially aligned with the x-axis, no sensor measurements have been taken, and the initial uncertainty is given by

$$\mathbf{P}_0 = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_r^2 \end{bmatrix} \tag{2.32}$$

The information is the inverse of the uncertainty

$$\mathbf{Y}_0 = \begin{bmatrix} \sigma_r^{-2} & 0 \\ 0 & \sigma_r^{-2} \end{bmatrix} \tag{2.33}$$

This example problem is demonstrated in Figure 2.5. In these simplified uncertainty minimization problems, the goal is to locate the angle around the circle the observer vehicle needs to travel to minimize the uncertainty in the estimated position (or maximize the information gain) through just a single measurement.

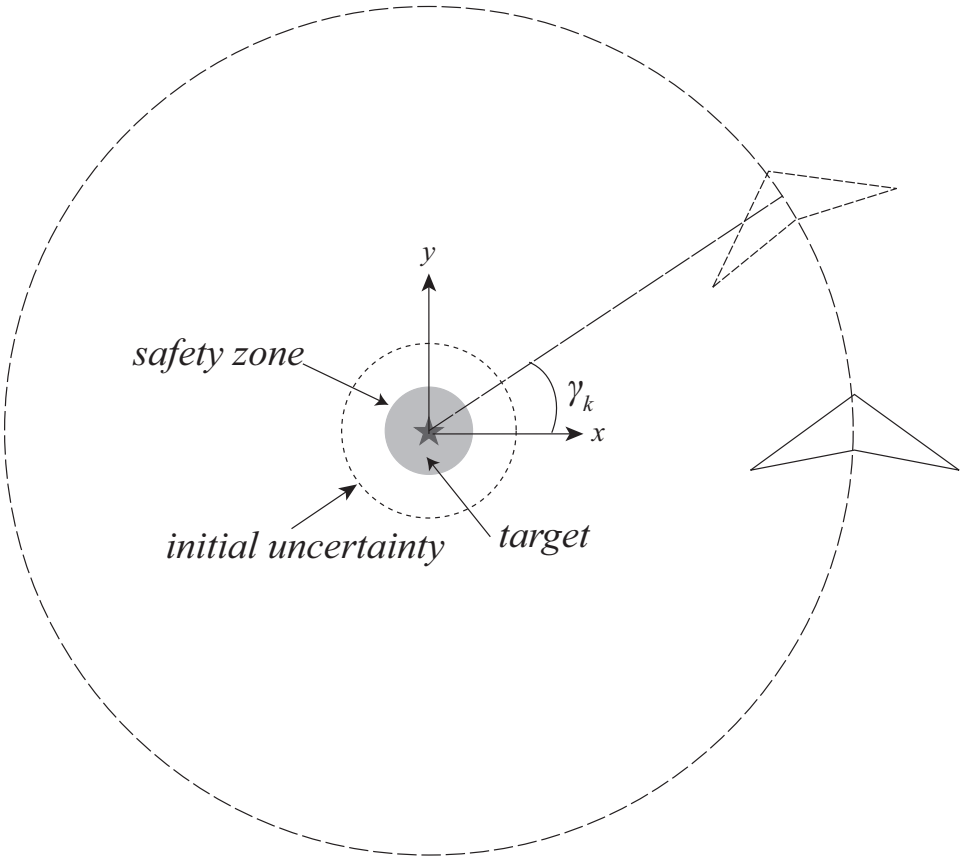


Figure 2.5. First example problem setup for using the FIM to minimize target location uncertainty. Here the initial uncertainty in target position is equal along the x and y axes.

Using Equation 2.31, and making the appropriate substitutions

$$\mathbf{Y} = \begin{bmatrix} \sigma_r^{-2} & 0 \\ 0 & \sigma_r^{-2} \end{bmatrix} + \frac{1}{r_k^2 \sigma_\nu^2} \begin{bmatrix} \sin^2 \gamma_k & -\sin \gamma_k \cos \gamma_k \\ -\sin \gamma_k \cos \gamma_k & \cos^2 \gamma_k \end{bmatrix} \tag{2.34}$$

Allowing σ_ν and r_k to go to unity for simplification

$$\mathbf{Y} = \begin{bmatrix} \sigma_r^{-2} + \sin^2 \gamma_k & -\sin \gamma_k \cos \gamma_k \\ -\sin \gamma_k \cos \gamma_k & \sigma_r^{-2} + \cos^2 \gamma_k \end{bmatrix} \tag{2.35}$$

For an optimization problem, the Fisher Information Matrix must be converted to a scalar value. Any number of matrix operations are acceptable, for this example, the natural log of the determinant of the inverse of the FIM is used. The optimization

is thus a minimization.

$$J = \log\det \mathbf{Y}^{-1} \tag{2.36}$$

$$J = -\log\det \mathbf{Y} \tag{2.37}$$

After taking the determinant

$$\det \mathbf{Y} = \sigma_r^{-4} + \sigma_r^{-2} \tag{2.38}$$

it is easy to see that there is no relation on the angle between the starting position of the vehicle and optimal location of the next measurement when the initial uncertainty is the same in both the x- and y- axes of the problem.

If, however, the initial uncertainty along the x-axis of the problem is significantly larger than the initial uncertainty along the y-axis of the problem, i.e.

$$\mathbf{P}_0 = \begin{bmatrix} (2\sigma_r)^2 & 0 \\ 0 & \sigma_r^2 \end{bmatrix} \tag{2.39}$$

Then the initial information is greater along the y-axis of the problem than along the x-axis

$$\mathbf{Y}_0 = \begin{bmatrix} (2\sigma_r)^{-2} & 0 \\ 0 & \sigma_r^{-2} \end{bmatrix} \tag{2.40}$$

there does exist a non-trivial optimal angle between the starting position of the vehicle and the location of the next measurement. This updated situation is given in Figure 2.6. Again, using Equation 2.31 as the starting point, and making the same simplifications as above, but using the new initial uncertainty

$$\mathbf{Y} = \begin{bmatrix} (4\sigma_r^2)^{-1} + \sin^2\gamma_k & -\sin\gamma_k\cos\gamma_k \\ -\sin\gamma_k\cos\gamma_k & \sigma_r^{-2} + \cos^2\gamma_k \end{bmatrix} \tag{2.41}$$

Using the same operations to find a scalar information cost

$$-\log\det \mathbf{Y} = -\log\left(\frac{1}{4\sigma_r^4} + \frac{\cos\gamma_k}{4\sigma_r^2} + \frac{\sin\gamma_k}{\sigma_r^2}\right) \tag{2.42}$$

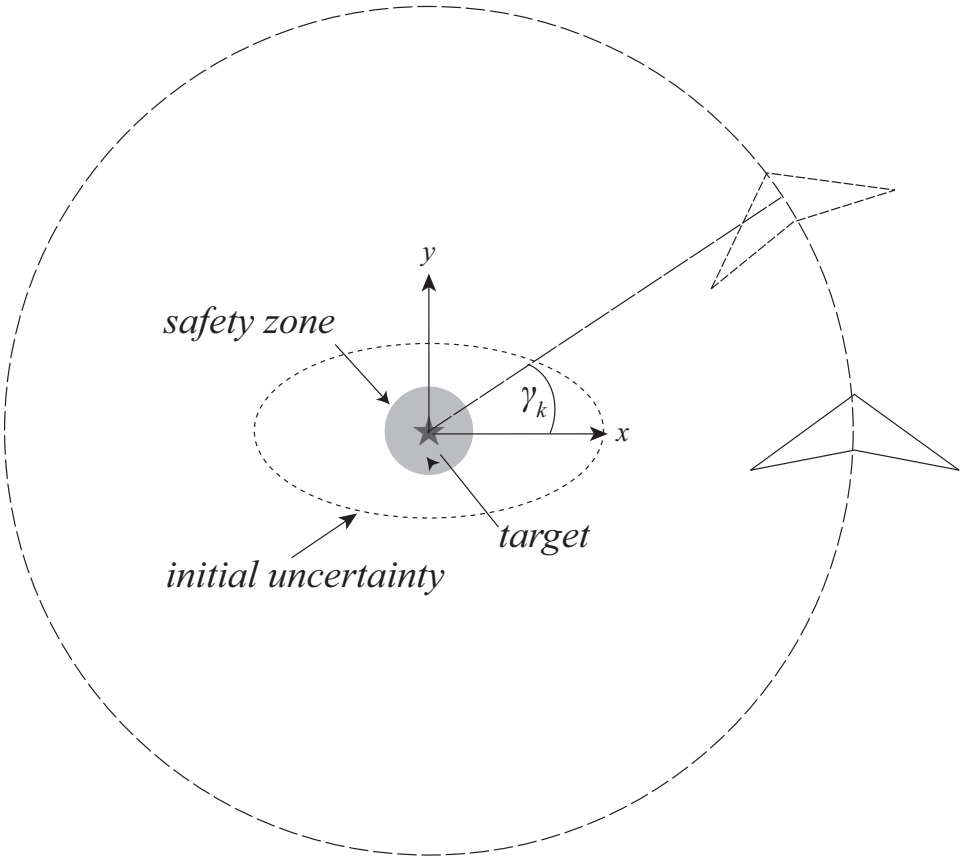


Figure 2.6. Second example problem setup for using the FIM to minimize target location uncertainty. Here the initial uncertainty in target position is greatest along the x axis.

The optimal angle, γ_{opt}, can be found by solving

$$\frac{\partial\left(-\log\det\mathbf{Y}\right)}{\partial\gamma_k}=0 \tag{2.43}$$

or by plotting $-\log\det\mathbf{Y}$ against γ_k as shown in Figure 2.7. The plot shows, and the differential equation confirms, the optimal angle between the initial position of the vehicle (aligned with the x-axis) and the location of the first measurement is $\pm\frac{\pi}{2}$. Intuitively, this result shows the maximum information will be gained when the vehicle takes measurements at orthogonal angles.

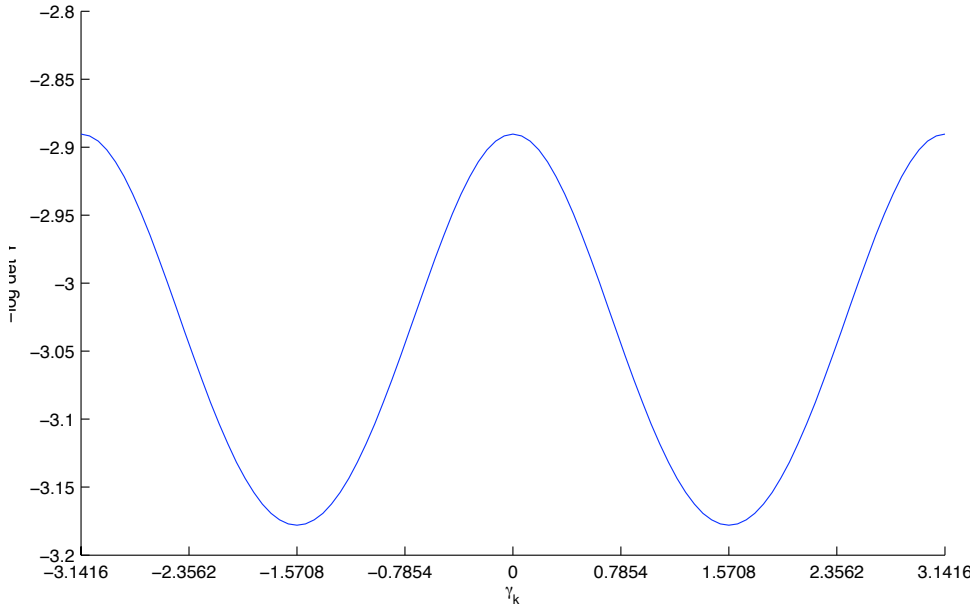


Figure 2.7. Plot of $-\log\det\mathbf{Y}$ versus γ_k for the second example of using the FIM to minimize target location uncertainty

2.7 Summary

This chapter has presented an overview of the UAV performing a surveillance and target tracking task as considered in this thesis. Because the UAV only has a monocular camera onboard, the path flown by the UAV greatly affects the uncertainty in the estimated position of the target being tracked. The Fisher Information Matrix (FIM) provides a convenient way to calculate the information gained by flying a unique path and is directly related to the uncertainty in the target position estimation. In the most simplified form, the target tracking problem can be defined as

$$\text{minimize} \quad -\log\det\mathbf{Y}_k \tag{2.44}$$

$$\text{subject to} \quad \text{trajectory constraints} \tag{2.45}$$

where $\mathbf{Y}_k$ is the Fisher Information Matrix at the end of the trajectory.

In the simplified form, this problem can be solved analytically, however, when complications like safety zones, sensor field of view limits, and more involved vehicle

models are considered, the problem must be solved using numerical optimization. This problem becomes even more difficult when high frame-rate sensors and longer planning horizons are considered, e.g. a 15 measurement per second sensor and a 15 second planning horizon yields 225 measurements that must be optimized. With the limited computational power available on small UAVs, the optimization problem generally cannot be solved in real-time.

Chapter 3

A Table of Optimal Trajectories

3.1 Introduction

As stated in Chapter 2, real time solution of the trajectory optimization problem is generally intractable on the computing hardware likely to be available on a μAV. To avoid this problem, a set of trajectories for representative target locations are pre-computed and then stored in a lookup table. To make the resulting table of trajectories generally applicable to different sensors and vehicles, the problem is first non-dimensionalized using sensor parameters and the observer vehicle speed. This chapter first discusses the non-dimensionalization of the problem in Section 3.2 and then discusses two methods of parameterizing the optimal trajectories. The first parameterization represents each trajectory as a sequence of turn-rate commands; the second parameterization represents each trajectory as a sequence of GPS-like waypoints. Both parameterizations are described in Section 3.3. Solution methods for the trajectory optimization problems defined by the two parameterizations are discussed in Section 3.4. A few comments about using the generated trajectories on vehicles with different flight characteristics and sensor properties are provided in Section 3.5. Finally, Section 3.6 enumerates the steps to perform in a single target localization task. In Chapter 4, simulation results showing the localization ability of the trajectory tables will be given.

3.2 Non-Dimensionalization of the Problem

For this work, the problem is non-dimensionalized to make the solution - the generated lookup tables of trajectories - more general. By non-dimensionalizing with respect to sensor parameters, two vehicles with different speeds or maximum turn rates can use the same lookup table given the same or similar sensor packages. In the waypoint-implementation, external influences like wind, can be compensated for by the trajectory-following controller, allowing a single lookup table to be used in almost all conditions for a given sensor package.

3.2.1 Kinematics Model

To non-dimensionalize the problem, vehicle kinematics are scaled with respect to sensor parameters. In this work, distances are scaled by sensor range R and time is scaled by the sensor frame sample time T_f:

$$\dot{\tilde{x}}_v = \frac{T_f}{R} v \cos\psi_v \tag{3.1}$$

$$\dot{\tilde{y}}_v = \frac{T_f}{R} v \sin\psi_v \tag{3.2}$$

$$\dot{\tilde{\psi}}_v = T_f u \tag{3.3}$$

3.2.1.1 Discrete Time Kinematics Model

A second order approximation is used to generate a discrete time model for vehicle kinematics with sample time T_s. The integration time is also scaled by the sensor frame sample time (i.e. $\tilde{\Delta}t = T_s/T_f$):

$$\tilde{\mathbf{x}}_{k+1} = \tilde{\mathbf{x}}_k + \frac{T_s}{T_f} \begin{bmatrix} \frac{T_f}{R} v \cos\psi \\ \frac{T_f}{R} v \sin\psi \\ T_f u \end{bmatrix} + \frac{1}{2}\frac{T_s^2}{T_f^2} \begin{bmatrix} -\frac{T_f^2}{R} v u \sin\psi \\ \frac{T_f^2}{R} v u \cos\psi \\ 0 \end{bmatrix} \tag{3.4}$$

The non-dimensionalized discrete time vehicle kinematics are therefore

$$\tilde{\mathbf{x}}_{k+1} = \tilde{\mathbf{x}}_k + T_s \begin{bmatrix} \frac{v}{R}\cos\psi \\ \frac{v}{R}\sin\psi \\ u \end{bmatrix} + \frac{T_s^2}{2} \begin{bmatrix} -\frac{v}{R}u\sin\psi \\ \frac{v}{R}u\cos\psi \\ 0 \end{bmatrix} \tag{3.5}$$

3.2.2 Fisher Information Matrix

The Fisher Information Matrix is based on the sensor measurement model, which also must be non-dimensionalized. Non-dimensionalizing the sensor model with respect to the sensor range gives

$$\tilde{\mathbf{H}}_k = \begin{bmatrix} -\frac{R\sin\gamma_k}{r_k} & \frac{R\cos\gamma_k}{r_k} \end{bmatrix} \tag{3.6}$$

The non-dimensionalized information gained about the target from a single measurement can now be expressed as

$$\tilde{\mathbf{Y}}_k = \tilde{\mathbf{Y}}_{k-1} + \tilde{\mathbf{H}}_k^T \Sigma_\nu^{-1} \tilde{\mathbf{H}}_k \tag{3.7}$$

After non-dimensionalization, Equation 2.30 becomes

$$\tilde{\mathbf{Y}}_k = \tilde{\mathbf{Y}}_{k-1} + \frac{R^2}{r_k^2\sigma_\nu^2} \begin{bmatrix} \sin^2\gamma_k & -\sin\gamma_k\cos\gamma_k \\ -\sin\gamma_k\cos\gamma_k & \cos^2\gamma_k \end{bmatrix} \tag{3.8}$$

Again, assuming a stationary target and writing Equation 3.8 as $\tilde{\mathbf{Y}}_k = \tilde{\mathbf{Y}}_{k-1} + \Delta\tilde{\mathbf{Y}}_k$, and the non-dimensional information gained about a target over a trajectory can be expressed as

$$\tilde{\mathbf{Y}} = \tilde{\mathbf{Y}}_0 + \sum_{k=1}^{K} \Delta\tilde{\mathbf{Y}}_k \tag{3.9}$$

The Fisher Information Matrix will later be used in the cost function of the optimization problem.

3.3 Trajectory Parameterization

In this section, the two methods of trajectory parameterization, turn-rate commands and GPS-like waypoints, are discussed and the resulting optimization problems are defined. The first parameterization was simply the optimization of a sequence of turn-rate commands. This approach, however, has several drawbacks. First, the number of turn-rate commands for a given trajectory varied with the initial distance from the observer vehicle to the target, i.e. a shorter trajectory had fewer turn-rate commands. This required either unused space to be reserved in the lookup table, or complicated memory access algorithms to be employed. On the other hand, a longer trajectory may have more than 200 turn-rate commands, which requires significant computational time even on a workstation-class computer. This led to the development of a second parameterization using waypoints. The waypoint parameterization allowed for easier optimization and storage as every trajectory relied on ten waypoints. The waypoint trajectories are also easier to non-dimensionalize and allow for easier implementation of the optimization constraints. Potentially the greatest benefit of this parameterization is that the parameterization is independent of external disturbances, such as wind.

To generate a target localization table for both the turn-rate- and waypoint-based implementations, the sensor field of view was uniformly discretized in the radial and angular directions: i.e. a 10×10 polar grid was defined over the sensor field of view and a nominal target location was defined at each grid point. This discretization is shown in Figure 3.1. In this figure, there are N and M divisions in the radial and axial directions, respectively. The optimal trajectory is generated for potential target located at the centroid of each cell. A schematic of the target localization tables showing nominal target locations and three sample trajectories associated with three nominal locations is shown in Figure 3.2. The figure shows nominal target locations and three representative trajectories for both the turn-rate- and waypoint-based implementations.

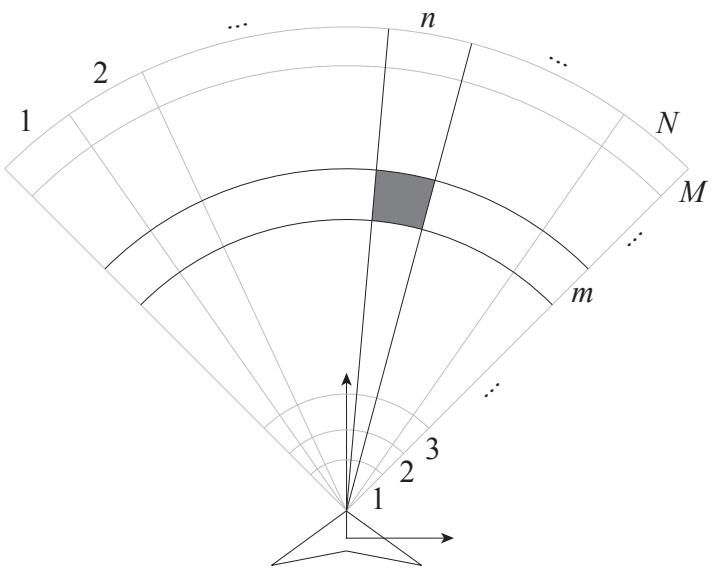
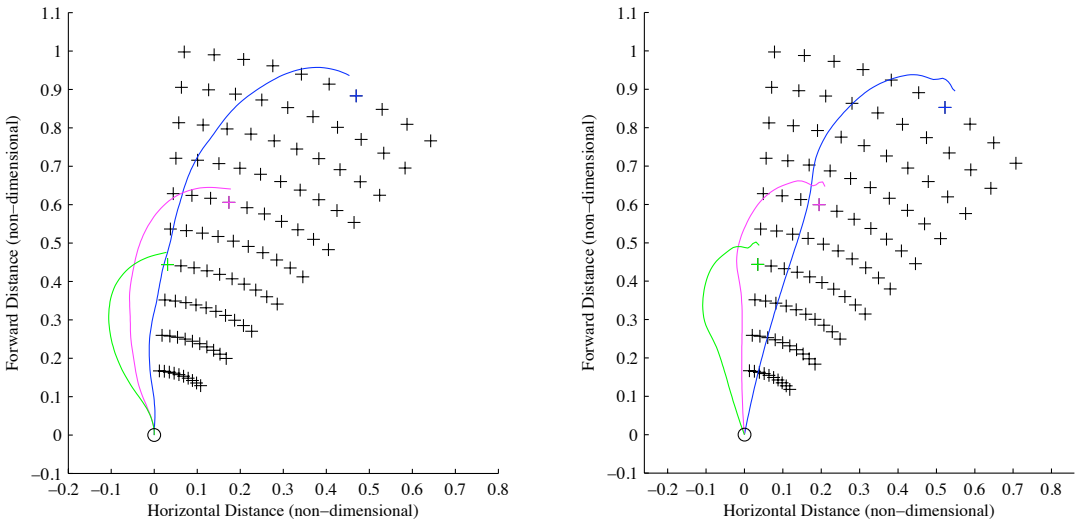


Figure 3.1. Discretization of the sensor field of view.



(a) Turn-rate-based implementation

(b) Waypoint-based implementation

Figure 3.2. Sample trajectory table target locations and sample trajectories for turn-rate- and waypoint-based implementations.

3.3.1 Turn-Rate Trajectory Parameterization

For the turn-rate-based implementation, each trajectory consists of a sequence of turn rate commands

$$\mathbf{u}_{mn} = \begin{bmatrix} u_{mn,1} & u_{mn,2} & \dots & u_{mn,N} \end{bmatrix} \tag{3.10}$$

where N is a planning horizon that extends from $t = t_0$ until the time the risk zone is encountered. The duration of each turn rate input is equal to the kinematics update rate of the vehicle.

3.3.2 Turn-Rate Trajectory Optimization

For the turn-rate-based implementation, the trajectory generation problem for target (m, n) can now be summarized as

$$\begin{aligned} \text{minimize} \quad & J(\mathbf{u}_{mn}) && (3.11) \\ \text{subject to} \quad & \tilde{\mathbf{x}}_{k+1} = f(\tilde{\mathbf{x}}_k, u_{k,mn}) && (3.12) \\ & u_{min} \leq u_{k,mn} \leq u_{max} && (3.13) \end{aligned}$$

where the cost function J is disclosed in Section 3.4; vehicle kinematics are given by Equation 3.5; and inputs u_k are limited by turn rate constraints.

3.3.3 Waypoint Trajectory Parameterization

The trajectories exist in non-dimensional space as a sequence of ten waypoints, $\tilde{\mathbf{X}}_{mn} = [\tilde{\mathbf{x}}_{mn,1}, \dots, \tilde{\mathbf{x}}_{mn,N}]$. Each waypoint, $\tilde{\mathbf{x}}_{mn,N}$, consists of an angle. θ, and a distance, r, to the waypoint relative to the nominal target location. Thus, relative to the target location, each waypoint exists at Cartesian coordinates

$$\begin{aligned} x_n &= r_n \cos\theta_n && (3.14) \\ y_n &= r_n \sin\theta_n && (3.15) \end{aligned}$$

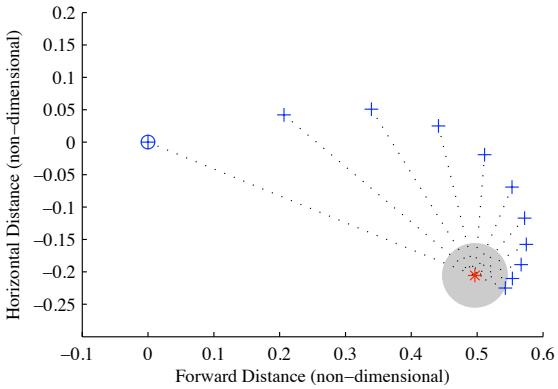


Figure 3.3. Waypoints are defined by a distance r from the target location at fixed angles.

where

$$\theta_n = [0 \ldots \pi] \tag{3.16}$$

$$r_n = r(\theta) \tag{3.17}$$

For this work, each waypoint exists at a fixed angle from the target location and the distance of the waypoint from the target is allowed to vary, subject to constraints. This is shown in Figure 3.3.

Each trajectory consists of a sequence of ten waypoints in non-dimensional space. To compute a path in physical space the way-points are first dimensionalized by multiplying by sensor range

$$\mathbf{X}_{mn} = R\tilde{\mathbf{X}}_{mn} = \left[\begin{array}{cccc} \mathbf{x_{mn,1}} & \mathbf{x_{mn,2}} & \ldots & \mathbf{x_{mn,10}} \end{array} \right] \tag{3.18}$$

Finally a spline is used to compute the complete path (Figure 3.4).

3.3.4 Waypoint Trajectory Optimization

For the waypoint-based implementation, the trajectory generation problem for target (m, n) can be summarized as

$$\text{minimize} \qquad J(\tilde{\mathbf{X}}_{mn}) \tag{3.19}$$

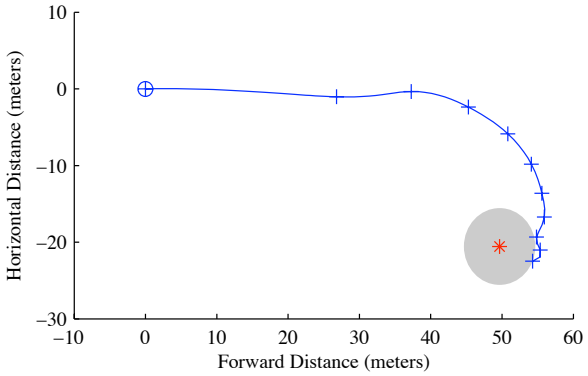


Figure 3.4. Ten waypoints are interpolated and dimensionalized to form a complete path.

$$\text{subject to} \qquad \tilde{\mathbf{x}}_k = f(\tilde{\mathbf{X}}_{mn}, T_s \frac{v}{R}, k) \tag{3.20}$$

$$\frac{R\dot{\psi}_{min}}{v} \leq \tilde{\kappa}_k \leq \frac{R\dot{\psi}_{max}}{v} \tag{3.21}$$

where, again, the cost function J is disclosed in Section 3.4; the vehicle path is computed using the interpolating function f and $\tilde{\kappa}_k$ is the curvature of the path (non-dimensionalized using sensor range), constrained by vehicle turn rate limits.

3.4 The Trajectory Optimization Problem

3.4.1 Cost Function

The cost function in this research is implementation dependent. Factors including the number of dimensions of the vector optimization problem and type of optimization method used greatly influence the components of the cost function.

For the turn-rate-based implementation, the cost function includes a term related to the uncertainty in the target state estimate, a term to keep the target within the field of view, and a term defining the safety or risk zone cost

$$J = (w_{info} + w_{fov})\, J_{info} + w_{safe} J_{safe} \tag{3.22}$$

The waypoint-based implementation cost function is similar, but removes the risk zone cost term as the optimizer implements the risk zone as a constraint

$$J = (w_{info} + w_{fov})\, J_{info} \tag{3.23}$$

3.4.1.1 Information Cost

The information cost is computed using the Fisher Information Matrix. Note that the target is assumed to be stationary. A scalar value for information cost is then given by

$$J_{info} = \log\det \tilde{\mathbf{Y}}^{-1} \tag{3.24}$$

$$J_{info} = -\log\det \tilde{\mathbf{Y}} \tag{3.25}$$

It is important to note that minimizing $\tilde{\mathbf{Y}}^{-1}$ is mathematically equivalent to maximizing $\tilde{\mathbf{Y}}$, or information about the target. Alternatively, minimizing $\tilde{\mathbf{Y}}^{-1}$ is a way of minimizing the uncertainty in the target state estimate.

3.4.1.2 Field of View Weight

To keep the target in the field of view, a weight is computed based on the bearing to the target, γ_k

$$w_{fov} = \left(\frac{\gamma_k}{\gamma_{max}}\right)^4 \tag{3.26}$$

While the field of view is also accounted for in the information cost, this term assists the optimization routine in finding a valid solution. Initial optimizations that did not include this weight resulted in poor, or no, convergence of the optimization.

3.4.1.3 Safety Cost

A safety zone is included around the target to ensure that sufficient stand-off distance is maintained to prevent collision with the target or detection of the vehicle. A high-order polynomial function is used to ensure high cost close to the

target and low cost far away.

$$J_{safe} = \sum_{k=1}^{K} \left(\frac{r_{safe}}{r_k} \right)^8 \tag{3.27}$$

3.4.2 Solution for Turn-Rate Parameterization

In principle, any optimization method could be used to generate a solution to the optimization problem defined by Equation 3.13. In this work the trajectory is discretized into K steps with constant turn rate in each step. The resulting vector optimization problem is solved using MATLAB's `fmincon` function. The information and safety zone weights (i.e. w_{info} and w_{safe}) were set to 1. To simplify the problem, a starting path with the observer vehicle flying straight at the target location is used to initialize the optimizer for each series of trajectories.

3.4.3 Solution for Waypoint Parameterization

To calculate solutions to the vector optimization problem defined by Equation 3.21, a minimization routine using a modified version of a gradient-based line search method was programmed and used. A Newton-Raphson method was not able to be used because the problem could not be proved to be strictly convex analytically. Thus, using the Hessian in the calculation of the search direction could cause the optimizer to search in the wrong direction.

The optimizer is able to handle constrained optimization problems, such as the one presented in this work. If the step in the search direction yields a minimization point that violates either a constraint, the step size is reduced by an optimization parameter, β. A new minimization point is then calculated using the reduced step size and the process repeats if necessary until a point is found that no longer violates the constraints of the optimization problem.

To improve the generated trajectories, a pseudo-tangency constraint is imposed by fixing the distance of the last two waypoints to the distance specified by a logarithmic spiral fitting the fixed final waypoint as well as the initial location of the observer vehicle.

Again, because of the nature of the line search method, a valid guess of a starting path is necessary to find a solution. To quickly generate valid paths, the optimizer is initialized with a path defined by a logarithmic spiral from the initial observer position to the target position. Strictly speaking, the path which results from the optimization is a local optimum near the initial guess.

3.4.4 Table Implementation

After solving the optimization problem for each of the nominal target locations, a set of trajectories is created. These trajectories are then stored in a table to be used on a variety of vehicles. For ease of implementation, the storage of the trajectories for this research has been in a two-dimensional table. Each trajectory is indexed by both a non-dimensional range and the bearing to each nominal target location. A trajectory can then be selected by matching as closely as possible the non-dimensional range and bearing to the actual target to those available in the lookup table.

3.4.4.1 Reflection Symmetry of the Table

For both the turn-rate- and waypoint-based implementations, only half of the sensor field of view is covered by the trajectory tables shown in Figure 3.2. Early results from the optimization routines showed near-perfect reflection symmetry about the longitudinal axis of the vehicle. Thus, the size of the lookup table can be made 50% smaller by exploiting this reflection symmetry. Targets in the left half plane of the observer vehicle are localized with a reflected trajectory from the lookup table.

3.5 Observer Vehicle and Sensor Variations

For the waypoint-based implementation, a trajectory following controller allows vehicles of different speeds to use the single trajectory stored in the lookup table to best localize a target. Since the table was generated by non-dimensionalizing the target localization problem using sensor range R, sensor update period T_f, and vehicle speed V, intuition suggests that trajectories stored in the lookup table

are optimal for any vehicle and sensor package that has the same "characteristic number" as the generated trajectory table. This "characteristic number" relates the sensor range and sample time, R and T_f to the observer vehicle speed, v.

$$CN = \frac{(R/T_f)}{v} \tag{3.28}$$

Intuitively, CN is a measure of measurements of the target that can be obtained before the target is reached. Intuition also suggests, and simulations will later show, that two vehicles which share the same characteristic number can use the same lookup table for selecting optimal target localization trajectories. Simulations will also show that good, though sub-optimal, trajectories can be still be used for vehicles with differing characteristic numbers.

3.6 Target Localization

This chapter has discussed the development of a lookup table to allow fast trajectory generation or selection even on the limited computing power available of a μAV of autonomous submunition. Target localization using the lookup table follows three steps:

1. An initial target location is passed to the table. If it is within the field of view, the trajectory associated with the closed cell centroid is selected. If the initial target location is outside the field of view the vehicle is commanded to turn towards the initial given position of the target and fly until the target is seen.

2. The trajectory, $\mathbf{u}_{mn}$ for the turn-rate-based implementation or $\mathbf{X}_{mn}$ for the waypoint-based implementation, is followed in open loop over a control horizon T_c. A target state estimate is computed using a Sigma Point Kalman Filter[35, 36]. The control horizon is dependent on the initial distance between the vehicle and the target at the time of the first camera measurement. When the control horizon is reached, a new trajectory is selected based on the current estimate of the target position.

3. The task is complete when the vehicle reaches the risk zone or the vehicle

passes the target. The next target in the sequence is selected, and the process repeats for all given targets.

3.7 Summary

This chapter has presented a method for computing and storing a set of trajectories for representative target locations to use in a target localization task for small- and micro- UAVs. This method is needed because current computational power available on these vehicles is not powerful enough to optimize the trajectory in real-time. To make the resulting table of trajectories generally applicable to different sensors and vehicles, the problem was non-dimensionalized using sensor parameters and the observer vehicle speed. Two parameterizations, using turn-rate commands and waypoints, were then described and solution methods for the trajectory optimization problems defined by the two parameterizations were discussed. Lastly, the steps to perform a single-target localization task were given. The next chapter will provide simulation results showing the localization ability of the trajectory tables.

Chapter 4

Pre-Computed Trajectories

4.1 Introduction

This chapter will present simulation results of the target localization problem using the method of pre-computed trajectories stored in lookup tables. Simulation results using trajectories with the turn-rate parameterization are presented first in Section 4.2. Using the turn-rate parameterization a study on lookup table size, i.e. the number of nominal target locations, is presented in Section 4.2.2. This study was performed to evaluate the effect of table density on target localization accuracy and was conducted using Monte Carlo simulations with 500 runs each. This study found the surprising result that a lookup table with 25 nominal target localizations evenly distributed throughout the sensor field of view performed better in target localization tasks than a tables with 400 or 100 nominal targets with similar distributions. This is followed in Section 4.2.3 by a sample sequential target localization task using the turn-rate parameterized table with five sequential targets. The waypoint-parameterized trajectories are then presented in Section 4.3. Comparison of the waypoint-parameterized trajectories from the lookup table with direct optimized trajectories is shown in this section as well. As suggested in Chapter 3, observers that share the same "characteristic number" but have different observer vehicle and sensor parameters can use the generate table directly. Results supporting this claim are presented in Section 4.3.2. Comparison of table trajectories with direct optimized trajectories for observer vehicles with different "characteristic numbers" is also presented in this section. A Monte Carlo

simulation of a single target task to determine the localization performance is given in Section 4.3.3. Using these simulations, a study of the ratio of information cost and information gain for table trajectories versus direct optimized trajectories is presented. In Section 4.3.4 a comparison of computation power required for table lookup trajectories and direct optimized trajectories is performed. The results from the localization performance and required computational power for the waypoint parameterization show that good localization performance can be achieved using significantly less computational power than an online optimization. A sequential target task problem and simulation results are presented in Section 4.3.5. Finally, a statement of the ability for fast adaptation using the table lookup trajectories is made and simulation results showing the importance of adaptation are given in Section 4.4.

4.2 Target Localization Using Turn-Rate Parameterization

This section discusses simulation results for the turn-rate parameterization. First, target localization using the trajectory table is illustrated using a sample single-target task. Next, a method for table compression is discussed. Finally, the use of the table for localizing multiple targets is illustrated.

4.2.1 Using the Table

A sequence of images showing localization of a single target is given in Figure 4.1. In the figure, true target location shown with $*$, estimated target location shown with $+$ and associated error ellipse. The green line in the figure shows the current path selected from the trajectory table, blue line shows the path flown. The procedure described in Section 3.6 is used to complete the target localization task. The target is initially within the field of view of the sensor, so a trajectory can be selected immediately. The vehicle follows the trajectory while estimating target state for a control horizon $T_c = 2$ seconds, hence 4.1 seconds into the flight, shown in subfigure (b), the vehicle is following its third trajectory. The vehicle continues to fly, selecting new trajectories as the control horizon is reached. The target

localization task is concluded as the vehicle passes the target. Recall that sensor noise is assumed to be Gaussian, here $\sigma_\nu = 0.0175$ rad (i.e. 1°) was used.

By the end of the task, the target has been localized to an accuracy of 0.42 meters and 0.24 seconds of CPU time was spent performing trajectory selection. As a comparison, direct optimization of the trajectory resulted in a localization accuracy of 0.26 meters, but required 289 seconds to optimize the trajectory with this parameterization. CPU times for this comparison were measured using a desktop workstation with an Intel 2.4 GHz processor.

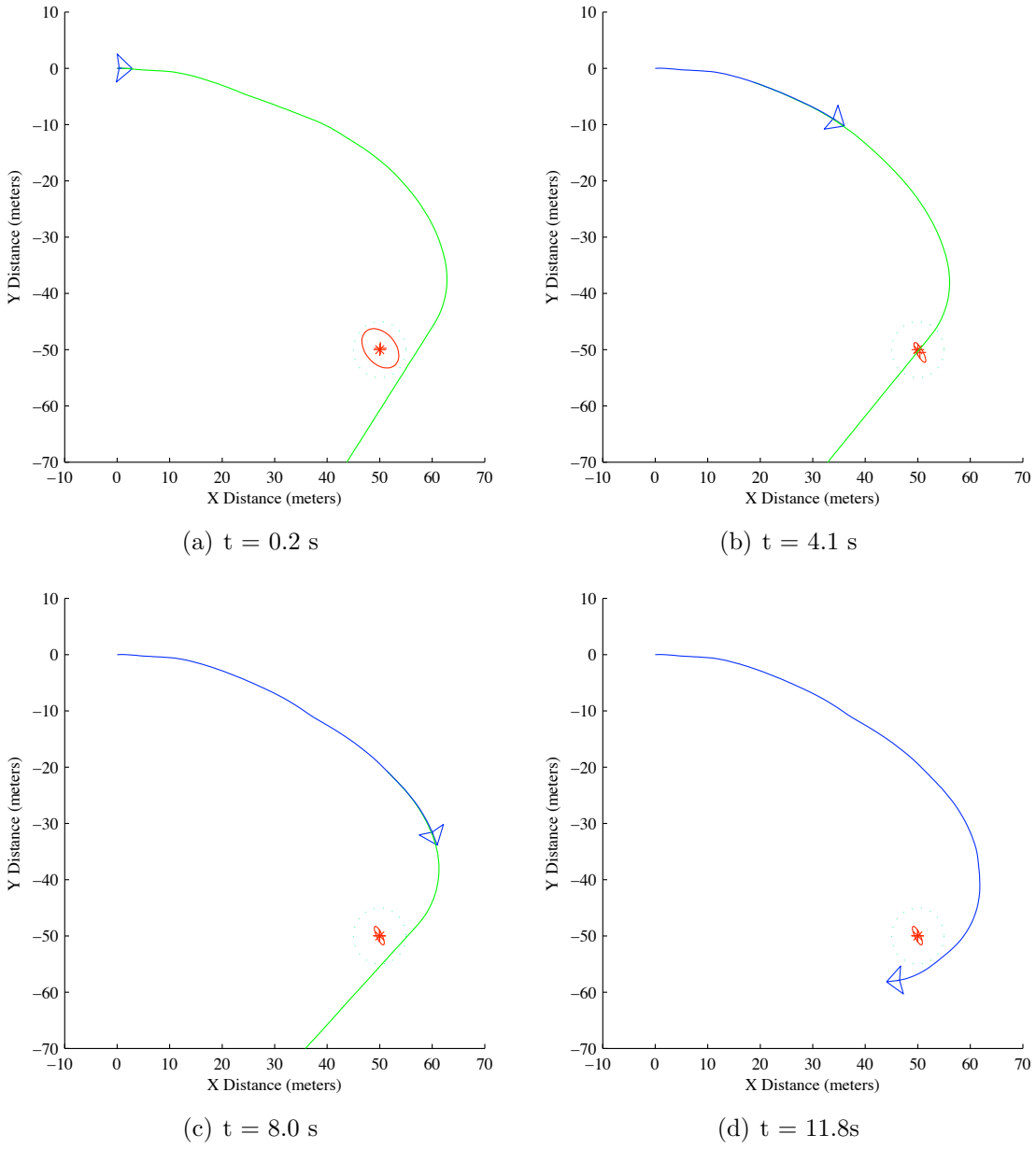
4.2.2 Compression by Decimation

Because the lookup table is designed to be used on vehicles that are very small and might even be destroyed when the mission is complete (e.g. autonomous munitions), minimizing the amount of memory required to store the table is desired. In addition to compressing the table through reflective symmetry, density of nominal target locations in the trajectory table has a large effect on the memory required to store the table.

A simple study showing the effects of the number of trajectories stored in the lookup table on the localization error was performed. Two lookup tables, one with 100 trajectories and one with 400 trajectories, were created directly. A third lookup table, with 25 trajectories, was created by post-processing the 100 trajectory lookup table: every second row and every second column was deleted from the table.[1] A Monte Carlo simulation was then performed for each of the three lookup tables. Random target locations within the sensor field of view were chosen and the vehicle flew the target localization algorithm as described in Section 3.6. Target state estimation was performed on-line.

The results of the Monte Carlo simulation, with 500 runs for each lookup table, are given in Table 4.1. The table shows values for the determinant, trace, and maximum eigenvalue of the covariance $\mathbf{\Sigma}$ of the *true target localization error* over the 500 run Monte Carlo simulation. The results show the 5×5 trajectory lookup

[1] This is compression by decimation, and is an easy but very crude method of data compression. More sophisticated approaches which compute average trajectories over neighboring nominal target positions may allow greater compression with less loss of accuracy. Incidentally, the term decimation originally referred to the punishment of a garrison of Roman legionnaires convicted of cowardice in battle: the soldiers were lined up and every tenth soldier was executed.



(a) t = 0.2 s

(b) t = 4.1 s

(c) t = 8.0 s

(d) t = 11.8s

Figure 4.1. Snapshots of a single target localization run using the turn-rate-parameterized trajectory table.

table actually performed the best job localizing the random single targets, with the 10×10 and 20×20 trajectory tables performing slightly worse. A scatter plot of the target localization error for all three Monte Carlo simulations is shown in Figure 4.2. Finally, a plot of the weighted error for each simulation run for each trajectory lookup table is given in Figure 4.3. The weighted, or normalized, error

Table 4.1. Effect of trajectory table compression on target localization accuracy.

table size	$\det \mathbf{\Sigma}$	$\mathrm{Tr}\mathbf{\Sigma}$	$\max(\mathrm{eig}\mathbf{\Sigma})$
5×5	0.0186	0.4330	0.3846
10×10	0.0254	0.4818	0.4217
20×20	0.0281	0.4815	0.4136

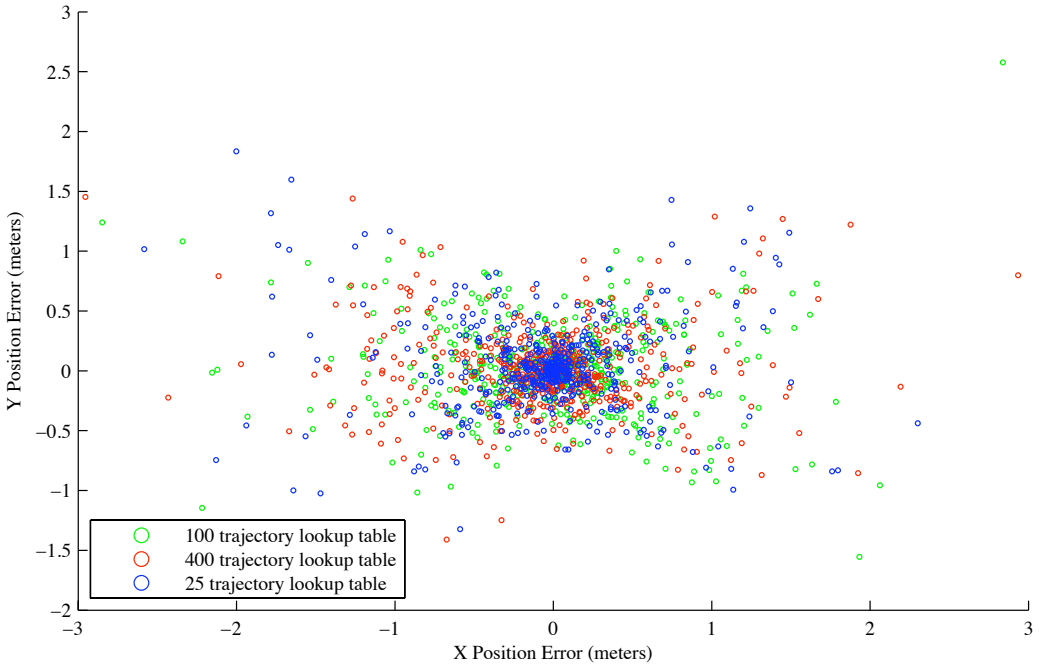


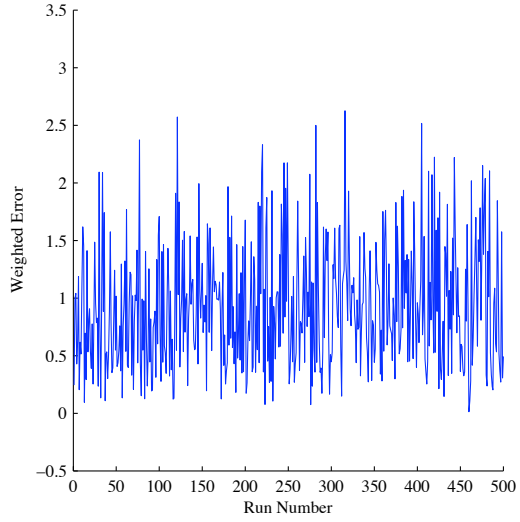
Figure 4.2. Scatter plot of target localization error for 500 run Monte Carlo simulation.

e_i for each run is computed from

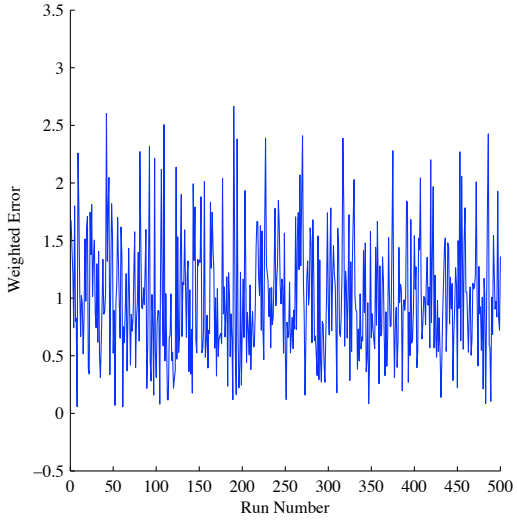
$$e_i = \sqrt{\frac{1}{2}(\mathbf{x}_i - \hat{\mathbf{x}}_i)^T \mathbf{P}^{-1} (\mathbf{x}_i - \hat{\mathbf{x}}_i)} \tag{4.1}$$

where $\mathbf{x}_i$ is the true target position for run i, $\hat{\mathbf{x}}_i$ is the estimated target position at the end of the run and $\mathbf{P}$ is the estimated covariance. The factor $1/2$ accounts for the dimension of the estimation problem (2D target position estimation), and for a consistent estimator the average value of e_i is unity. This is the case for all three trajectory lookup tables.

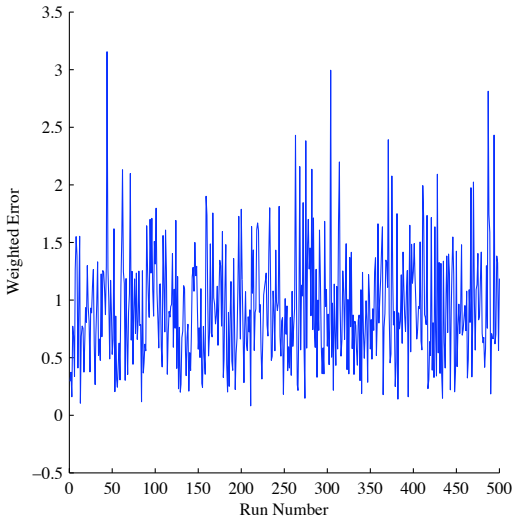
These results suggest that using the turn-rate parameterization there is very little difference between the optimal target localization trajectory and a trajectory



(a) 5×5 lookup table



(b) 10×10 lookup table



(c) 20×20 lookup table

Figure 4.3. Weighted (i.e. normalized) error for 500 simulations with different size lookup tables.

that is sub-optimal but still "pretty good". In the 5×5 trajectory table the actual target position will generally be farther away from the nominal target position used to generate the trajectory than for either the 10×10 or 20×20 tables. It can thus be inferred that the cost function used in this turn-rate parameterization is very "flat" in the vicinity of the optimal value.

4.2.3 Sequential Target Localization

The trajectory table can also be used in a sequential target localization task. The order of targets to be visited and an initial estimate of position is determined by a human operator or higher level planner and sent to the vehicle. The vehicle chooses an appropriate trajectory from the table for the first target and flies the path. When a terminal condition for the target is reached (e.g. the covariance of the target state estimate has reached a certain value, or the safety zone for the target has been reached), the next target is selected from the list and the process repeats.

If a target is outside sensor range, the vehicle is commanded to turn towards the initial assumed position of the target and fly straight towards that point until the target enters the field of view. A trajectory can then be selected from the table. If the target does not enter the field of view, the observer vehicle will fly a search path of expanding circles until a target is located or the search area is cleared.

Results of a simulation of this mission are shown in Figure 4.4. In all of the figures, Blue 'wings' show vehicle position, solid blue line shows the path flown, the dotted line shows path selected from trajectory table. True target position is shown with red $*$, estimated target position (for the target currently being localized) is shown with + and error ellipse. The first subfigure shows the layout of the targets and position of the vehicle immediately after takeoff. The observer vehicle has picked an optimal trajectory out of the lookup table for the first target, shown by the magenta dotted line. The vehicle continues to fly this path for the control horizon before choosing a new trajectory. Subfigure (b) shows the S-shaped trajectory flown between the first and second targets, along with a long optimal trajectory flown to localize the third target. Because the fourth target is not in view, the control system chooses to fly a hard right turn to acquire the target with the sensor system. Once acquired, an optimal trajectory is selected from the lookup table and shown by the magenta dotted line. The third subfigure shows the vehicle after passing the fourth target. Because the fifth target is well outside the sensor range of the vehicle, the control system flies a trajectory to minimize the distance between the observer vehicle and suspected location of the target as quickly as possible. This trajectory is not obtained directly from the trajectory

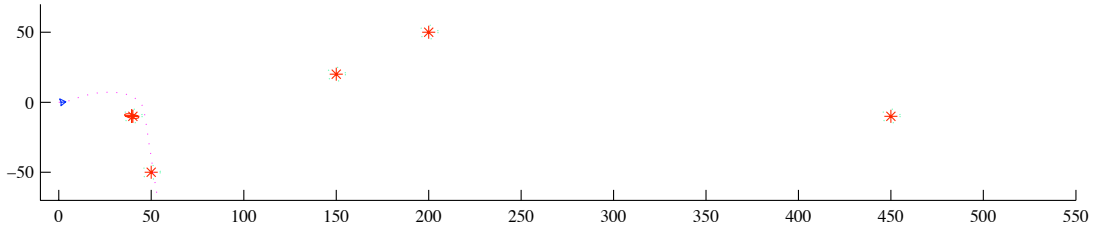
look up table, hence no dotted line is shown. Finally, in subfigure (d), the complete path flown by the observer vehicle is shown. The long straight line between the fourth and fifth targets indicates the time during which the fifth target was outside of the sensor range of the observer. When the sensor does acquire the target, a curved optimal path is flown to localize the fifth target. During the flight, portions of 18 trajectories in the lookup table were flown.

4.3 Target Localization Using Waypoint Parameterization

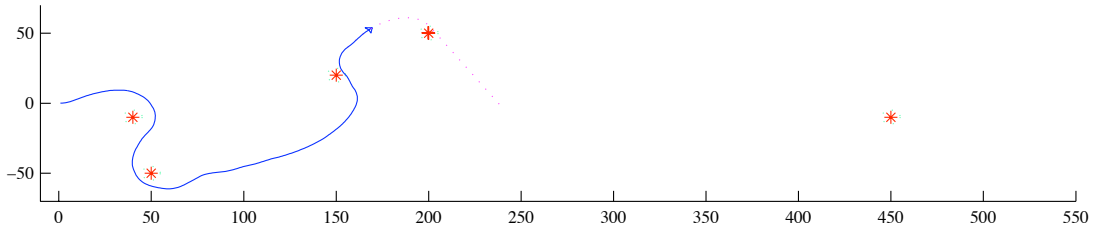
This section discusses simulation results for the waypoint parameterization. Similar to Section 4.2, target localization using the trajectory table is illustrated first using a sample single-target task. Observer vehicle and sensor variations are then discussed. A comparison of localization performance and computation power required for the parameterized method versus a direct optimization is presented. Finally, the use of the waypoint-parameterized table for localizing multiple targets is illustrated.

4.3.1 Using the Table

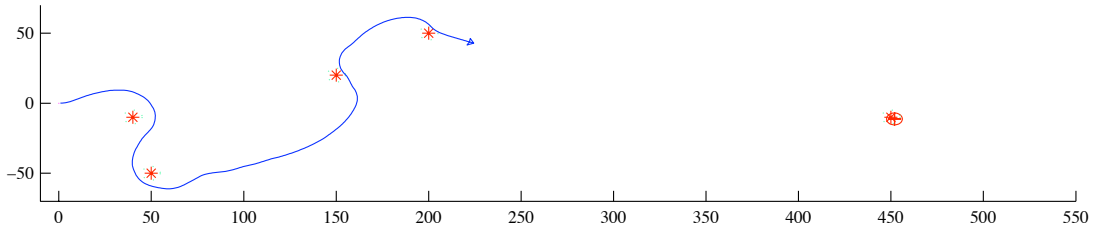
A sequence of images showing localization of a single target using the waypoint-based parameterization is given in Figure 4.5. In the figure, true target location shown with $*$, estimated target location shown with $+$ and associated error ellipse. The green line shows the current path selected from the trajectory table, blue line shows the path flown. The results show the target is initially within the field of view of the sensor, so a trajectory can be selected immediately. Exactly like the turn-rate-based parameterization, the vehicle follows the trajectory while estimating target state for a control horizon $T_c = 2$ seconds and the target localization task is concluded as the vehicle passes the target. Again, sensor noise is assumed to be Gaussian, here $\sigma_\nu = 0.0175$ rad (i.e. 1°) was used. Using this parameterization and allowing for adaptation, the target was localized to an accuracy of 0.075 meters using 0.13 seconds of CPU time. Direct optimization resulted in a localization accuracy of 0.064 meters using 2.2 seconds of CPU time. Again, CPU times for



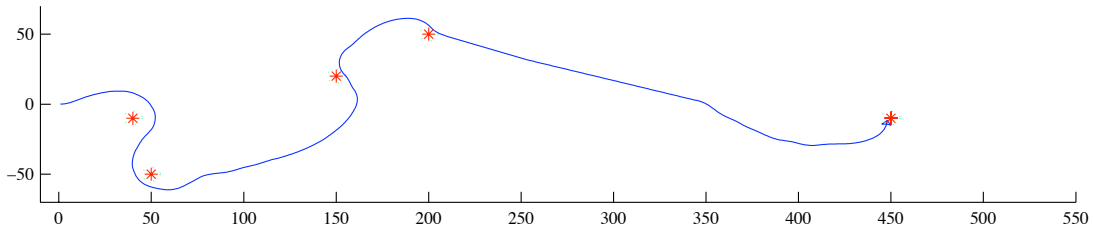
(a) t = 0.2 s



(b) t = 34.7 s

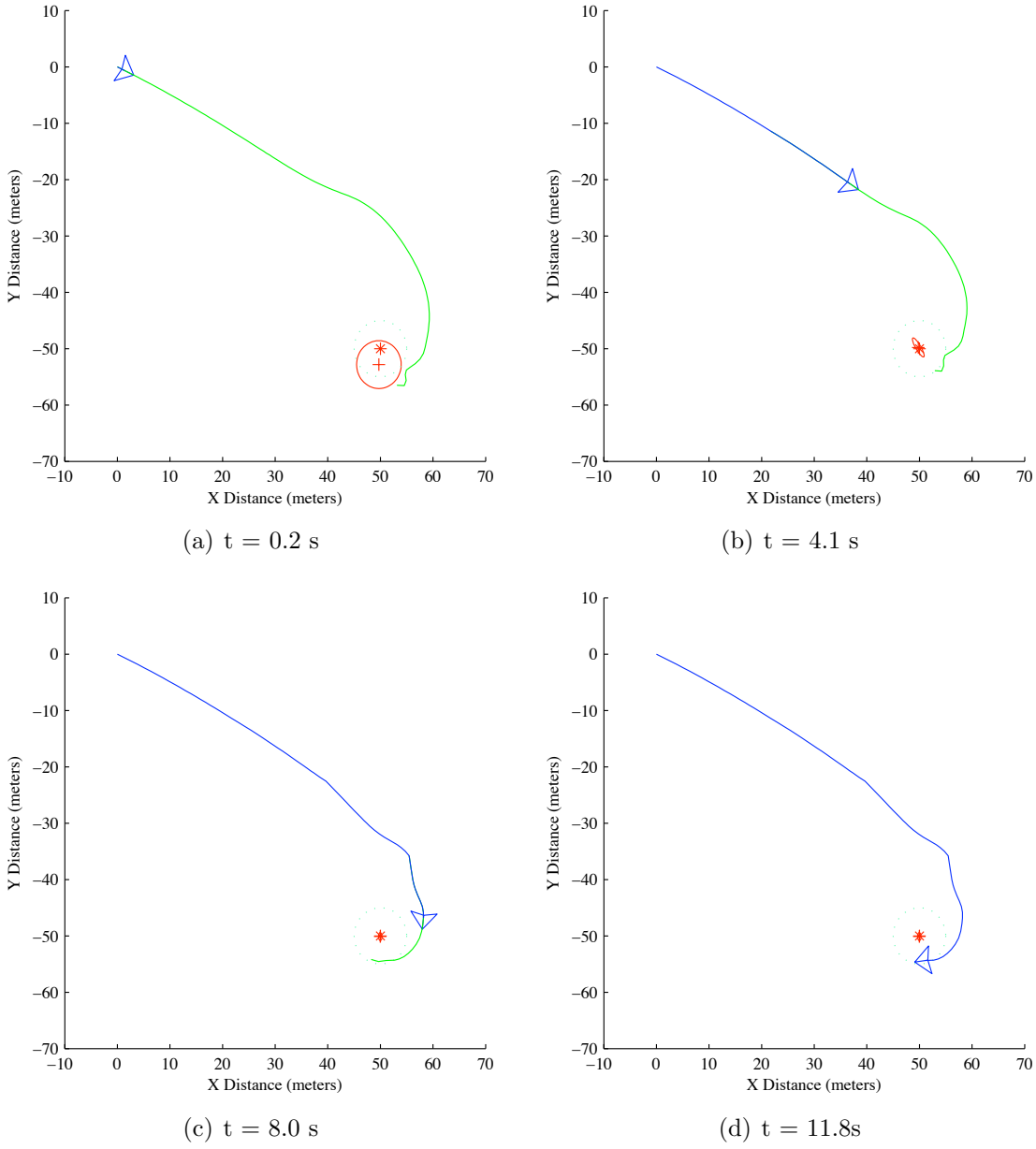


(c) t = 41.5 s



(d) t = 70.1 s

Figure 4.4. Sequential target localization using the turn-rate parameterization.



(a) t = 0.2 s

(b) t = 4.1 s

(c) t = 8.0 s

(d) t = 11.8s

Figure 4.5. Snapshots of a single target localization run using the waypoint-parameterized trajectory table.

this comparison were measured on a desktop workstation with an Intel 2.4 GHz processor.

Figure 4.6 shows that the trajectory created by dimensionalizing a set of ten waypoints stored in the lookup table and then interpolating generates essentially the same trajectory as a direct optimization in dimensional space. For practical

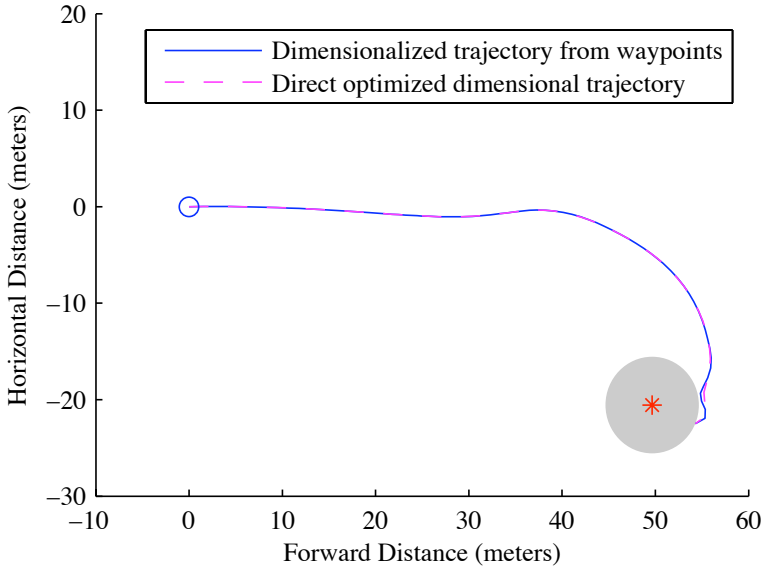
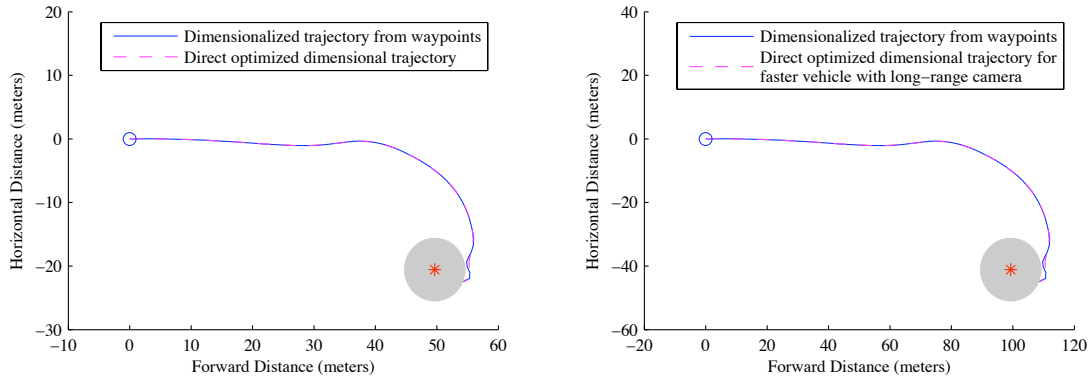


Figure 4.6. Direct optimization in dimensional space and using waypoints from a lookup table yield essentially the same path.

purposes, the trajectories are the same because the small variation in paths near the target occur when the target is outside the field of view of the observer vehicle's sensor package and, thus, is not providing any information gain.

4.3.2 Observer Vehicle and Sensor Variations

Because the trajectory is stored as a series of waypoints, a trajectory following controller allows vehicles of different speeds to use the single trajectory stored in the lookup table to best localize a target. Since the table was generated by non-dimensionalizing the target localization problem using sensor range R, sensor update period T_f and observer vehicle speed $\hat{v}$, intuition suggests that trajectories stored in the lookup table are optimal for any vehicle and sensor package that has the same "characteristic number" as the generated trajectory table. This "characteristic number" relates the sensor range and sample time, R and T_f to the observer vehicle speed, v and is given by Equation 3.28. Figure 4.7 shows that different observer vehicle and sensor combinations can still fly near-optimal paths using the same table generated for a single "characteristic number." Flying at a slower airspeed and with a short-range sensor, the vehicle in the left figure has the



(a) $v = 9m/s, R = 100m$ (b) $v = 18m/s, R = 200m$

Figure 4.7. Different observer vehicle and sensor combinations yield observers with the same "characteristic number" and can use the generated table directly.

same "characteristic number" as the vehicle in the right figure that is flying at a faster airspeed but also has a longer-range sensor. Both vehicles can directly use the lookup table generated using the waypoint parameterization for the specific "characteristic number."

Simulations have further shown that observer setups having different CN from that of the table (e.g. due to a difference in observer vehicle speed) can still localize a given target well using the trajectory from the lookup table. Because the trajectories are stored as non-dimensional waypoints, an observer vehicle setup with a sensor of any range can dimensionalize the trajectories to a valid real space path. Figure 4.8 shows a path that was optimized within dimensional space for a vehicle and sensor package with "characteristic number" CN twice as large as the CN of the trajectory table compared with the path from the trajectory table. Simulation results have shown that the difference in cost is less than 0.5% when a path (i.e. sequence of waypoints) obtained from the trajectory table is flown.

4.3.3 Simulation Results

A series of simulations was conducted to evaluate the performance of the trajectory table versus a direct optimization. For the simulations, a target was placed at a random location within the field of view of the sensor. The trajectory corresponding to the nearest nominal target location was chosen and the observer aircraft flew

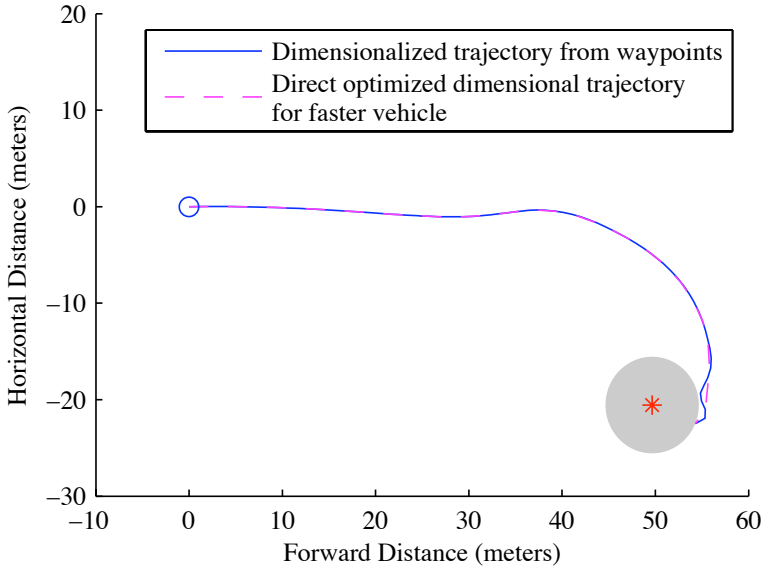


Figure 4.8. Comparison of a directly optimized path versus the trajectory table for different "characteristic numbers."

and took measurements along the entire trajectory. In this study, no adaptation of the trajectory occurred as the target state estimate was refined. This was then compared to a trajectory optimized for the actual (random) target location. The ratio of optimization cost from the lookup table trajectory to the optimization cost from a direct optimized trajectory versus target distance is shown in Figure 4.9. On average, a trajectory from the lookup table results in 81.0% the optimization cost as the direct (true) optimized trajectory for a random target placed in the sensor field of view. It is important to note that the optimization cost is the parameter minimized during the optimization. A larger magnitude negative number, as provided by direct optimization, in the optimization cost is thus preferable to a smaller magnitude negative number that is calculated from the table trajectory.

When comparing information gain alone, which is directly representative of the target localization performance of the trajectory, the table lookup method performs extremely well. For the 500 simulations, the lookup table method provides, on average, 90.0% the information gain as the true optimized path. This is because the field of view weight in the optimized cost function, while necessary for the optimizer to find a solution, does not affect the information gain in real-life. As can be seen in Figure 4.10, in some cases, the table lookup method performs

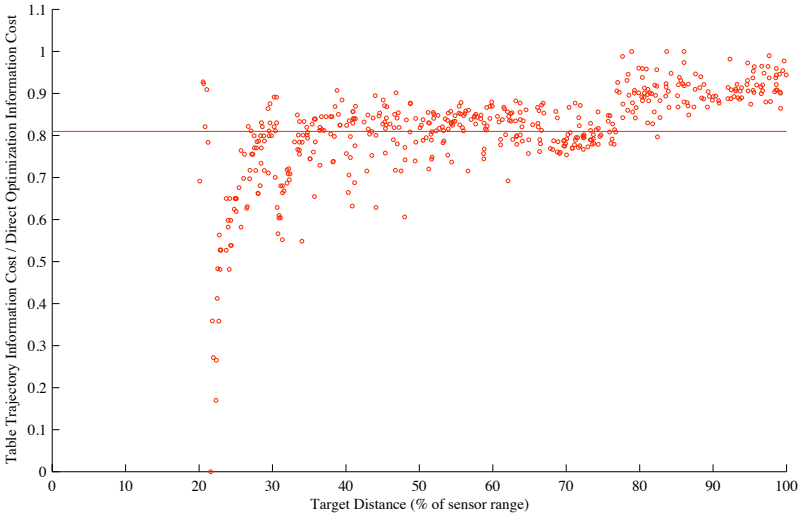


Figure 4.9. Comparison of the cost for trajectories from the lookup table versus a direct optimized trajectory for 500 random target locations.

significantly (greater than 20%) better than the directly optimized path. In these cases, small differences in the trajectory result in the observer vehicle obtaining a small increase in number of measurement locations where the target is within the field of view of the sensor. It is important to note in these results that the difference in performance is due mostly to difference in the actual target location and the nominal target location associated with the initial waypoint trajectory. Additionally, adaptation was not considered in these simulations, which likely would significantly improve the localization performance when using the waypoint-based lookup table trajectories.

4.3.4 Comparison of Required Computational Power

The table trajectory can also be retrieved from memory and dimensionalized online much faster than an optimization can take place. Times for generating a sequence of ten waypoints in dimensional space for a waypoint-following controller are given in Table 4.2 and shows that, on average, the table lookup occurs more than 130 times faster than the online optimization. It should be noted that the median optimization time of 1.28 seconds represents flight over 10-15% of the sensor range for a nominal μAV. Thus, trajectories computed online are likely to be obsolete

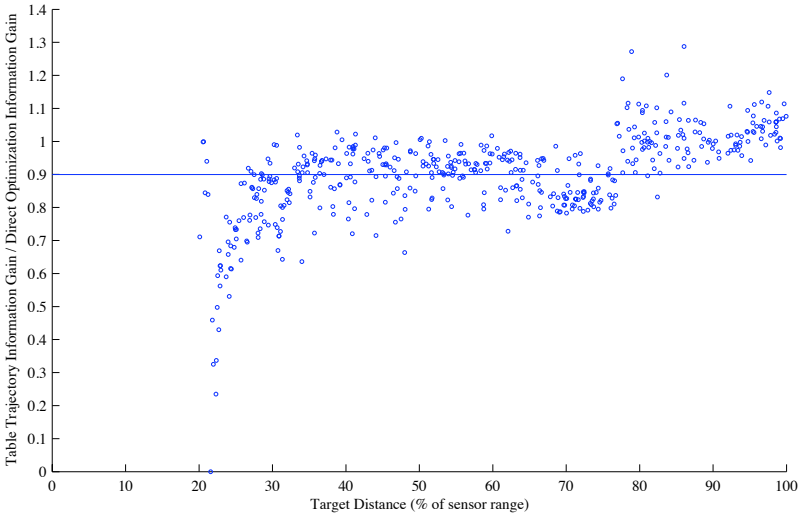


Figure 4.10. Comparison of the information gain alone for trajectories from the lookup table versus a direct optimized trajectory for 500 random target locations.

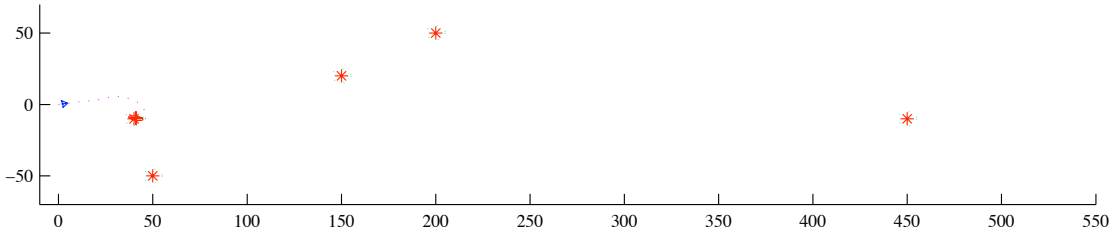
before they can be flown. Note also that the CPU times reported were measured using a workstation-class computer with an AMD Opteron processor clocked at 2.6 GHz. The computation time will be significantly longer using a processor carried onboard a μAV.

Table 4.2. Comparison of CPU times for generating a dimensional trajectory from the lookup table versus online optimization using a 2.6 GHz AMD Opteron processor.

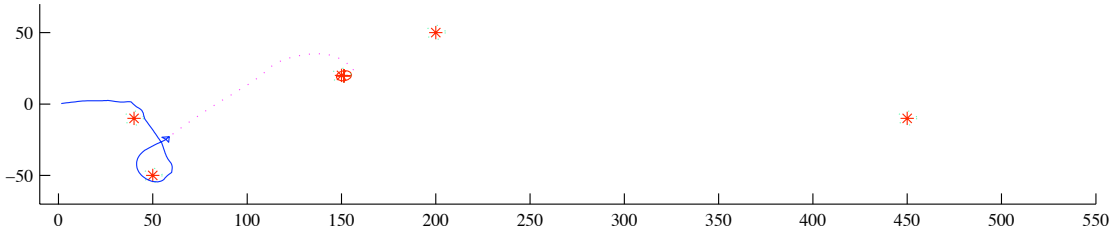
	Lookup Table	Online Optimization
Minimum	0.0084 s	0.4822 s
Maximum	0.0403 s	2.5074 s
Median	0.0084 s	1.2779 s
Mean	0.0089 s	1.2364 s

4.3.5 Sequential Target Localization

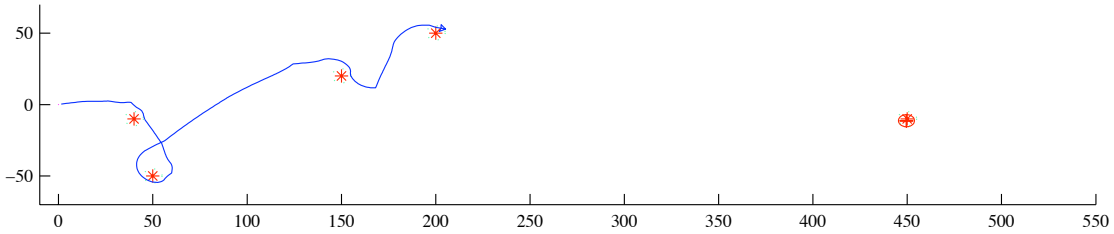
A mission for a small or micro UAV may consist of localizing and tracking a sequence of targets provided by a human operator. Here it is assumed that initial uncertain estimates of target locations are available and that the 'visit sequence' is determined by the human operator.



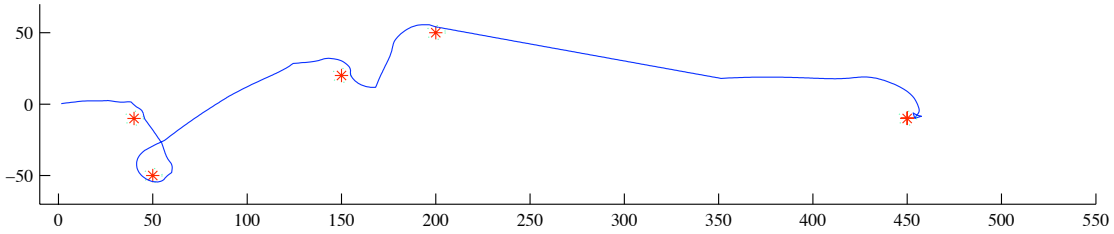
(a) t = 0.2 s



(b) t = 14.2 s



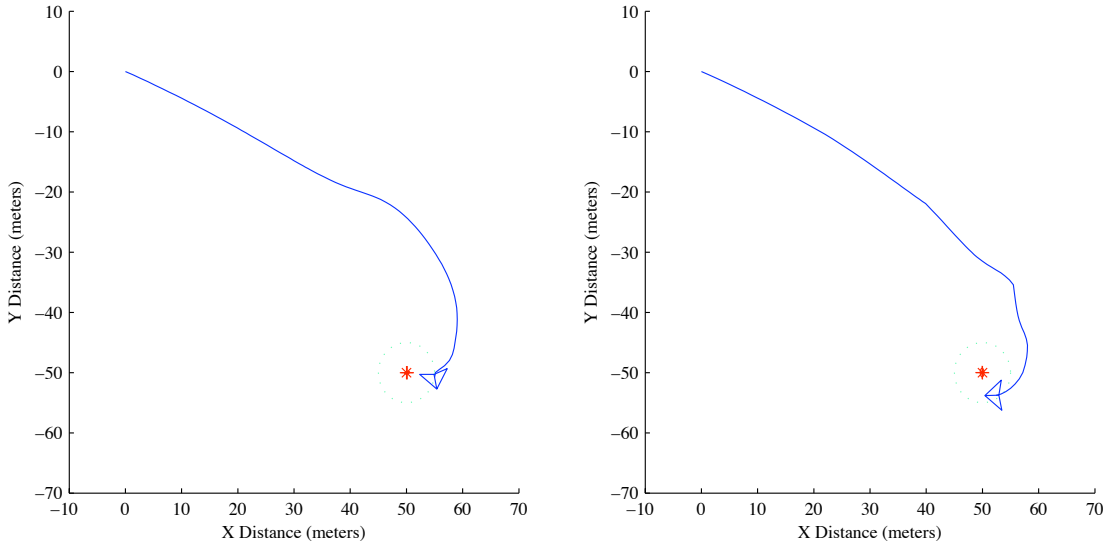
(c) t = 34.7 s



(d) t = 57.1 s

Figure 4.11. Sequential target localization using the waypoint parameterization.

Results of a simulation of this mission are shown in Figure 4.11. In the figures, blue 'wings' show vehicle position, the solid blue line shows the path flown, and the dotted line shows path selected from trajectory table. True target position is shown with red $*$, estimated target position (for the target currently being localized) is shown with + and error ellipse.



(a) No adaptation

(b) Adaptation: $T_c = 2s$

Figure 4.12. Allowing for path adaptation as the target state estimate is refined improves the localization accuracy of the lookup table provided paths.

4.4 Adaptation

As the vehicle follows a trajectory selection from the table, target state estimation occurs in real time. By selecting a new trajectory from the table when the estimated target state has changed, the target localization trajectory can be made adaptive to changes. In this research, a particular trajectory is followed for a control horizon T_c and a new trajectory is selected at the end of the control horizon. The benefits of adaptation is shown in Figure 4.12. Without adaptation, the lookup table method localizes the target within 0.16 meters. With adaptation and a control horizon of two seconds, the lookup table method localizes the target to within 0.075 meters, or better than double the localization accuracy.

4.5 Summary

Simulation results showing the usefulness and accuracy of the parameterized trajectories stored in lookup tables for target localization tasks have been presented. First, a turn-rate parameterization was used in both single- and multiple- target localization tasks. A study of table nominal target density was performed and

found the surprising result that a 5×5 table performed the localization task better than both the 10×10 and 20×20 sized tables. This smaller density 5×5 table requires 75% less storage space than the 10×10 table, also reducing the cost of the observer vehicle. Results using the waypoint parameterization are then presented. Single and multiple target localization using the stored trajectories provide excellent localization when compared to direct optimized trajectories. On average, the table trajectories will provide 90% of the information gain about a target that the online optimization would. This result does not take into account the advantage of fast trajectory adaptation that is possible when using the table trajectories. A study of required computational power showed that the mean time required to compute a trajectory dropped from 1.24 seconds using true optimization to 0.0089 seconds using the table trajectories on an AMD Opteron processor. Finally, a simple study shows that allowing adaptation when using the table trajectories doubles the accuracy of the localization task. Because of the minimal computation time required (0.0089 seconds mean) to calculate a trajectory using the table, allowing for adaptation would have no significant effect on other observer vehicle tasks.

Chapter 5

Conclusion

Target localization and tracking is a key application of micro air vehicles (μAVs) and autonomous submunitions. Because of limitations in sensing that can be carried, state estimation and the coupled problem of trajectory generation must be solved to enable useful operation in realistic scenarios.

This thesis has focused on the problem of real-time trajectory generation to maximize the information gained about a target. The motivating problem is tracking a target using a μAV, and this led to the major contribution of this research: development of a system for trajectory selection suitable for use on the limited computing hardware likely to be available on a μAV or autonomous submunition.

The system is based on a pre-computed lookup table of trajectories. A trajectory is represented as either a sequence of turn-rate input commands or a sequence of waypoints non-dimensionalized with respect to sensor range. The turn-rate commands or waypoint positions are computed off-line for a particular choice of vehicle speed, sensor update period, and sensor range. A characteristic number, CN, is defined and provides an indication of the number of measurements of target state that can be obtained.

This parameterized approach to trajectory generation for optimal target localization significantly reduces the real-time computational load on a small- or micro- UAV's processor, freeing capacity for other tasks such as state estimation, navigation, or communication. The goal of this research, to develop a method of near-optimal trajectory planning for target localization that is practical for use on small UAVs with limited computational power was met with trajectories providing

90% of the information gain as a true optimal trajectory and trajectory design occurring 137 times faster than the direct optimization.

Simulation results for two different parameterizations for the target localization trajectory design problem, turn-rate- and waypoint-based, are presented in this research.

In the turn-rate-based approach, the results show good localization performance. In a Monte Carlo simulation consisting of 500 simulations, the maximum localization error was less than 6%. More interesting, however, was that the trajectory table does not need to have a dense nominal target distribution to achieve this localization performance. Results from a table compression test showed that a trajectory table with 25 nominal target locations performed as well as tables with 100 and 400 nominal target locations. This test conclusively proves that this method to generating near-optimal trajectories for target localization not only reduces computational load on the observer vehicle, but also uses only a small amount of storage memory.

To conclude the turn-rate parameterization results, an error study on the aforementioned Monte Carlo simulation was performed. The study confirmed the estimator used within the simulations, an Unscented Kalman Filter (UKF), works correctly. Finally, a sequential target localization task simulation was performed using the trajectory table. In this task, portions of 18 trajectories were flown, showing how the table can be extended for multiple target missions and showing the ability of the table to allow the observer vehicle to perform fast adaptation.

The initial results of the waypoint-parameterization also show good localization performance. Because the trajectories are stored as a finite set of ten waypoints, the memory requirements for this parameterization are minimal, with the entire trajectory table and associated information taking up only 10kb of memory storage space. The waypoint-based trajectories were then dimensionalized and compared with trajectories that were optimized directly in real-space. The results of this comparison show the two trajectories are nearly identical in real-space and the small differences have little or no effect on the localization as the path differences occur when the target is outside of the field of view of the sensor system. The waypoint-based trajectory was then dimensionalized for an observer vehicle with a significantly different "characteristic number" and compared with a trajectory

created through direct optimization. Again, the two paths are nearly identical, with only small differences in some turns.

A Monte Carlo simulation consisting of 500 runs was performed using the waypoint-based trajectory table. Localization results were then compared with those obtained by optimizing a trajectory for the random target location. The dimensionalized waypoint-based table trajectories provide, on average, 81% of the information cost and 90% of the information gain, when compared to the direct optimized trajectory for a random target location. A few factors influence the localization performance and were detailed in the initial presentation of the results.

Finally, a comparison of computation times required for the trajectory design was done. This comparison shows that direct optimization of a trajectory takes 1.28 seconds on average, or about 10-15% of the sensor range of the observer vehicle. However, retrieving a non-dimensional trajectory from the lookup table and then dimensionalizing the waypoints occurs 137 times faster, or approximately 0.0084 seconds. This trajectory design time is constant for trajectories of any size, from 25-100% of the sensor range.

5.1 Summary of Contributions

5.1.1 Fast, Adaptive Trajectory Selection

A table of trajectories for a set of nominal target locations has been proposed and implemented. When a target is detected, or specified by a human operator, the vehicle selects the trajectory corresponding to the nearest nominal target location and follows the trajectory. An estimator fuses data from the vision system with knowledge of the observer vehicle state to compute an estimate of the target state along with an associated covariance. When the target state estimate has sufficiently improved, or when a control horizon is reached, a new trajectory is selected from the lookup table. This enables adaptive replanning of the trajectory. This approach results in greatly reduced on-board computing time.

5.1.2 Non-dimensional Trajectories

To generalize the trajectory table to various vehicle/sensor combinations the trajectory generation problem was non-dimensionalized using sensor range, vehicle speed and sensor update rate. The trajectory table thus has an associated characteristic number which is an indication of the number of measurements which can be obtained while following a path. Any vehicle/sensor combination which shares this characteristic number with the table can obtain optimal trajectories for target localization. Vehicle/sensor combinations with differing characteristic numbers can still use the same table, although the trajectories may be sub-optimal. In the cases investigated in this research it was found that the performance reduction associated with differing characteristic numbers was actually very small.

5.1.3 Performance Verification

Results of Monte Carlo simulations show that the information gained about the target using the trajectory table is almost the same as that gained by direct computation of optimal trajectories (90% of the information is gained when following trajectories obtained from the table) at vastly reduced computational overhead. On average, trajectory generation using the trajectory table occurs 137 times faster than trajectory generation using an online optimization routine.

5.2 Suggestions for Future Work

There are a number of directions to be taken for future work on this research. To demonstrate the applicability of this research in real-world problems, the generated trajectory tables should be initially tested on a ground-based robot. After successful testing, the trajectory tables can then be demonstrated on small UAVs.

Extending the optimization and table-based trajectories to localize the target in three dimensions while maintaining the versatility and applicability afforded by the non-dimensional problem formulation should also be considered. Other optimization criteria, such as minimizing the time or fuel required to achieve a specified maximum localization error could also be considered and trajectory tables could be created using those criteria.

Finally, this method of generating near-optimal trajectories could also be used to provide better estimates of "cost-to-go" in dynamic programming problems and other methods of online optimization in observer vehicles where enough computation power is available.

Bibliography

[1] Kailath, T., A. H. Sayed, and B. Hassibi (2000) *Linear Estimation*, 1st ed., Prentice-Hall.

[2] Bar-Shalom, Y., X. R. Li, and T. Kirubarajan (2001) *Estimation with Applications to Tracking and Navigiation*, Wiley Interscience.

[3] Simon, D. (2008) *Optimal State Estimation*, Wiley Interscience, Hoboken, New Jersey.

[4] Proctor, A. and E. N. Johnson (2005) "Vision-only Approach and Landing," in *AIAA Guidance, Navigation and Control Conference*, AIAA Paper 2005-5871, American Institute of Aeronautics and Astronautics, San Francisco, California.

[5] Roumeliotis, S. I., A. E. Johnson, and J. F. Montgomery (2002) "Augmenting Inertial Navigation with Image-Based Motion Estimation," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, Washington, DC.

[6] Wu, A. D., E. N. Johnson, and A. A. Proctor (2005) "Vision-Aided Inertial Navigation for Flight Control," in *AIAA Guidance, Navigation and Control Conference*, AIAA Paper 2005-5998, American Institute of Aeronautics and Astronautics, San Francisco, California.

[7] Prazenica, R. J., A. Watkins, A. J. Kurdila, Q. F. Ke, and T. Kanade (2005) "Vision-Based Kalman Filtering for Aircraft State Estimation and Structure from Motion," in *AIAA Guidance, Navigation and Control Conference*, AIAA Paper 2005-6003, American Institute of Aeronautics and Astronautics, San Francisco, California.

[8] Webb, T. P., R. J. Prazenica, A. J. Kurdila, and R. Lind (2005) "Vision-Based State Estimation for Uninhabited Aerial Vehicles," in *AIAA*

Guidance, Navigation and Control Conference, AIAA Paper 2005-5869, American Institute of Aeronautics and Astronautics, San Francisco, California.

[9] Marks, R. L. (1995) *Experiments in Visual Sensing for Automatic Control of an Underwater Robot*, Ph.d. thesis, Stanford University, Department of Aeronautics and Astronautics, Stanford CA 94305, also published as SUDAAR 681.

[10] Leabourne, K. N., S. M. Rock, S. D. Fleischer, and R. L. Burton (1997) "Station Keeping of an ROV Using Vision Technology," in *OCEANS 97 Conference*, vol. 1, MTS/IEEE, Halifax, Nova Scotia, pp. 634–640.

[11] Richmond, K. and S. Rock (2005) "A Real-time Visual Mosaicking and Navigation system," in *Proceedings of the Unmanned Untethered Submersible Technology Conference*, Autonomous Undersea Systems Institute, Durham, New Hampshire.

[12] Cheng, Y., M. Maimone, and L. Matthies (2005) "Visual Odometry on the Mars Exploration Rovers," in *IEEE Conference on Systems, Man and Cybernetics*, The Big Island, Hawaii.

[13] Meingast, M., C. Geyer, and S. Sastry (2005) "Vision Based Terrain Recovery for Landing Unmanned Aerial Vehicles," in *Conference on Decision and Control*, IEEE, Bahamas.

[14] Prazenica, R., A. Kurdila, R. Sharpley, and J. Evers (2006) "Vision Based Geometry Estimation and Receding Horizon Path Planning for UAVs Operating in Urban Environments," in *American Controls Conference*, Minneapolis, Minnesota.

[15] Huster, A. and S. Rock (2003) "Relative Position Sensing by Fusing Monocular Vision and Inertial Rate Sensors," in *Proceedings of the 11th International Conference on Advanced Robotics (ICAR)*, Coimbra, Portugal.

[16] Pollefeys, M., L. V. Gool, M. Vergauwen, K. Cornelis, F. Verbiest, and J. Tops (2003) "3D Capture of Archaeology and Architecture with a Hand-Held Camera," in *ISPRS workshop on Vision Techniques for Digital Architectural and Archaeological Archives*, Ancona, Italy, pp. 262–267.

[17] Jung, I.-K. and S. Lacroix (2003) "High Resolution Terrain Mapping using Low Altitude Aerial Stereo Imagery," in *Ninth IEEE International Conference on Computer Vision*, vol. 2, pp. 946–951.

[18] Han, M. and T. Kanade (July 7-14, 2001) "Multiple Motion Scene Reconstruction from Uncalibrated Views," in *Proceedings of the Eighth IEEE International Conference on Computer Vision*, Vancouver, Canada.

[19] Wang, C., H. Ma, and D. J. Cannon (1997) "Human-Machine Collaboration in Robotics Integrating Virtual Tools with a Collision Avoidance Concept Using Conglomerates of Spheres," *Journal of Intelligent and Robotic Systems*, **18**(4), pp. 367–369.

[20] Thrun, S., D. Fox, and W. Burgard (2005) *Probabilistic Robotics*, MIT Press.

[21] Foxlin, E. and L. Naimark (2003) "VIS-Tracker: A Wearable Vision-Inertial Self-Tracker," in *IEEE VR2003*, IEEE, IEEE, Los Angeles, California USA.

[22] Davison, A. J. (2003) "Real-Time Simultaneous Localisation and Mapping with a Single Camera," in *International Conference on Computer Vision*, Nice, France.

[23] Neira, J., M. I. Ribeiro, and J. D. Tardós (1997) "Mobile Robot Localization and Map Building using Monocular Vision," in *Fifth International Symposium on Intelligent Robotic Systems*, Stockholm, Sweden.

[24] Langelaan, J. W. (2007) "State Estimation for Autonomous Flight in Cluttered Environments," *Journal of Guidance, Control and Dynamics*, **30**(5), pp. 1414–1426.

[25] Kim, J. and S. Sukkarieh (2007) "Real-time implementation of airborne inertial-SLAM," *Robotics and Autonomous Systems*, **55**, pp. 62–71.

[26] Latombe, J.-C. (1991) *Robot Motion Planning*, Kluwer Academic Publishers, Norwell, Massachusetts.

[27] LaValle, S. M. (2006) *Planning Algorithms*, Cambridge University Press. URL `http://msl.cs.uiuc.edu/planning/`

[28] Huang, S., N. M. Kwok, G. Dissanayake, Q. P. Ha, and G. Fang (2005) "Multi-Step Look-Ahead Trajectory Planning in SLAM: Possibility and Necessity," in *IEEE International Conference on Robotics and Automation*, Institute of Electrical and Electronics Engineers, IEEE, Piscataway, New Jersey.

[29] Frew, E. W. (2003) *Observer Trajectory Generation for Target-Motion Estimation Using Monocular Vision*, Ph.D. thesis, Stanford University, Stanford, California.

[30] Grocholsky, B., A. Makarenko, and H. Durrant-Whyte (2003) "Information Theoretic Coordinated Control of Multiple Sensor Platforms," in *IEEE International Conference on Robotics and Automation*, vol. 1, IEEE, Piscataway, New Jersey, pp. 1521–1526.

[31] Ousingsawat, J. and M. E. Campbell (2007) "Optimal Cooperative Reconnaissance Using Multiple Vehicles," *Journal of Guidance, Control and Dynamics*, **30**(1), pp. 122–132.

[32] Sinclair, A. J., R. J. Prazenica, and D. E. Jeffcoat (2007) "Simultaneous Localization and Planning for Cooperative Air Munitions," in *International Conference on Cooperative Control*.

[33] Nygards, J., P. Skoglar, J. Karlholm, R. Bjorstrom, and M. Ulvklo (2005) *Towards Concurrent Sensor and Path Planning*, *Tech. rep.*, FOI.

[34] Geiger, B. R., J. F. Horn, A. Delullo, L. N. Long, and A. F. Neissner (2006) "Optimal Path Planning of UAVs Using Direct Collocation with Nonlinear Programming," in *AIAA Guidance, Navigation and Control Conference*, Keystone, Colorado.

[35] van der Merwe, R. and E. Wan (2001) "The Square-Root Unscented Kalman Filter for State and Parameter-Estimation," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, Salt Lake City, UT.

[36] van der Merwe, R., E. Wan, and S. Julier (2004) "Sigma Point Kalman Filters for Nonlinear Estimation and Sensor Fusion: Applications to Integrated Navigation," in *AIAA Guidance, Navigation and Controls Conference*, AIAA, Providence, RI USA.