PARTICLE FILTER-BASED ARCHITECTURE FOR VIDEO TARGET TRACKING AND GEO-LOCATION USING MULTIPLE UAVS

A Dissertation
Presented to
The Academic Faculty

by

Christopher James Sconyers

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in
Electrical and Computer Engineering

Georgia Institute of Technology

May 2013

PARTICLE FILTER-BASED ARCHITECTURE FOR VIDEO TARGET TRACKING AND GEO-LOCATION USING MULTIPLE UAVS

Approved by:

Dr. George Vachtsevanos
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Justin Romberg
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Thomas Michaels
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Eric Johnson
School of Aerospace Engineering
Georgia Institute of Technology

Dr. Anthony Yezzi
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: December 19, 2012

For my wife, Bernadette. I would be lost without you.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor, Dr. George Vachtsevanos. I am incredibly grateful for the opportunity to work with him and to share his knowledge and experience. I would like to thank him for introducing me to and allowing me to participate in many new avenues of research, of which I have benefitted greatly. And, I owe him immensely for the many years of pushing and prodding me to finish my work.

I would like to thank my lovely wife, Bernadette, whose support and belief in me kept me going. She sacrificed a lot to help me complete my research, and I cannot begin to express my gratitude for that. This would not be possible without her.

I would like to thank my parents, Woody and Elaine, who inspired me to achieve this goal. Their encouragement began long before I entered higher education, and continues to this day. They helped me to shape my ambitions, and showed me how to reach them.

I would like to thank my committee members, Dr. Thomas Michaels, Dr. Anthony Yezzi, Dr. Justin Romberg, and Dr. Eric Johnson. Besides their confidence in me, they have been invaluable in their comments, suggestions, and critiques.

I would like to give special thanks to my friend Dr. Ioannis A. Raptis, who showed me how to conquer the mountain that is rotorcraft control. Without his help, my UAVs would crash without mercy.

Last, I would like to thank the friends and colleagues I have worked with over the years. Each of them gave to me in their own way, and I hope I have given back to them in kind.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

APF	Auxiliary Particle Filter
API	Application Programming Interface
CCD	Charge-coupled Device
CDF	Cumulative Density Function
CDKF	Central Difference Kalman Filter
CoG	Center of Gravity
CPU	Central Processing Unit
CSW	Cumulative Sum of Weights
EKF	Extended Kalman Filter
EM	Electromagnetic [Emission]
EO/IR	Eelctro Optic / Infrared
FFT	Fast Fourier Transform
GDOP	Geometric Dilution of Precision
GPS	Global Positioning System
GPU	Graphics Processing Unit
IMM	Interacting Multiple Model Filter
IMM-PDA	IMM-Probabalistic Data Association Filter
IMU	Inertial Measurement Unit
LIDAR	Light Detection and Ranging
LOS-Rate	Line-of-Sight Rate
MAV	Micro Aerial Vehicle
OID-PF	Optimal Importance Density Particle Filter
OID-PF(RR)	OID-PF (Reversed Resampling)
PDF	Probability Density Function
PF	Particle Filter
RADAR	Radio Detection and Ranging
RF	Radio Frequency
SPKF	Sigma-point Kalman Filter
TD-APF	Transition Density-based APF

TDOA	Time Difference of Arrival
TOA	Time of Arrival
TOF	Time of Flight
TPP	Tip-path-plane
TWS	Track-while-scan
UAV	Unmanned Aerial Vehicle
UKF	Unscented Kalman Filter

SUMMARY

Research in the areas of target detection, tracking, and geo-location is most important for enabling an unmanned aerial vehicle (UAV) platform to autonomously execute a mission or task without the need for a pilot or operator. Unmanned systems that are typically designed to be small and lightweight may not be able to support devices that can provide accurate guidance or localization. Small class UAVs and video camera sensors complemented with "soft sensors" realized only in software as a combination of *a priori* knowledge and sensor measurements are called upon to replace the cumbersome precision sensors on-board a large class UAV. The objective of this research is to develop a geo-location solution for use on-board multiple UAVs with mounted video camera sensors only to accurately geo-locate and track a target.

The Kalman filter is still widely used in state estimation, and has been applied many times to the problem of target tracking and geo-location. However, the limitations of the Kalman filter, linear or linearized systems and Gaussian noise, require precise simplifications of the problem space or careful construction of supporting systems, like the interacting multiple model filter. Recently, the particle filter is being adapted to the problem of target geo-location. Current particle filter applications may be split into those that use time of arrival or time difference of arrival measurements from large-scale sensor technologies to track a target, or a variety of sensors including video sensors to geo-locate a stationary ground target, usually from fixed-wing aircraft. Advances in video image frame tracking are combined with simple or advanced topography approximations to project estimated vectors-to-target onto the ground, although uncertainty in measurements and topography estimations translate to large geo-location uncertainty.

There is a need to develop and implement on-board small UAVs suitable sensing technologies that are lightweight, flexible, and cost effective, like EO/IR, passive sonar(particularly for underwater vehicles, or passive radio frequency spectrum emission detection, which in combination with appropriate estimation algorithms can address efficiently and effectively the target detection and geo-location problem. This research introduces an estimation solution that combines the power of the particle filter with the utility of the video sensor as a general solution for passive target geo-location on-board multiple UAVs.

The particle filter is taken advantage of, with its ability to use all of the available information about the system model, system uncertainty, and the sensor uncertainty to approximate the

statistical likelihood of the target state. When applied to target geo-location the particle filter accepts video sensor data in the form of the detected location of the target in an image frame and estimates the location of the target within its environment based on knowledge about the sensor and assumptions on the target model.

The observation model for the video sensor is the basis for determining the probable location of the target, and this is constructed in two parts: the projection model and the transformation model. The combination of these two models transforms points related to the target location in the world coordinate system to points as interpreted by the combined camera-UAV system. The particle filter uses this transformation as a means of evaluating the likelihood of target location estimates.

The state space model for the target system is constructed as an approximation of the behavior of the target. Only ground targets are considered and the state space model is designed with ground vehicles as potential targets. The non-holonomicity of the car is deconstructed and approximated as a set of linear and non-linear operating modes, with a Markov switching model for changing between operating modes.

The geo-location particle filter is tested online and in real-time in a simulation environment involving multiple UAVs with video cameras and a maneuvering ground vehicle as a target. During simulation, the geo-location particle filter estimates the target location in meters with a relative error of as small as 4% (one percent being one meter of error for every 100 meters distance to the UAVs). The addition of UAVs or particles to the system improves the location estimation accuracy with minimal addition of processing time, and the UAV control and trajectory generation restrict the UAV to a desired range to minimize error.

The main contributions of this research are as follows:

- A robust particle filtering algorithm capable of geo-locating a target from a UAV platform using limited video sensor information.
- A multi-modal state estimation solution for target behavior and location estimation.
- A real-time geo-location solution with flexibility in design complexity and robustness to maximize probability of mission completion.
- A scalable framework for the addition and removal of multiple UAVs, with minimal computation cost requirements.
- A closed-loop target tracking system, using the geo-location particle filter and a UAV control and trajectory generator to minimize the geo-location estimation error.

CHAPTER I
PROBLEM DESCRIPTION & BACKGROUND

1.1 INTRODUCTION

Unmanned aerial vehicles (UAVs) are fast becoming an integral part of both military and, to a lesser degree, commercial airspace. The need for removing the human element from the situation can be viewed from a number of different angles, or as is more commonly known as the three D's: missions that are dull, dirty, or dangerous [7]. A pilot may make errors in judgement, may become tired during long operations, or may be put in harms way. Replacing the pilot with an operator flying a UAV remotely reduces the danger to personnel to a minimum, while creating an autonomous unit may mitigate human error.

Research in the areas of target detection, tracking, and localization are most important for creating a UAV platform that can autonomously execute a mission or task without the need for a pilot or operator. While instrumentation abounds in conventional aircraft, unmanned systems are typically designed to be small, cheap, and lightweight, and devices that can provide accurate guidance or localization may not be available. "Soft sensors," or sensors that are realized only in software as a combination of *a priori* knowledge and indirect hard sensor data, are required to replace the cumbersome precision sensors on-board aircraft. Global Positioning Systems (GPS) and Inertial Measurement Units (IMU) can provide information to the UAV about its own location. However, information about objects outside the UAV, objects such as geographic landmarks, potential obstacles to flight, and mission-critical targets, is not easily discerned from the limited sensor hardware available without the aid of a human operator.

This proposal examines the problem of target tracking and localization with the goal of developing a software solution for automatic localization using only video cameras on-board a single or multiple UAVs. It begins with an investigation of the history of the problem of tracking and localization (Section 1.2) and state-of-the-art solutions applicable to general cases and cases specific to UAVs (Section 1.3). It proposes a software architecture, based on the particle filter (Chapter 2), that can detect a ground target in a camera image frame, then quickly, accurately, and robustly locate and track the target from a single or multiple UAVs (Chapter 3). It provides a derivation of UAV motion dynamics and a control and trajectory generation method (Chapter 4). Lastly, it shows the accuracy and efficiency of this geo-location particle filter approach through simulated trials (Chapter 6).

1.2 HISTORICAL METHODS

1.2.1 LOCALIZATION GEOMETRY

The steps involved in locating ground targets, detection, tracking, transformation, and fusion, each present inaccuracies and opportunities for false information. This amounts to an unreliable estimate of an object's location, or even completely false information, such as reporting the detection of targets that do not exist. The algorithmic and heuristic techniques presented in this chapter have been developed to minimize or eliminate these limitations.

Early localization techniques regard the simple geometry of the problem. The target, representable by a point in space, can be given a coordinate along a single plane from the perspective of another point, the viewer. The orientation of this coordinate plane is dependent on the spatial relation between the viewer and the target, and ignores the third dimension: the distance from the viewer. The heart of the localization problem is using two-dimensional perspectives to estimate a three-dimensional coordinate, along with developing a proper transformation between this relative coordinate system and an absolute coordinate system. Geo-location, as presented in this thesis, deals primarily with ground targets and their GPS coordinates.

While the concept of using geometry to estimate points in space is over 2000 years old, starting with triangulation, it has been used extensively all the way through the present for navigation, surveying, localization, and more.

Triangulation is the simple method of using two known points to locate a third point using the geometric properties of a triangle [49]. Using approximate measurements of angles of incidence between the lines of sight from two observers to a fixed point, and the baseline between the observers, the target point, forming the apex of the formed triangle, can be localized as being at the intersection of the two lines of sight and its coordinates found using the law of sines.

Similar to triangulation, trilateration uses known distances from fixed locations to a point to determine its location [28]. Using the measured distances between references and target, a circle can be defined around each of the reference points, and the target will lie at the intersection of the circles. This only works for targets located on the same plane as all three reference points.

Trilateration can be used for three dimensional coordinates, but because of the nonlinear equations describing the spheres about the reference points, certain constraints must be made on the location

of these reference points to simplify the problem. Using three points, one must be defined at the origin, another along one of the Cartesian axes, and all three must lie on the same axis plane, typically defined as the x-y plane. It should be noted that any three points, with the target point, can be transformed into an arbitrary Cartesian coordinate system to meet these positioning requirements. The intersection between the three resulting spheres will result in up to two possible locations for the target point, and in some cases no solution is possible. Often this technique requires a fourth reference, or otherwise logical deduction to determine the correct estimation, such as where the target point is expected to be in either the transformed or world coordinate system.

Multilateration works like trilateration, but instead of known distances between references and a target, an emitted signal from the target object is received by fixed observers, and the time difference of arrival (TDOA) of the signal is used to calculate the target's position [46]. The TDOA of a signal between two receivers forms a hyperboloid surface on which the target emitter lies. Adding a third or fourth receiver reduces the location to existing on the curve between two intersecting hyperboloids or at a single coordinate, respectively.

Because it is easier to obtain a more accurate TDOA measurement than range or angle measurements, the multilateration technique is usually more accurate than triangulation or trilateration. The algorithm is also easily scalable, such that adding reference points refines the accuracy of the calculation. Thus, multilateration is commonly used as a localization solution.

All of these geometric techniques, and any method that require sensor data, introduce measurement error into the calculation [49]. Given aspects of the problem, such as distances between references and target, the accuracy and nature of the sensors, and the number of steps involved, it's likely these errors will grow into unwieldy inaccuracies in the location estimation. Multiple sensor readings from alternate reference locations can help refine the estimation, but not all geometric estimation methods accept multiple data points, and averaging techniques can exaggerate error or bias.

Methods exploiting statistical distributions have thus grown in popularity as a means of combining sets of sensor data to estimate or even predict target locations to a higher degree of accuracy. Data regression, Kalman filtering, and assorted Monte Carlo techniques, including particle filtering, are some of the more prevalent methods.

Data regression analysis, developed around the turn of the 19th century, statistically combines inaccurate data to produce an estimation close to the expected result. Beginning with an assumed linear data model, the unknown parameters in the model are approximated for each data point, each producing an error with the expected model. These errors are minimized using a least squares approximation, resulting in a parametric, closed-form solution that most closely represents the sample data. For nonlinear data models, this becomes an iterative process with no closed solution and the possibility of non-convergence. Unfortunately, the expected target behavior and measurement model of the geo-location problem are nonlinear.

A reasonable assumption in current technology, aiming for real-time consistent data acquisition and processing, is that a stream of sensor data on the target will be fed to the localization system. With discretized time-based data, the Kalman filter has become more common in the geo-location problem [5]. The Kalman filter uses previous state estimations to predict the current state, then combine the predicted state with current measurements to arrive at a potentially more accurate estimation.

Kalman filter state estimation is only valid under certain model constraints. The system must be a linear Markov chain with Gaussian perturbation. For nonlinear systems, there are a number of alternate Kalman filters, including the popular extended Kalman filter, unscented Kalman filter, and variations. The measurement noise must be Gaussian, but in some systems, non-Gaussian noise can be approximated in Gaussian form. The Kalman filter will then provide a first-order approximation.

Monte Carlo methods can be used to circumvent the problems associated with nonlinear regression analysis and Kalman filter shortcomings, while smoothing the uncertainties in location data. The general method is to use a mix of random sampling to locate areas of high probability, and algorithmic computation to refine estimations. These include algorithms such as random walk, but more importantly for the application of localization is the Particle Filter.

The Particle filter, a Bayesian estimator like the Kalman filter, relies on a known system model with a known or approximated probability distribution function (pdf) [14]. A number of randomized samples are distributed using the expected pdf of the system model. The particles at time k are propagated according to the system model similar to the Kalman filter, weighted according to the expected pdf and current measurements, and resampled by weight for the next iteration. The distribution of weighted particles at time k+1 can provide a close approximation to actual

data over noisy, inaccurate observations. Where the Kalman filter can only provide a first-order approximation in the face of non-Gaussian noise, the Particle filter can provide a much more accurate estimation given enough sample particles [14, 43].

1.2.2 DETECTION

While simple geometrical concepts can be used to identify an object's location using reference points, there still remains the difficult task of detecting, identifying, and tracking the respective reference points in the problem. Using the latest sensing technology, UAVs are able to distinguish their target as well as determine their positions relative to the target, relative to a local reference, or globally. The most notable techniques are RF signal tracking, active sensing, and passive sensing using cameras, and each type comes with associative costs and rewards.

A simple directional antenna is enough to determine a direction to a target that is emitting an EM transmission, though more sophisticated sensors improve the results. Triangulation between multiple UAVs, or multiple snapshots in time from a single UAV, will provide an accurate location on the transmitting target. Of course, targets may not always be emitting constantly, making tracking difficult, or may not emit any signals at all–such as a person carrying no electronics devices–making tracking impossible.

Common in many military and aeronautical applications, active sensing techniques like radar, LIDAR, or sonar use emissions like electromagnetic or acoustic waves are used to scan for and image targets. The emissions are reflected back to a receiver that can interpret characteristics like the signal travel time to determine target range, which coupled with the direction to the target can provide a rough estimate of its position. In these systems, with measurement noise that is often Gaussian and easily modeled, the system can be improved with a simple Kalman filter or the inclusion of multiple receivers, though even such improvements are not always necessary.

These active sensing techniques are unfortunately not always viable using current UAV technology. Small reconnaissance and surveillance UAVs are weight restricted such that it is not possible to include heavy sensing equipment on-board, such as RADAR. The General Atomics Predator, an actively deployed reconnaissance UAV is designed with a synthetic aperture radar on-board, though it may be removed to reduce weight [1]. Another disadvantage to note is susceptibility to jamming, reducing the accuracy of the detection and tracking systems, sometimes making them completely ineffective.

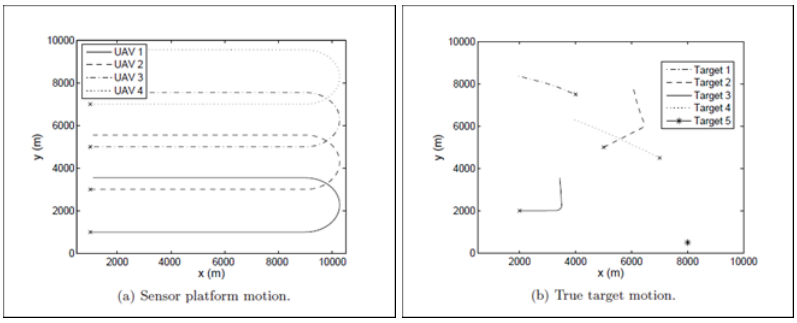
Passive sensing like video or infrared imaging are more suitable and adaptable for use on UAVs. Cameras are capable of imaging a large area at one time and capture image frames in quick succession using small, lightweight, and cheap equipment. Unfortunately, this sensing technology relies entirely on emissions from outside the UAV, most often in or near the range of visible light. Like the human eye, a camera is capable of capturing a large field of view, but unlike the human brain, image processing software is in early stages of being able to interpret an entire captured scene accurately and in real time. However, for tracking single non-cooperative ground targets from the air, video cameras are highly capable of target detection and tracking.

Many image processing algorithms exist to detect an object in an image or track an object in an image stream. Most algorithms seek to simplify the image data, such as segmentation, or eliminate areas of the image with low likelihood of containing a target object, such as feature detection or the application of neural networks. The image processing system is supplied with *a priori* data about the desired target, which is used to locate areas with a high probability of containing the target. Simple geometric transformations can translate a likely location within an image to a relative direction from the camera's reference frame. More sophisticated algorithms can approximate the distance to the target based on *a priori* size and shape data as compared to the size and shape in the image, though this data is not always available.

1.3 RELATED RESEARCH

York and Pack compare two methods for target tracking, Kalman filtering and triangulation, for localizing moving targets using multiple UAVs [49]. The moving targets are assumed to be emitting RF signals in all directions. In simulation, the UAVs follow a surveillance control architecture to search for, detect, and locate the mobile targets. Using simple Kalman filtering and triangulation, the Kalman filter was shown to take longer to localize multiple targets due to the number of iterations required to reach a steady state, while triangulation can produce results in a single iteration. The Kalman filter tracking algorithm performed more poorly due to the inconsistency of RF transmissions from the target, though it produced more accurate UAV orbit trajectories.

Although not applied to the problem of target tracking, Mao et al. used the extended Kalman filter (EKF) to localize a UAV with a temporary failure in its positioning system utilizing range measurements to two other UAVs with known location [29]. For a closed system, this is similar



(a) Sensor platform motion. (b) True target motion.

Figure 1.1: *Simulation scenario involving multiple UAVs and multiple maneuvering targets [45].*

to the problem of localizing a target using radar or TDOA measurements with two UAVs. A trilateration technique, along with the UAV motion dynamics, was used for estimating the UAVs position and linearized to be implemented in the EKF. Three UAVs were flown in simulation, and the UAV with a malfunctioning positioning system was localized to within a 40m margin of error (with UAVs spaced on the order of 5km apart). However, the EKF depends on knowledge of the UAV dynamics, and a carefully constructed covariance for its estimation errors. This knowledge may not be available in a realistic target tracking application.

Savage et al. apply the unscented Kalman filter (UKF) to locate a stationary target using TDOA measurements. Simulations show a smooth convergence of the UKF to an accuracy below the simulated TOA noise level. However, convergence is slow, and is impacted by the accuracy of the initial estimated state [46].

Plett et al. continued with the Kalman filter approach with a technique for combining sensor data (sequential DOA readings in time and space) using a sigma-point Kalman filter (SPKF), specifically the central difference Kalman filter (CDKF), to locate ground targets [39]. The SPKF is used over an EKF to achieve higher estimation accuracy without the need for computing derivatives. The results show that the technique provides an accurate location and dynamics estimation on a target being orbited by a single UAV, which is easily scalable to multiple UAVs. However, the choice of Kalman filter is based on the assumption that all pdfs are Gaussian unimodal, which is not guaranteed for passive video sensing and tracking.

The unscented Kalman filter has been used together with the interacting multiple model (IMM) filter to introduce target motion into the tracking problem [5, 44, 45]. Multiple mode state dynamics are not handled well by the Kalman filter. The IMM filter uses parallel mode estimations and creates a Gaussian mixture to approximate a single posterior density. In [44] and [45], this

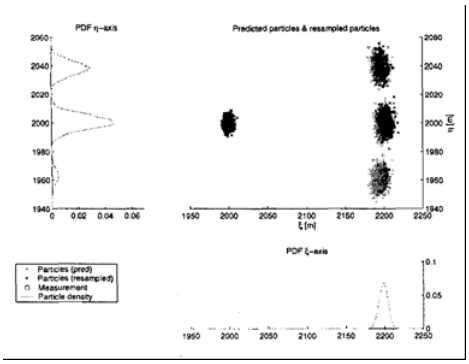


Figure 1.2: *Splitting particle based on multiple maneuvers [21].*

is applied to a simulation of single and multiple target with position and velocity estimations generated from TDOA measurements in a noisy or cluttered environment, with an S-D Lagrangian relaxation assignment to resolve the data association. This has the benefit of isolating multiple targets for separate target tracking without being prohibitively computationally expensive. In simulation, this is found to achieve a target localization accuracy on the order of tens of meters with the highest assumed measurement noise level (1ns for TDOA measurements), given multiple emitting targets and sensing platforms (Figure 1.1).

The particle filter may be more suited to the task of tracking a target with unknown dynamics, given the non-linear and non-Gaussian nature of the problem, with the advantage that the estimation error is independent of the state dimension. Doucet and Gordon developed a particle filter algorithm for tracking a maneuvering target, assuming a linear observation relationship [11]. The particle filter track estimate proved to be able to track a simulated maneuvering target in the presence of clutter. For the same simulated target, the previously developed interacting multiple model probabilistic data association (IMM-PDA) technique failed to accurately track the target [4].

In 2000, Karlsson and Bergman used the auxiliary particle filter (APF) for tracking a moving target without linearizing the target state space [21]. Target maneuvers were modeled using a Markov chain and splitting and weighting each particle according to the model (Figure 1.2). For simulations of an Air Traffic Control track-while-scan (TWS) system, using radar measurements at four second intervals, the auxiliary particle filter outperformed the IMM filter using parallel EKFs.

Morelande et al. compared a number of different particle filters in a comparative study on their

use for the target tracking problem [34], including the transition density-based auxiliary particle filter (TD-APF) and the optimal importance density particle filter (OID-PF) and with reversed resampling (OID-PF(RR)). It was found that the OID-PF(RR) and the TD-APF outperform other particle filters and classical filters (such as the PDAF and MKF) in both efficiency and tracking performance, when tracking a single non-maneuvering target.

Li et al. take the APF tracking implementation a step further with a smoothing particle filter [25]. By stratified sampling theory, it is possible to resample a set of weighted pdfs into a single distribution. This is applied in the smoothing particle filter by generating a smoothed pdf based on the Markov switching probabilities between maneuvering modes, which in turn is used to propagate particles according to the distribution. This has the advantage of preserving the number of particles per iteration and preserving the estimation and prediction for a randomly maneuvering target. Simulation results show the smoothing particle filter approach achieves similar, if better results than the APF.

The smoothing particle filter and auxiliary particle filter are again applied to the problem of tracking a maneuvering target, this time using multiple sensors [20]. Kamel and Badawy implemented both algorithms in a multiple sensor environment, tracking a target maneuvering through the environment. Simulations showed that the smoothing particle filter performed better than the auxiliary particle filter, narrowing the target location to less than 100 meters, where the APF had an accuracy of between 100 and 400 meters.

The previous research in the field of target tracking has assumed the utilization of either active sensing techniques, like radar or laser range-finding, or passive techniques depending on RF transmissions, like TDOA measurements. In the general case, these assumptions are not always valid. A passive approach like video tracking and geo-location can be used cheaply and ubiquitously, requiring only a line-of-sight to the target and sufficient processing power to detect and track the target object in a video frame.

Gibbins et al. use video geo-location to locate a fixed point target on the ground for use in and improvement of image enhancement for further tracking and surveillance applications [13]. A tracked target in an image frame (using FFT cross correlation with sample patches) is localized by converting from image to UAV coordinates, then passing a vector through the target point, with the altitude (or range) given by the intersection with an elliptical approximation of the Earth. This 3D coordinate can be transformed from the UAV's coordinate space to GPS coordinates.

Target location estimates are merged to enhance the prediction accuracy, but this process is not explained in the study. Experimental trial flights show geo-location errors to be between 20 and 30 meters for UAV altitudes between 150 and 700 meters.

Conte et al. use video geo-location on a micro aerial vehicle (MAV) with an on-board camera [8]. In this implementation, the target location is determined by an image registration technique, a process that combines, via pattern matching, two or more images separated in time or space. After a target is detected in a video frame (based on an operator defined template), the image is matched to a reference image and the resulting image transformation is used to generate a 3D coordinate for the target. The advantage of this technique over more the classical ray-tracing approach is that the target location is calculated directly from the transformations, with no assumptions made about the altitude or range from the sensing platform based on approximate topography. Conte et al. use a recursive least squares filter to refine the estimate of the target position. The algorithm was flown on board the PingWing MAV platform in an experiment to localize a stationary ground target at an airfield. The target was localized to within 2.3m (at 30 samples) with a MAV altitude of 70m.

In 2006, Ludington developed a particle filter algorithm for tracking targets in a video stream [26, 27]. The algorithm identifies the state of the target as a rectangle bounding it in the image, with an adaptive combination of random walk and jump propagation models. Each particle is weighed by comparing the image cues contained within the rectangle bounds to a preselected sample of the target. This technique has the advantages of both detecting and tracking the target in a cluttered and noisy video stream, as well as smooth reacquisition of the target if it is lost or occluded, with the disadvantage of the processing requirements of a particle filter mitigated by use of an adaptive control over the number of particles required. In conjunction with a geo-registration or particle filter localization approach, the target's ground position can be estimated.

CHAPTER II

PARTICLE FILTER STATE ESTIMATION

2.1 PARTICLE FILTER

The particle filter is a Monte Carlo estimator used to numerically approximate integration of

$$I = \int f(x) \cdot \pi(x)\, dx, \tag{2.1}$$

where $\pi(x)$ is a probability distribution, or probability density function (PDF), of variable x restricted to $x \in \mathbb{R}^{n_x}$, $\int \pi(x)\, dx = 1$, and $\pi(x) \geq 0$. The integral resolves the expectation I with respect to the arbitrary function $f(x)$, which is most commonly expressed as a moment of x. I, in this application, is estimated about the first moment of x, the state of the target. The particle filter reduces the complexity of this integration through a combination of simplified statistical procedures, importance sampling, and numerical approximation techniques.

2.1.1 SEQUENTIAL MONTE CARLO ESTIMATION

By generating N samples $\{x^i; i = 1, \ldots, N\}$, $x^i \in x$, according to the distribution $\pi(x)$, for a sufficiently large N it can be shown [22] that the sample mean

$$I_N = \frac{1}{N} \sum_{i=1}^{N} f\left(x^i\right), \tag{2.2}$$

and it is theorized that $\lim_{N \to \infty} I_N = I$, allowing for a numerical technique to estimate I. Importantly, this means that a finite number of discrete samples may approximate $\pi(x)$.

It is, however, unlikely that $\pi(x)$ is known and thus generating samples from it is difficult. Simpifications and work-arounds of the problem exist to reduce integration complexity, such as uniform sampling or importance sampling methods like the VEGAS algorithms or the popular Metropolis-Hastings algorithm [37, 24, 31, 15]. An importance distribution, $q(x)$, is defined to be used in place of $\pi(x)$ with the requirement that $q(x)$ has the same support as $\pi(x)$. The importance distribution can be used to alter Eq. (2.1) to

$$I = \int f(x) \frac{\pi(x)}{q(x)} q(x)\, dx, \tag{2.3}$$

such that

$$I_N \approx \frac{1}{N} \sum_{i=1}^{N} f\left(x^i\right) \frac{\pi\left(x^i\right)}{q\left(x^i\right)} = \sum_{i=1}^{N} f\left(x^i\right) w'\left(x^i\right), \tag{2.4}$$

where $w\left(x^i\right)$ are the normalized importance weights generated by the likelihood ratio between the true and importance distributions [14],

$$w\left(x^i\right) = \frac{w'\left(x^i\right)}{\sum_{j=1}^{N} w'\left(x^j\right)}, \tag{2.5}$$

$$w'\left(x^i\right) = \frac{\pi\left(x^i\right)}{q\left(x^i\right)}. \tag{2.6}$$

Importance sampling has the added benefit of variance reduction during estimation. Using an importance density that is biased towards desired regions or shapes of the probability space focuses sampling on so-called important values. Through importance sampling and resampling, samples in these biased regions are emphasized, narrowing–or effectively reducing–the sampling variance. Importance resampling is discussed in a subsequent section. The likelihood ratio Eq. (2.6) differentiates between the biased and the true probability densities.

The particle filter takes advantage of the above to approximate the posterior density $\pi(x) \equiv p(\mathbf{X}_k|\mathbf{Z}_k)$, where $\mathbf{X}_k$ and $\mathbf{Z}_k$ in this application are the sequences of target states and observations, respectively, up until time k. Given the random measure $\left\{\mathbf{X}_k^i, w_k^i\right\}_{i=1}^{N}$, where $\mathbf{X}_k^i$ is a set of support points characterizing $\mathbf{X}_k$ with normalized weights w_k. $p(\mathbf{X}_k|\mathbf{Z}_k)$ can then be given, for first moment expectations, as

$$p(\mathbf{X}_k|\mathbf{Z}_k) \approx \sum_{i=1}^{N} w_k^i \delta\left(\mathbf{X}_k - \mathbf{X}_k^i\right), \tag{2.7}$$

Following the same procedure as above and defining the importance density as $q(\mathbf{X}_k|\mathbf{Z}_k) \triangleq q(x_k|\mathbf{X}_{k-1}, \mathbf{Z}_k)\, q(\mathbf{X}_{k-1}|\mathbf{Z}_{k-1})$, Eq. (2.7) can be manipulated [43] to show that

$$p\left(x_k|\mathbf{Z}_k\right) \approx \sum_{i=1}^{N} w_k^i \delta\left(x_k - x_k^i\right), \tag{2.8}$$

with normalized weights

$$w_k^i \propto w_{k-1}^i \frac{p\left(z_k|x_k^i\right) p\left(x_k^i|x_{k-1}^i\right)}{q\left(x_k^i|x_{k-1}^i, z_k\right)}, \tag{2.9}$$

or

$$w_k^i \propto w_{k-1}^i p\left(z_k|x_k^i\right), \tag{2.10}$$

if the importance density is chosen as the transition prior density, $q\left(x_k|x_{k-1}, z_k\right) = p\left(x_k|x_{k-1}\right)$, depending on the assumption that the system follows a zero-mean Gaussian process noise model; an assumption that will be applied for the problem of target tracking and geo-location.

2.1.2 IMPORTANCE RESAMPLING

The likelihood ratio relation between the importance and true probability densities Eq. (2.6) is viable when the sample sequence x_k is independent and identically distributed. Samples x_k evolve over iterations according to the importance denisty $q\left(x_k|x_{k-1}, z_k\right)$, and due to their independence will diverge over time. This divergence is called *particle degeneracy*.

It is possible to make a selection of $q\left(x_k|x_{k-1}, z_k\right)$ that constrains degeneracy [43, 37]. However, this selection is application dependent, and typical applications will see importance densities that promote, instead of limit, sample divergence, such as densities with long tails or multi-modal densities. Another approach is to re-initialize the particle filter when degeneracy occurs, but this is not desirable as information stored in the particle filter is lost, reducing the efficiency and utility of the algorithm.

Importance resampling counters the problem of particle degeneracy by regenerating the internal approximation of the PDF, $\pi\left(x\right)$. The weighted samples comprising the approximate probability distribution are shifted towards regions of expected higher probability. If degeneracy is explained

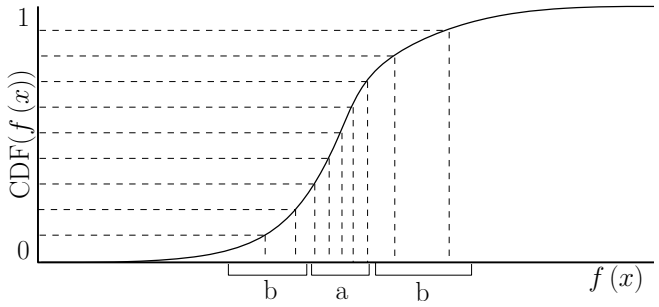


Figure 2.1: *CDF (re)sampling procedure. Region a results in more sampled particles. Regions b result in fewer sampled particles.*

Table 2.1: *Systematic Resampling Pseudocode*

Given $\{x_k, w_k\}$

- c =Cumulative sum of w_k
- $i = 1$
- Draw $u \sim U(0,1) * {}^1/_N$
- FOR $j = 1 \ldots N$
  - WHILE $\frac{j-1}{N} + u > c(i)$
    - $i = i + 1$
  - END WHILE
  - $x_k^{j*} = x_k^i$
  - $w_k^{j*} = {}^1/_N$
- END FOR

---

as samples diverging away into low probability regions, then importance resampling is gathering the samples back into the more important regions of the state space.

The goal of resampling is to draw new samples from the weighted PDF (Figure 2.1) approximated by the samples x_k and their weights w_k, and many resampling algorithms are available. The systematic resampling algorithm has been chosen for its simplicity of design and implementation with respect to the particle filter. Systematic resampling uses uniformly distributed cross-sections of the cumulative distribution function (CDF) of $\pi(x)$ to generate N new samples, x_k^*. However, to avoid costly sorting and construction of the CDF, a cumulative sum of weights (CSW) is generated from the set w_k, and x_k^* is drawn from the set x_k. After resampling, the samples x_k^* uniformly approximate $\pi(x)$, and so the weights are set to $w_k^* = {}^1/_N$. The systematic resampling algorithm is expressed in Table 2.1.

Samples that diverge away from the true probability $\pi(x)$ are denoted as x_k^f, for $\left\{ \; x_k^f \;\; | \;\; x_k^f \in x_k, \; \pi\left(x_k^f\right) \approx 0 \right\}$. All other samples are nearer the true probability and are denoted as x_k^n, for $\left\{ \; x_k^n \;\; | \;\; x_k^n \in x_k, \; \pi\left(x_k^n\right) > 0 \right\}$. As sample divergence increases, the sample weights w_k^f respective to x_k^f decrease relative to w_k^n, such that after normalization $w_k^n \gg w_k^f$. In the extreme case, the set x_k^n contains only one sample with a weight $w_k^n = 1$, with all other weights $w_k^f = 0$.

Using the measure

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} \left(w_k^i\right)^2}, \tag{2.11}$$

the effective level of degeneracy is measured as an index $N_{eff} \in [1, N]$, with smaller values corrsepsonding to higher levels of degeneracy. A large level of degeneracy, for example the extreme case with only one non-zero weight, would result in an index $N_{eff} \approx 1$. A threshold, N_{thr}, is set such that if $N_{eff} < N_{thr}$, an importance resampling cycle is triggered, which reconstitutes the approximate PDF and reduces the level of degeneracy, or raises N_{eff}. An appropriate threshold may be chosen as $N_{thr} = {}^{N}/_{3}$ [43, 36].

2.1.3 THE PARTICLE FILTER ALGORITHM

The generic particle filter can be summarized as a four step process: particle evolution, particle weighting, estimation via expectiation, and particle resampling. This process is repeated for each iteration of the particle filter, typically performed as sensor measurements arrive. An initialization step is performed before operation to generate the first particle set.

Initializing the particle filter involves assigning values to all particles according to a notion of the current state of the system, in this case the state of the target being geo-located. Particle initialization can be a long and varied task, but for the purposes of this research, the direct approach is taken. Specifically, an imprecise estimate of the target state is derived using one or more initial measurements.

With sufficient sampling or, as is described Section 3.1, knowledge of the sampling mechanism–*i.e.* the sensors–a probability distribution may be approximated for the state of the target. The initial particles, x_0, are sampled from this distribution. Because the particles already conform to the distribution, they are each assigned initial uniform weights of $w_0^i = {}^{1}/_{N}$. Although this particle

distribution may not be representive of the actual probability distribution of the target state, subsequent iterations of the particle filter will reduce such imprecision.

Each iteration of the particle filter begins with a new set of particles sampled from the importance density $q(x_k|x_{k-1}, z_k)$. This step can be simpilifed to sampling from the transitional prior $p(x_k|x_{k-1})$ if a zero mean process noise model is assumed. With a knowledge of the state space model of the target system, sampling from $p(x_k|x_{k-1})$ can be rewritten as

$$x_k = f(x_{k-1}, v_k), \tag{2.12}$$

where $f(x, v)$ is the discrete state space process update function andv_k is the process noise. For additive process noise, $f(x, v) = f(x) + v$. Particles are assumed to be independent of each other, as each is an independently sampled point in state space. Therefore, all N particles are individually evolved according to Eq. (2.12) for each iteration of the particle filter.

Note, generating new samples using zero process noise removes any randomness from the particle filter, and with it the ability of particles to converge outside the specific surfaces allowed by $f(x, 0)$, which my not intersect with the correct target states. For deterministic systems, it is better to use an alternative state estimation algorithm. However, some level of uncertainty always exists in the target system for the geo-location problem, and this uncertainty is reflected by a non-zero process noise.

The newly generated particles must be re-evaluated according to new measurements. The new particle weights are assigned according to Eq. (2.10). The likelihood $p\left(z_k|x_k^i\right)$ may be derived from a known state space model observation function, $z = h(x, w)$, where w is the observation noise. For the case of additive zero-mean white noise w, the likelihood function may be rewritten for each particle x_k^i and measurement $z_k$as

$$z_k^i = h\left(x_k^i\right) + w, \tag{2.13}$$

$$p\left(z_k|x_k^i\right) = \mathcal{N}\left(z_k^i; z_k, \sigma_{w,k}\right). \tag{2.14}$$

In this case, $\sigma_{w,k}^2$ is the known variance of the measurement z_k, and $\mathcal{N}(\cdot)$ is the Gaussian likelihood function. Substituting Eqs. (2.13) and (2.14) in Eq. (2.10), the new weight for each particle is

given as

$$w_k^i = \mathcal{N}\left(h\left(x_k^i\right); z_k, \sigma_{w,k}\right) w_{k-1}^i. \tag{2.15}$$

For cases without additive zero-mean white observation noise, a similar procedure may be followed to derive a weight update function. Following this calculation, unnormalized weights $\tilde{w}_k$ are normalized,

$$w_k = \frac{\tilde{w}_k}{\sum_{i=1}^{N} \tilde{w}_k^i}, \tag{2.16}$$

such that $\sum_{i=1}^{N} w_k^i = 1$.

After the two previous steps, an approximate PDF composed of the particles and weights exists for the current target state. The state estimation value of interest is the first moment expectation value for the state and the straightforward method for calculating this is to use a weighted mean,

$$\bar{x}_k = \sum_{i=1}^{N} x_k^i w_k^i. \tag{2.17}$$

Higher moments may also be calculated, or more sophisticated estimations may be extracted from the PDF, such as excluding invalid regions or mixing PDFs from other estimations.

Finally, a resampling step may be performed as necessary and according to the procedure defined in the previous section and Table 2.1. The PDF may be further enhanced to improve accuracy or particle filter performance, such as with a regularization pass to reduce the effects of discretization of the PDF [10, 38, 36, 2, 43].

2.1.4 PSEUDOCODE

The complete pseudocode for the general particle filter algorithm, including a resampling step, is shown in Table 2.2.

2.1.5 MULTI-MODAL PARTICLE FILTER

In complex systems, the state space model may be best represented using a Markov switching model. For example, a target state may switch between either staying constant or changing at a constant rate with a given probability that it will change between modes at a given time. This

Table 2.2: *Particle Filter Pseudocode*

Given $\{x_{k-1}, w_{k-1}\}, z_k$

- FOR $i = 1 \dots N$
  - Draw $x_k^i \sim p\left(x_k^i | x_{k-1}^i\right)$
  - $\tilde{w}_k^i = w_{k-1}^i p\left(z_k | x_k^i\right)$
- END FOR
- w_k =Normalize weights $\tilde{w}_k$
- Estimate state $\bar{x}_k$
- IF $\frac{1}{\sum_{i=1}^{N}\left(w_k^i\right)^2} < N_{thr}$, with $1 \leq N_{thr} \leq N$ chosen to prevent degeneracy
  - Resample $\left\{x_k^{i*}, 1/N\right\}$ such that $\mathtt{P}\left\{x_k^{i*} = x_k^j\right\} = w_k^j$ (see Table 2.1)
  - $\{x_k, w_k\} = \{x_k^*, 1/N\}$
- END IF

---

multi-modal behavior is difficult to represent in a simple state space update function $f(x)$. Yet, the random nature of the particle filter accomodates such models.

A Rao-Blackwell particle filter is implemented to reduce the dimension of the multi-modal state space while preserving the Markov switching behavior. The particle states, x_k, are extended to include an additional state, r_k, denoting the mode the particle–and by extension the target state–is currently operating in [35, 43]. As such, for a Markov chain with M modes, $r_k \in R$ and $R = \{1 \dots M\}$. This mode determines how each particle is updated for each iteration of the particle filter. The state space update function is similarly extended to include all relevant modes of operation:

$$f(x, r, v) = \begin{cases} f_1(x, v) & r = 1 \\ f_2(x, v) & r = 2 \\ \vdots & \vdots \\ f_M(x, v) & r = M \end{cases}. \tag{2.18}$$

The transitional probabilities defined in the system Markov chain are used to determine the switching probabilities from one mode to another for each iteration of the particle filter. All

transitional probabilties are collected into Π, an $M \times M$ matrix with each entry

$$\Pi_{ij} = p_{ij} = P\left(r_k = j | r_{k-1} = i\right), \tag{2.19}$$

belonging on row i and column j. The probability of a particle not changing modes is given when $i = j$, along the diagnoal of Π. As probability measures, then necessarily $\sum_{j=1}^{N} \Pi_{ij} = 1$. These transitional probabilities may be derived directly from the system process model, determined empirically, or adapted online or offline to achieve appropriate behavior from the particles and the final state estimation.

The particle mode is updated at each iteration either before or after the original particle state is updated. Each particle has a chance to change from its current mode to another mode according to the probabilities in Π. First, a cumulative distribution matrix, F, is constructed, such that

$$F_{ij} = \sum_{\varphi=1}^{j} \Pi_{i\varphi}. \tag{2.20}$$

Then, consider the following, with $r_{k-1} = i$ and $u_k \sim U(0, 1)$:

$$r_k = j \quad \text{if} \quad F_{ij-1} < u_k \leq F_{ij} \ . \tag{2.21}$$

This formula preserves the markov property of the system and equates to particles switching modes according to Π.

Modes may or may not be taken into account during subsequent steps of a particle filter iteration. For example, different modes of operation may require separate measurement functions, $h_r(x, w)$, or likelihood functions, $p_r(z_k | x_k)$. These are designed as needed according to the system under examination, and are trivially implemented. State estimation may also be performed differently, such as with the existence of discontinuous or disparate modes of operation. For geo-location, all modes of operation are designed to transition smoothly together, such that $f(x)$ has at least C^0 continuity despite transitions.

Particles are not directly restricted to operate in a mode of operation that matches the current target behavior, a valid mode. With finer detail in the Markov chain, the chance of a particle switching to an invalid mode may be limited, but such a specific design of the Markov chain may be impractical. Because resampling is performed independently of the mode of the particles, r_k, it is

the deciding factor in eliminating particles that switch to invalid modes. Therefore, particles may be encouraged via exaggeration of transition probabilities to switch more often than necessary. This effectively allows the particle filter to "search" both the state space and Markov chain for the correct particle state and behavior, while still restricting the search space to the most likely mode candidates.

CHAPTER III

TARGET TRACKING AND GEO-LOCATION

Geo-location is primarily a problem of geometry. After sensing and detecting an object, a geometric relation is drawn between the UAVs and targets, forming the basis for which all geo-location solutions arise. When using observation techniques that lack the critical information of range between the observers and the target, the geometric relation becomes incomplete. The particle filter geo-location approach uses assumptions about the behavior of the observers, the behavior of the targets, and assumptions about the scenario and the environment to fill in any missing information, refine the accuracy of the information, and complete the geometric geo-location solution.

3.1 SENSORS AND MEASUREMENTS

Where the problem of target sensing and detection differs from geo-location is the common lack of range information for the target. Thus, the derived information during the detection phase is the location of the target on a surface defined by the type of detecting equipment being used. For visible light and infrared cameras–EO/IR sensors–and many other sensing apparatus, this surface is a flat plane onto which the image is projected, also called an image plane or image frame. For instance, a photograph is the projection of the image a camera captures onto a flat piece of paper. A video stream is a similar projection onto a number of sequential image frames then displayed on a screen.

The precise location of the target, assuming the target is a point, on the image plane is determined by the projection geometry of the sensing device. There are two principal projection techniques that are discussed in the following paragraphs: pinhole projection and lens projection. Electronic sensors, such as charge-coupled device (CCD) cameras, may use either of these types of projection techniques, although lens cameras are more common.

The simpler type of projection is a pinhole camera projection (Figure (3.1)). This projection type uses light rays reflected from objects in a scene, passing through a point at the center of projection–a pinhole–and intersecting a plane on which a photographic plate is affixed. The distance from this plate to the pinhole is the focal length of the camera. The image is reversed in both the horizontal and vertical axes of the image frame of the plate. The image frame is defined with the origin, $(0, 0)$, at the center of the captured image, the $+x$ axis along the horizontal line

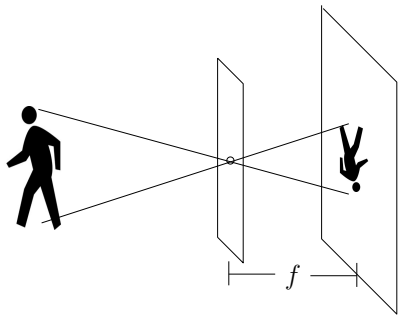


Figure 3.1: *Pinhole camera projection with focal length f.*

of the image and to the right of the frame, and the $+y$ axis along the vertical line of the image and to the top of the frame. In reality, pinhole cameras produce blurry images due to convolution of the light rays passing through a finite aperture, with smaller apertures causing diffraction of the light. Image processing algorithms, such as deconvolution, may reduce this blurring effect.

In computer graphics, projection is accomplished with a pinhole camera type projection with a negative focal length f or an image frame that exists between the center of projection and the scene at coordinates $z = -f$ [12]. When using a homogeneous coordinate system, this simplifies projection to a simple affine transformation. In this system, a point location in three dimensions is given as

$$p = \begin{bmatrix} p_x & p_y & p_z & 1 \end{bmatrix}^T. \tag{3.1}$$

Assuming p lies in the camera frame of reference[1], the pinhole projection of p is written as $p^I = \begin{bmatrix} p_x^I & p_y^I & p_z^I & 1 \end{bmatrix}^T = W(P * p)$, where P, the projection matrix, is given as

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/f & 0 \end{bmatrix}, \tag{3.2}$$

and the function $W(\cdot)$ returns a coordinate vector to homogeneous unity by dividing the vector by its fourth coordinate. Therefore,

$$p^I = W(P * p) = W\left(\begin{bmatrix} p_x & p_y & p_z & \frac{p_z}{f} \end{bmatrix}^T\right) = \begin{bmatrix} \frac{fp_x}{p_z} & \frac{fp_y}{p_z} & f & 1 \end{bmatrix}^T \tag{3.3}$$

[1] The camera frame of reference has the origin at the center of projection, the z axis along the direction the camera is facing, and the y axis along a fixed, arbitrarily chosen direction, usually denoted as the "up" direction, or "up vector."

relates a point in three dimensional space to a location on the image frame. The pinhole projection type is used in the simulations in Chapter 5.

The lensed camera addresses the problem of image blurring found with pinhole cameras by focusing the gathered light rays onto a point. However, this focusing effect only occurs for objects at a given distance from the camera, blurring objects at any other distances. For a camera with a single, thin lens of focal length f and a movable image plane located at p_z^I, the optimum object distance is given by

$$p_z^* = \frac{f p_z^I}{p_z^I - f}, \tag{3.4}$$

and the projected point for objects at this distance will lie on the image plane at

$$p_x^I = \frac{f}{p_x \left(p_z^* - f\right)}. \tag{3.5}$$

A single lens produces a number of optical artifacts on the image plane, such as various scattering and blurring effects. Most artifacts may be reduced or eliminated by using a compound lens system along with lens coating. Due to the varying sizes, refractive indices, and placement of the lenses in cameras, the construction of a complete projection transformation is dependent on the design of the lensed camera in use.

Notice that the projected point in both cases depends on the distance from the center of projection to the location of the object along the z axis. It is also true that with knowledge of this range information, these transformations can be reversed to recover the location of an object in the camera space. Without this critical range information, it is not possible to perform the reverse transformations, necessitating a more complex geo-location procedure.

A three-dimensional object has a non-zero volume, and therefore must be represented in the image frame as more than a single point. The metrics of interest are based on the assumption that objects are roughly ellipsoid, with a center of mass located around the volumetric centroid of the object. In the two-dimensional projection in the image frame, the object may be defined as having a height and width along the horizontal and vertical axes of the image plane, respectively, that bound the object inside a rectangle, with a location given at the centroid of this rectangle (Figure 3.3). Therefore, an object in the image frame has coordinates

$$p' = \begin{bmatrix} p_x^I & p_y^I & s_x & s_y \end{bmatrix}^T. \tag{3.6}$$

In this context, s_x and s_y are the width and height of the bounding rectangle, respectively. Any uncertainty in these measures, perhaps arising from image artifacts and blurring or from uncertainty and error in the detection method, may be transformed by these same reverse transformations, producing a similar uncertainty in the location of the object in the camera space.

3.2 TRACKING GEOMETRY

The geometric structure of the range-less geo-location problem begins with the triangle and the law of cosines. The law of cosines governs the relationship between the three edge lengths and three inner angles of a triangle. Importantly, if at least two inner angles and one edge length, or at least two edge lengths and one inner angle are known, then all inner angles and edge lengths may be derived. In geo-location, this is translated to knowing the location of two fixed observers or UAVs, and the direction vector from each observer to a target at an unknown location. This is the basis for the triangulation method of localization (Figure 3.2).

Because measurements are taken in the image frame, they must be converted to the space-fixed, inertial frame which the observers and the target occupy. This is a simple multi-step process involving a series of translation and rotation matrices. The initial image frame location, p^I, is reconstituted from the measures in Eq. (3.6), and converted to the camera space using $p^C = W^{-1}\left(p^I, p_z\right)$, the inverse of the projection function. Now, p^C is the location of the target in the camera space, and the remaining transformations involve a series of multiplication of translation and rotation matrices. The UAV is a moving, rotating body in the space-fixed frame, and the camera is a self-rotating body that is not centered on the UAV, and so each of these aspects is addressed in the final transformation.

The complete transformation procedure is

$$p^S = T_U R_U T_C R_C p^C, \tag{3.7}$$

where the translation, T, and rotation, R, matrices are defined as

$$T = T\left(d\right) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.8}$$

and

$$R = R(\phi,\theta,\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 & 0 \\ \sin\psi & \cos\psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta_k & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta_k & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi & 0 \\ 0 & \sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3.9}$$

where $d = \begin{bmatrix} d_x & d_y & d_z \end{bmatrix}^T$ is the displacement from the local coordinate system origin–*i.e.* d_C is the displacement of the camera from the center of the UAV–and ϕ, θ, and ψ are the roll, pitch, and yaw with respect to the parent frame–*i.e.* R_C resolves the orientation of the camera with respect to the UAV. As the camera is assumed to have no roll component, then it is assumed that $\phi_C = 0$. This operation amounts to the target location in camera frame to coincide with the UAV body frame, translating to the UAV body frame, rotating to the space-fixed frame, then finally translating to the space-fixed frame.

The space-fixed frame is an arbitrary inertial frame with an origin at a fixed point on the earth, for example a nearby structure or point of interest with a well-known position in the positioning frame of choice. For example, using the space-fixed coordinate relative to a GPS marker, the target location may be transformed easily to GPS coordinates for precise earth-fixed location. And in fact, as long as any fixed point is known in this space-fixed system, any Earth-fixed Earth-centered (ECEF) or Earth-centered inertial (ECI) coordinate system may be used for global positioning or referencing. This final step is dependent on the application domain that the UAVs belong to and is not covered here.

Note, the range information is critical to the transformation process. The initial conversion from the image frame to the camera frame depends on knowing p_z, which is assumed to be unknown. Alternatively, a direction to the target may be derived from the same coordinate transformation. A value is chosen, $\hat{p}_z$, such that the vector from the center of projection to $\hat{p}^I = \begin{bmatrix} p_x^I & p_y^I & \hat{p}_z \end{bmatrix}^T$ is parallel to the vector from the center of projection to $\begin{bmatrix} p_x & p_y & p_z \end{bmatrix}^T$. The point $\hat{p}^I$ is transformed to the camera frame, $\hat{p}^C = W^{-1}\left(p^I, \hat{p}_z\right)$, and then to the space-fixed frame using the same transformation procedure in Eq. (3.7). The origin of the camera frame p_0^C is transformed as well.

Figure 3.2 sets up the basic triangulation problem. A vector may be formed from these two points in the space-fixed frame, along which it is known that the target (T) must lie. This vector, when

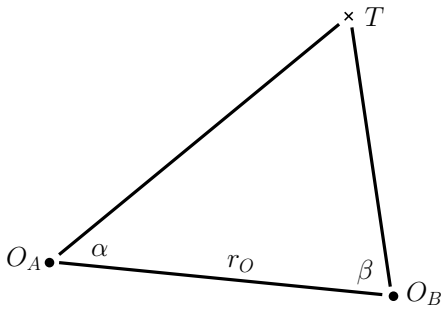


Figure 3.2: *Basic triangulation problem.*

used with two UAVs ($O_A$and O_B) or two separate measurements from a single UAV, may be used in the triangulation problem. The two vectors may be extended to form each edge of the triangle in Figure and the resulting angles (α and β) known from the direction of the vectors completes the requirements for the law of cosines. Uncertainty in image detection translates to a narrow volume within which the target may be located, with an associated volumetric probability density function (PDF), and the intersection of two or more volumetric PDFs form an approximate PDF of the location of the target.

Volumetric PDFs, however, are difficult to define analytically and mix together. The particle filter geo-location scheme aggregates the same information and generates a similar PDF of the target location. The particle filter also integrates the target behavior to modify the PDF according to the target motion.

3.3 STATE SPACE CONSTRUCTION

The state space model describes the behavior of the intended target, necessary for estimation and prediction of the target's behavior. The localization system can use this to match inaccurate or incomplete observations to the expected observations according to the target's motion and measurement model, collectively called the state space model, taking the general form

$$\frac{dx\left(t\right)}{dt} = f\left(x\left(t\right), v\left(t\right), t\right), \tag{3.10}$$

$$z\left(t\right) = h\left(x\left(t\right), w\left(t\right), t\right), \tag{3.11}$$

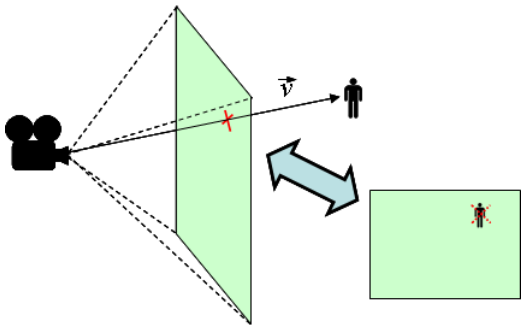


Figure 3.3: *Camera projection example.*

$f(\cdot)$is the transfer function, propagating the state in time, $z(t)$ the measurement function, and $v(t)$ and $w(t)$ the process and observation noise/disturbance, respectively.

Here, the state $x(t)$ contains information on the target, like its position and velocity. In two dimensions, this can be represented as

$$x(t) = \begin{bmatrix} x^{\tau}(t) & y^{\tau}(t) & \dot{x}^{\tau}(t) & \dot{y}^{\tau}(t) \end{bmatrix}^{\mathrm{T}}, \tag{3.12}$$

where (x^{τ}, y^{τ}) are the world-space coordinates of the target in two dimensions, and $(\dot{x}^{\tau}(t), \dot{y}^{\tau}(t))$ the target velocity.

The transformed point, $\hat{x}_k^S$, defines the observation of the target in the space-fixed coordinate system of choice: in this case, the coordinate system in which the UAV lies. The system could be GPS coordinates, or relative to a local origin like a locator beacon, with the assumption that the coordinates can be locally defined as Cartesian. Because this observation lies on a ray cast from the camera center of projection (Sec (3.1)), intersecting the target location, a unit vector $v = x_{0,k}^S - \hat{x}_k^S$, $\overrightarrow{v} = {}^{v}/_{\|v\|}$can be constructed that describes the direction (angle) to the target, as in Figure 3.3. Thus,

$$\overrightarrow{v}(t) \equiv z(t) = h(x(t), x_U(t), x_C(t), w(t), t), \tag{3.13}$$

All that remains is an understanding of the target's expected dynamics, in the form of $f(\cdot)$, to complete the model, allowing for a more accurate estimation of the target state $x(t)$.

Simple Linear Model — The simplest assumption, besides a completely stationary target, is a target with a fixed velocity. In two dimensions, the discretized transfer function is

$$x_k = f\left(x_{k-1}, v_{k-1}\right) = Ax_{k-1} + Bv_{k-1}, \tag{3.14}$$

$$A = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; B = \begin{bmatrix} \frac{\Delta T^2}{2} & 0 \\ \frac{\Delta T^2}{2} & 0 \\ 0 & \Delta T \\ 0 & \Delta T \end{bmatrix}, \tag{3.15}$$

where ΔT is the time interval between discrete samplings.

This model is straightforward and easy to implement. However, it is an unrealistic assumption except in a few special cases. Objects such as fixed-wing aircraft or ground vehicles on a long, straight road may exhibit this behavior of a fixed-velocity for brief periods. Ballistic trajectories may be locally linearized in such a manner, with effective accuracy given a small enough ΔT. A target with sufficient acceleration relative to its velocity, such as a person rounding a corner during a run, could confuse an estimator using this expected model.

The model can be adjusted with the addition of a third term B_a:

$$x_k = f\left(x_{k-1}, v_{k-1}\right) = Ax_{k-1} + Bv_{k-1} + B_a, \tag{3.16}$$

where

$$B_a = \begin{bmatrix} 0 & 0 & a_x & a_y \end{bmatrix}^T. \tag{3.17}$$

For $a = (a_x, a_y)$, this adds a constant acceleration to the simple linear model. By using this in conjunction with the constant velocity model, actions such as the target speeding up, slowing down, and stopping may be accounted for. The target state space model is also updated to allow the particle filter to adjust the acceleration a:

$$x = \begin{bmatrix} x^\tau & y^\tau & \dot{x}^\tau & \dot{y}^\tau & a_x & a_y \end{bmatrix}^T. \tag{3.18}$$

Non-linear Model — A target may frequently exhibit specific non-linear behavior, such as during a maneuver like a coordinated turn. For example, an aircraft may execute a turn at a constant speed and angular velocity, Ω. Its state can then be modeled as [3]

$$x_k = f\left(x_{k-1}, v_{k-1}\right) = A\left(\Omega\right) x_{k-1} + B v_{k-1}, \tag{3.19}$$

$$x = \begin{bmatrix} x^\tau & y^\tau & \dot{x}^\tau & \dot{y}^\tau & \Omega \end{bmatrix}^{\mathrm{T}}, \tag{3.20}$$

$$A\left(\Omega\right) = \begin{bmatrix} 1 & 0 & \frac{\sin(\Omega\Delta T)}{\Omega} & -\frac{(1-\cos(\Omega\Delta T))}{\Omega} \\ 0 & 1 & \frac{(1-\cos(\Omega\Delta T))}{\Omega} & \frac{\sin(\Omega\Delta T)}{\Omega} \\ 0 & 0 & \cos\left(\Omega\Delta T\right) & -\sin\left(\Omega\Delta T\right) \\ 0 & 0 & \sin\left(\Omega\Delta T\right) & \cos\left(\Omega\Delta T\right) \end{bmatrix} \quad B = \begin{bmatrix} \frac{\Delta T^2}{2} & 0 \\ \frac{\Delta T^2}{2} & 0 \\ 0 & \Delta T \\ 0 & \Delta T \end{bmatrix}. \tag{3.21}$$

The state space model has been extended to include the target's angular velocity, on which the state matrix, A, has a non-linear dependence. The dynamics of this mode of behavior most closely describes movement along the arc of a circle with radius $r = v/\Omega$, for a constant tangential velocity v. This behavior may approximate an aircraft during a turning maneuver, a ballistic trajectory over short time periods, a car making a turn at an intersection or during all or part of a curve in a road, or other target behaviors. While allowing more freedom of movement and covering more possible target behaviors, the model still assumes a constant activity, in this case governed by a constant angular velocity.

Maneuvering Model — This model attempts to account for changes in the target's behavior by extending or parameterizing the desired model in such a way as to predict abrupt changes in dynamics. The changes may consist only of allowing for a variable constant, such as turn acceleration; or of adding entirely new dynamic behavior to a preexisting model, such as extending the simple linear model to include turns, or maneuvers; or a combination of the two.

In any case, to enable prediction via posterior construction (see Section 2.1), the probability that a maneuver may be made must be known. This is known as a jump Markov model, in that

the probability that a given maneuver will exit into another maneuver is subject to a Markov switching system [5, 48].

In formal terms,

$$x_k = f_k\left(x_{k-1}, m_k, v_{k-1}\right), \tag{3.22}$$

$$z_k = h_k\left(x_k, m_k, v_k\right), \tag{3.23}$$

where m_k is the model variable, representing the model that is in effect at time k. The probability of transition from any model $m \in M$, where M is the set of n model identifiers $M = \{1, 2, \ldots, n\}$, is defined as

$$\pi_{i,j} \triangleq \mathbf{P}\left\{m_k = i | m_{k-1} = j\right\} \ (i, j \in M), \tag{3.24}$$

with $\pi_{i,j}$ forming the elements of an $n \times n$ transition matrix Π.

In a jump Markov linear model, for example, Eqs. (3.22) and (3.23) can be expressed as

$$x_k = A_{k-1,m_k} x_{k-1} + v_{k-1,m_k}, \tag{3.25}$$

$$z_k = C_{k,m_k} x_k + w_{k,m_k}, \tag{3.26}$$

which highlights that the state space model now has a dependence on the currently selected model m_k.

Estimators like the particle filter can be extended to account for multiple models subject to a Markov switching system with known transition probability matrix, Π. Some solutions [21, 25, 48] include splitting particles to cover probable maneuvers or changes in maneuver at time k, or generating and resampling from a posterior that encompasses the probable maneuvers at time k.

3.4 PARTICLE FILTER GEO-LOCATION

The information presented in this section is used to complete the particle filter design. The state space models for the UAV and target are used to update the particles, along with the multi-modal behavior of the target. The observation model derived from the camera design and the camera space transformation adjusts measurements to match the UAV camera perspective. Finally, the likelihood function used to weight the particles is determined by the observation uncertainty, dependent on the camera and the detection system in use.

3.4.1 PARTICLE EVOLUTION

Each particle in the particle filter represents a guess on the current state of the target. The complete state is written as

$$x = \begin{bmatrix} x^\tau & y^\tau & z^\tau & \dot{x}^\tau & \dot{y}^\tau & \dot{z}^\tau & a_x & a_y & a_z & \Omega \end{bmatrix}^T. \tag{3.27}$$

This state encapsulates the position and velocity of the target as the primary measures of interest. The acceleration and angular acceleration are only used in such cases where the multi-modal behavior requires them, as is shown in the following paragraphs. All measures have been updated from two to three dimensions with the addition of z^τ, $\dot{z}^\tau$, and a_z, and the changes to the state space models are trivial. It is assumed that all turns will happen in roughly two-dimensions, or on a plane, and so the angular acceleration remains a scalar.

The following target state space models are used in conjunction within a multi-modal framework:

$$f_1(x_k, v_k) = Ax_k + Bv_k, \tag{3.28}$$

$$f_2(x_k, v_k) = Ax_k + Bv_k + B_a, \tag{3.29}$$

$$f_3(x_k, v_k) = A(\Omega)\, x_k + Bv_k. \tag{3.30}$$

More complex varieties of state space model are available, and many are dependent on the type of vehicle being tracked. The three models of constant velocity (Eq. (3.14)), constant linear acceleration (Eq. (3.16)), and constant angular acceleration (Eq. (3.19)) capture the most likely behaviors of any object in motion. All process disturbances, v_k, are considered to be additive.

The Markov switching behavior is achieved through use of the multi-modal particle filter and the

mode state r_k, for $r = \{1, 2, 3\}$. The final particle update function is

$$f(x_k, r_k, v_k) = f_r(x_k, v_k). \tag{3.31}$$

The mode transition probability matrix Π is constructed from empirical study of the target of interest, measuring the probability that a target will continue, slow down, speed up, stop, turn, or perform other actions under various circumstances.

During the particle evolution stage of the particle filter execution, each particle will mimic the expected behavior of the target according to its own guess of the target state. A particle that assumes the target is moving at 50 km per hour to the east will adjust itself each time step according to this assumption, with an additional random disturbance. With more particles evolving as such over time, it becomes more likely that one or more particles will reach a state coinciding with the measurements taken of the target.

3.4.2 PARTICLE WEIGHTING

Each particle may be considered to be a virtual target itself. The location of the virtual target may be extracted and used to create a virtual measurement according to Eq. (2.13). Because the target state is estimated in the space-fixed coordinate system, it must first be transformed to the camera frame to accurately simulate the camera behavior. The same procedure to transform from the camera frame to the space-fixed frame Eq. (3.7) is reversed to perform the converse transformation.

In practice, the individual rotation and translation matrices may be inverted to create the separate reverse translations. However, it may be shown that by inverting the resultant 4×4 transformation matrix T, the entire reverse transformation is recovered as well [9]. For

$$p^S = T_U R_U T_C R_C p^C = \bar{T} p^C, \tag{3.32}$$

and $p^S = \begin{bmatrix} x^\tau & y^\tau & z^\tau & 1 \end{bmatrix}^T$ the reverse transformation from the space-fixed frame to the camera frame is

$$p^C = R_U^{-1} T_U^{-1} R_U^{-1} T_U^{-1} p^S = \bar{T}^{-1} p^S. \tag{3.33}$$

With the virtual target state in the camera frame of reference, the same projection matrix as-

sociated with the camera in use is applied. A pinhole camera model is used in the simulations, so Eq. (3.3) is applied, with $p = \begin{bmatrix} p_x^C & p_y^C & p_z^C & 1 \end{bmatrix}^T$. The entire transformation procedure is coincident with the observation function of the state space model $h(\cdot)$, which is also mentioned in Eq. (2.13), with an observation noise of zero. This noise is relevant to the particle weighting according to the likelihood function.

These two operations–frame transformation and projection transformation–are performed for each particle, resulting in a virtual measurement associated with each guess of the target state. As with the actual measurements, an amount of uncertainty is associated with these measurements according to errors in the location and orientation of the UAV, as measured by the on-board GPS and IMU; error in the orientation of the camera sensor, as measured by encoders on the positioning servos or the camera gimbal; and uncertainty and error associated with detection of the target in the image frame, which is dependent on the detection algorithm in use.

These errors and uncertainties accumulate over the successive transformations. Unfortunately, the probability distribution of the product of two or more random variables is difficult to describe analytically. Alternatively, the PDF associated with the complete transformation may be roughly approximated using a sum of Gaussian distributions. And, it is assumed that the measurement errors and uncertainties conform to distributions that are linearly independent. The sum of independent Gaussian distributions,

$$\sum_{i=1}^{N} \mathcal{N}\left(\mu_i, \sigma_i^2\right) = \mathcal{N}\left(\sum_{i=1}^{N} \mu_i, \sum_{i=1}^{N} \sigma_i^2\right), \tag{3.34}$$

is a Gaussian distribution with first and second moments equal to the sum of the first and second moments of the initial distributions [16].

The complete probability distribution due to uncertainty and error in measurement may be reduced to an approximate Gaussian distribution. This forms the basis for the likelihood function. Now, Eq. (2.10) is used to weight each particle, $i = \{1...N\}$, according to the actual measurement, p^I–the location of the target in the image frame–and the virtual measurement of the particle, $p^{I,i}$:

$$w_k^i = \mathcal{N}\left(p_k^{I,i}; p_k^I, \Sigma_k\right) w_{k-1}^i, \tag{3.35}$$

where Σ is the covariance matrix of the Gaussian distribution approximating the uncertainty of the measurements.

Table 3.1: *Particle Filter Geo-Location Pseudocode*

Given $\{x_{k-1}, r_{k-1}, w_{k-1}\}, p_k^I$

- FOR $i = 1 \ldots N$
  - $x_k^i = f\left(x_{k-1}^i, r_{k-1}^i, v_{k-1}^i\right) = f_r\left(x_{k-1}^i, v_{k-1}^i\right)$
  - $p_k^{C,i} = \bar{T}^{-1} \cdot \left[\begin{array}{cccc} x_k^{\tau,i} & y_k^{\tau,i} & z_k^{\tau,i} & 1 \end{array}\right]^T$
  - $p_k^{I,i} = W\left(P * p_k^{C,i}\right)$, for pinhole projection transformation $W(\cdot)$
  - $\tilde{w}_k^i = \mathcal{N}\left(p_k^{I,i}; p_k^I, \Sigma_k\right) w_{k-1}^i$
- END FOR
- w_k =Normalized weights $\tilde{w}_k$
- Estimate state $\bar{x}_k = \sum_{i=1}^{N} \left[\begin{array}{cccc} x_k^{\tau,i} & y_k^{\tau,i} & z_k^{\tau,i} \end{array}\right]^T w_k^i$
- IF $\frac{1}{\sum_{i=1}^{N}\left(w_k^i\right)^2} < N_{thr}$, with $1 \leq N_{thr} \leq N$ chosen to prevent degeneracy
  - Resample $\left\{x_k^{i*}, 1/N\right\}$ such that $\mathtt{P}\left\{x_k^{i*} = x_k^j\right\} = w_k^j$ (see Table 2.1)
  - $\{x_k, w_k\} = \{x_k^*, 1/N\}$
- END IF

3.4.3 STATE ESTIMATION

The final step in producing a state estimate is to take the expected value according to the approximate distribution of the N weighted particles. The measure of interest is the first moment of this distribution, which may be extracted as the weighted mean of the particles:

$$\bar{x}_k = \mathrm{E}\left[x_k\right] = \sum_{i=1}^{N} \left[\begin{array}{ccc} x_k^{\tau,i} & y_k^{\tau,i} & z_k^{\tau,i} \end{array}\right]^T w_k^i. \tag{3.36}$$

Additional moments and measures may be collected from the particles for further use, such as to gauge the expected accuracy of the particle filter state estimation.

3.4.4 PARTICLE FILTER GEO-LOCATION PSEUDOCODE

The complete pseudocode for the particle filter geo-location algorithm is shown in Table 3.1.

3.5 MULTIPLE OBSERVER (UAV) EXTENSION

The framework presented above allows for a variable length of observation structure z_k at each time step k. This makes the addition of one or more observers, in this case UAVs, to the scenario simple and quick. For a single observer, $p\left(z_k|x_k\right)$ in Eq. (2.10) can be assumed to be

$$p\left(z_k|x_k^i\right) = \mathcal{N}\left(h\left(x_k^i\right); z_k, \sigma_z^2\right) \tag{3.37}$$

with σ_z^2 the nominal accuracy of measurement z_k, and $\mathcal{N}\left(x; m, \sigma^2\right)$ the normal probability density of x with mean m and variance σ^2. With n observers, this is simply

$$p\left(z_k|x_k^i\right) = \mathcal{N}_n\left(H\left(x_k^i\right); Z_k, \Sigma_z\right) \tag{3.38}$$

$$H\left(x_k^i\right) = \left[\begin{array}{cccc} h_1\left(x_k^i\right) & h_2\left(x_k^i\right) & \cdots & h_n\left(x_k^i\right) \end{array}\right]^{\mathrm{T}} \tag{3.39}$$

$$Z_k = \left[\begin{array}{cccc} z_{k,1} & z_{k,2} & \cdots & z_{k,n} \end{array}\right]^{\mathrm{T}} \tag{3.40}$$

where Σ_z is the $n \times n$ covariance of the $n \times 1$ measurement Z_k, $\mathcal{N}_n\left(\cdot\right)$ is the multivariate normal density function, and $h_o\left(\cdot\right)$ is the observation function for each observer for $o \in \{1, 2, ..., n\}$.

All that remains is defining the multiple measurements Z_k and $h_o\left(x_k^i\right)$ and their associated covariance Σ_z. For multiple UAVs, this can be as simple as concatenating each successive UAV measurement $z_{k,o}$, assuming they are independent, and creating a diagonal Σ_z such that the diagonal values are the measurement accuracy $\sigma_{z,o}^2$ for each UAV. For multivariate measurements $z_{k,o}$, $\sigma_{z,o}^2$ becomes a covariance matrix $\Sigma_{z,o}$, and Σ_z becomes a block diagonal covariance matrix.

The particle filter estimator is thus quickly scalable with the addition of UAVs. For a multivariate system with n observers and block diagonal covariance

$$\Sigma_z = \begin{bmatrix} \Sigma_{z,1} & 0 & \cdots & 0 \\ 0 & \Sigma_{z,1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Sigma_{z,n} \end{bmatrix} \tag{3.41}$$

$p\left(z_k|x_k\right)$ can be reduced to

$$p\left(z_k|x_k\right) = \mathcal{N}_n\left(H\left(x_k^i\right); z_k, \Sigma_z\right) = \prod_{o=1}^{n} \mathcal{N}\left(h_o\left(x_k^i\right); z_{k,o}, \Sigma_{z,o}\right) \tag{3.42}$$

Thus, each new UAV adds only density estimation and one product to the weight calculation, keeping complexity under control while improving target location estimation accuracy.

Any likelihood function derived from expected value operator $\mathrm{E}\left(f\left(X\right)\right)$ may be similarly reduced to the product of individual likelihood functions. In general, the likelihood function is only multiplicative across random variables when these variables are independent, or uncorrelated. In terms of uncorrelated sets of variables, the covariance between these sets is $\mathrm{cov}\left(X, Y\right) = 0$. By the definition of covariance, $\mathrm{cov}\left(X, Y\right) = \mathrm{E}\left(XY\right) - \mathrm{E}\left(X\right)\mathrm{E}\left(Y\right)$, a covariance of zero between X and Y results in $\mathrm{E}\left(XY\right) = \mathrm{E}\left(X\right)\mathrm{E}\left(Y\right)$.

The issue of correspondence, or the misidentification of one or more targets within a scenario, is not addressed in this research. It is assumed that only one target exists at all times in the region of interest, so measurements from multiple observers are accepted as measurements of the same target.

3.6 TARGET DYNAMICS

The target behavior integrates into the particle filter primarily in Eq. (2.12). It is through the state space model update function that particles are informed how to behave, and by matching the particle and target behavior, the particle filter estimate may more closely track the target state. As the observation function transforms the target coordinates to a coordinate frame coincident with camera measurements (Section 3.2), target motion only needs to be represented through the particle update function.

The target behavior modeled for this research is that of a car. The car is a non-holonomic system that accepts a thrust command and an angular velocity command. However, the target state space model is simplified further to introduce use of the maneuvering model (Section 3.3). The car system is represented as a simple Newtonian system with three types of behavior: constant velocity, constant acceleration, and turning with constant velocity (Eqs. (3.28)-(3.30)).

The non-holonomic behavior of the car is recreated through a set of state space modes S. This set includes stopping, accelerating, decelerating, coasting, and turning:

$$S = \{\text{STOP, ACCEL, DECEL, COAST, TURN}\}. \tag{3.43}$$

During simulation, the target switches behavior between these modes according to a rule set defined below. For each mode switch, the previous target state is preserved, only the update function and its parameters are changed. However, the parameters are derived from the state at the time of the mode switch, such that acceleration and deceleration occur along the current velocity vector or on a vector matching the target orientation when stopped. In this manner, non-holonomicity of the target system is preserved.

The switching between modes in set S is tuned to mimic the driving conditions of the target and to preserve at least C^1 continuity. Different driving patterns, such as driving in the city, on highways, in the desert, *etc.*, will result in different switching frequencies and patterns. Here, a suburban driving pattern is assumed, with frequent stops and turns to simulate intersections, inconsistent traffic patterns, and fewer instances of high speed driving.

Figure 3.4 diagrams the Markov switching system with the possible mode switching allowed. Dashed lines indicate a lower switching probability than full lines. Stopping may only be reached after decelerating to a stop, and other modes may only be reached from a stop after first accelerating.

The driving mode diagram in Figure 3.4 is used to construct a transition probability matrix Π that will be used as a basis for the target maneuvering model. Each transition line in the diagram translates to a probability $\pi_{i,j}$, from Eq. (3.24). These are tuned through experimentation and simulation to most closely match the desired target behavior. Table 3.2 is the final tuned transition matrix Π.

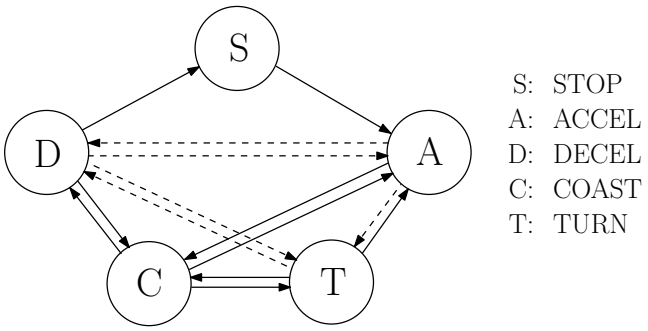


Figure 3.4: *Driving modes and Markov switching system.*

Table 3.2: *Target transition probability matrix.*

$$\Pi = \begin{bmatrix} 0.995 & 0.005 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.992 & 0.002 & 0.004 & 0.001 \\ 0.003 & 0.001 & 0.991 & 0.002 & 0.002 \\ 0.000 & 0.002 & 0.002 & 0.995 & 0.001 \\ 0.000 & 0.003 & 0.001 & 0.002 & 0.993 \end{bmatrix}$$

This matrix is used to form the rule base for switching driving modes only. The switching behavior is extended with additional rules enforcing a maximum target velocity of $80^{\mathrm{km}/\mathrm{h}}$, minimum and maximum turning radii, and minimum and maximum height values (target motion along the z-axis).

3.6.1 MULTIPLE TARGET EXTENSION

The research presented in this thesis focuses mainly on single target detection and geo-location. However, multiple targets of interest may be in the same scene. Tracking multiple targets at the same time is a difficult problem requiring many additions and enhancements to the geo-location particle filter. This section notes a few possible routes for achieving multiple target geo-location estimates, however, the implementation of a multiple target tracking system is beyond the scope of this research.

Initially, each target must be detected and differentiated using the sensors and imaging system. For video sensors, this may include a more advanced segmentation algorithm, or background subtraction and optical flow algorithms. For radar or other active sensors, TDOA measurements may be analyzed and classified according to the likelihood of individual detection versus false alarm [45].

Once the multiple targets have been detected and separated, the geo-location particle filter must apply the measurements towards evaluating the likelihood of particles estimating the state of each

of the targets of interest. A naive approach to redesigning the particle filter might then be to use a separate geo-location particle filter for each target estimation. Assuming a detection and differentiation algorithm with bounded error, this eliminates the chances of losing target track or confusing targets due to sensor uncertainty. However, duplicating the particle filter for each target is costly.

Another possibility is to extend the target state to encompass all targets in the region of interest. Particles may then individually obtain both good state estimates and bad state estimates. Particle weights may be extended to contain information about the target states for which a particle has a good track, with an appropriate resampling algorithm that splits and merges particles accordingly. Groups of particles thus track groups of targets, with the ability for particles to switch groups as they gain and lose track of targets. With this method, starvation may occur, where a target may have very few or no particles tracking it.

A compromise between the above two systems is to assign particles to one or more targets. Each particle accepts measurements for and tracks its assigned target, and is weighted against particles tracking the same targets. During resampling, particles with poor tracks on one or more targets are eliminated as usual. Particles with good tracks are duplicated and extended with new target assignments. This method avoids both problems of having multiple particle filters to track each target, and the problem of target starvation.

CHAPTER IV

UAV AND CAMERA CONTROL

4.1 INTRODUCTION

The UAV investigated in this thesis is the main representative of the rotorcraft family which is the helicopter. Such rotorcraft UAVs of this design include the Yamaha R-MAX or the Boeing A160 Hummingbird. The rotorcraft platform is chosen for its versatility of flight modes, including hovering, flying at low speeds, and orbiting with a small radius. In the recent years there has been considerable research related to the helicopter autonomous flight control problem [23, 17, 30, 41, 42].

Many varieties of sensors are available to detect and track targets, such as video and infrared devices, radar, and LIDAR. Passive sensing techniques like video and infrared are of interest for tracking applications due to their low or nonexistent emissions, high resolution, and wide operating range. Part of the UAV path planning or control algorithms must be devoted to pointing the video or infrared sensors at the target, and ensuring the UAV is within a desired range of the target.

Gimbaled camera systems grant the ability to aim a camera sensor in a desired direction, and have been used on-board UAVs for target tracking systems [13, 40]. The UAV controller is designed with the assumption that the camera gimbal will point the camera to the target, taking into account the direction of the target and the orientation of the UAV.

It is accepted that under certain flight conditions the UAV body itself may intersect the line-of-sight from the camera to the target, possibly occluding the target. The suggested UAV controller does not attempt to orient the UAV to prevent camera occlusion, which under certain conditions may be difficult or impossible (*i.e.* steep tilt angles that result in lateral UAV motion). However, the controller design attempts to limit the roll and pitch of UAV for stability purposes, and assuming the target is always at a lower height than the UAV, it is safe to assume the target is always within the visual range of the camera and gimbal system.

This chapter address the dual problem of rotorcraft control and trajectory generation for the tracking of a moving target. The flight controller is designed based on the rotorcraft nonlinear dynamics that include the basic equations of motion and a simplified model of the torque and force generation. The controller considers the discrete rotorcraft dynamics to accommodate the

discretization effects that emerge from the execution of the algorithm by a microprocessor. The flight controller is utilizing a backstepping approach for systems in feedback form. The intermediate control signals (a.k.a. pseudo controls) for each level of the feedback system are appropriately chosen to stabilize the helicopter dynamics. The goal of the proposed trajectory generator is to provide the rotorcraft with an additional level of autonomy in moving target tracking missions. The trajectory generator is composed by two modes depending on the motion of the target. When the target is moving the rotorcraft is kept within a specified sensor range. When the target is stationary the rotorcarft is subject to orbiting motion around a fixed radius from the target. The applicability of the combined controller-trajectory generator module is successfully verified through numerical simulations.

4.2 HELICOPTER MODEL

4.2.1 MATHEMATICAL NOTATION

For simplicity, the abbreviations C_t, S_t, and T_t with $t \in \mathbb{R}$ represent the trigonometric functions $\cos(t)$, $\sin(t)$, and $\tan(t)$, respectively. For a vector $\vec{w} = \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix}^T \in \mathbb{R}^3$, the notation $\hat{\vec{w}}$ denotes the skew-symmetric matrix of the vector. The operand $\|(\cdot)\|$ denotes the Euclidean norm.

4.2.2 ROTORCRAFT SYSTEM DYNAMICS

The first step toward the development of the rigid body's equation of motion is the definition of two reference frames. The first one is the inertia frame defined as $\mathcal{F}_I = \left\{O_I, \vec{i}_I, \vec{j}_I, \vec{k}_I\right\}$. The second is the body fixed reference frame defined as $\mathcal{F}_B = \{O_B, i_B, j_B, k_B\}$ where the center O_B is located at the Center of Gravity (CoG) of the helicopter. The direction of of the body fixed frame unitary vectors can be seen in Figure 4.1, although the axis vectors $\vec{j}_B, \vec{k}_B, \vec{j}_I, \vec{k}_I$ are reversed from typical placement [41, 47, 32] as shown. This is to comply with the space-fixed frame desired for tracking calculations (Section 3.2).

The angular velocity with respect to the body frame is $\omega^B = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^T \in \mathbb{R}^3$. Positive direction of the angular velocity components refers to the right-hand rule of the respective axis.

The external forces and moments acting on the CoG of the helicopter are denoted by f^B and τ^B, expressed in the body frame coordinates. The rotation matrix R is parametrized with respect to the three Euler angles roll (ϕ), pitch (θ), and yaw (ψ) and maps vectors from the the body

fixed frame $\mathcal{F}_B$ to the inertia frame $\mathcal{F}_I$. The same standard rotation matrices as in Eq. (3.9) and multiplied over all three rotations results in

$$R = \begin{bmatrix} C_\psi C_\theta & -S_\psi C_\phi + C_\psi S_\theta S_\phi & S_\phi S_\psi + C_\phi S_\theta C_\psi \\ S_\psi C_\theta & C_\phi C_\psi + S_\phi S_\theta S_\psi & -C_\psi S_\phi + S_\psi S_\theta C_\phi \\ -S_\theta & C_\theta S_\phi & C_\theta C_\phi \end{bmatrix}. \tag{4.1}$$

The orientation vector is given by $\Theta = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ and the associated orientation dynamics are governed by $\dot{\Theta} = \Psi(\Theta)\omega^B$, where

$$\Psi(\Theta) = \begin{bmatrix} 1 & S_\phi T_\theta & C_\phi T_\theta \\ 0 & C_\phi & -S_\phi \\ 0 & S_\phi / C_\theta & C_\phi / C_\theta \end{bmatrix}. \tag{4.2}$$

Let $p^I = \begin{bmatrix} p_x^I & p_y^I & p_z^I \end{bmatrix}^T \in \mathbb{R}^3$ be the position vector of the CoG of the helicopter with respect to the inertial coordinates, and $v^I = \begin{bmatrix} v_x^I & v_y^I & v_z^I \end{bmatrix}^T \in \mathbb{R}^3$ be the linear velocity vector in inertial coordinates. The complete dynamic equations of the rigid body in the special orthogonal configuration space, $SE(3) = \mathbb{R}^3 \times SO(3)$, are

$$\dot{p}^I = v^I, \tag{4.3}$$

$$\dot{v}^I = \frac{1}{m} R f^B, \tag{4.4}$$

$$\dot{R} = R\hat{\omega}^B, \tag{4.5}$$

$$\mathcal{I}\dot{\omega}^B = \omega^B \times \left(\mathcal{I}\omega^B\right) + \tau^B, \tag{4.6}$$

where $\mathcal{I}$ denotes the inertia matrix of the helicopter with respect to the body fixed reference frame.

4.2.3 FORCE AND MOMENT MODELS

There are four control commands associated with helicopter piloting. The control input is defined as $u = \begin{bmatrix} T_M & T_T & a & b \end{bmatrix}^T$ where T_M and T_T are the thrust magnitudes generated by the main and tail rotor, respectively. The other two control commands are the flapping angles a, b, and they represent the tilt of the Tip-Path-Plane (TPP) at the longitudinal and lateral axis, respectively.

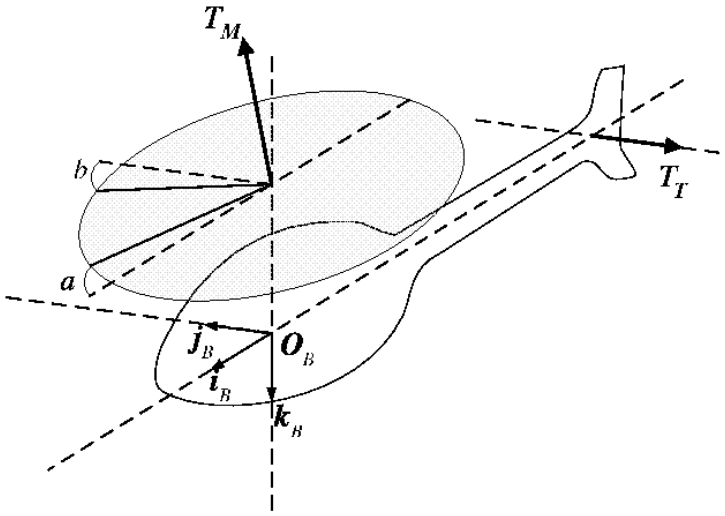


Figure 4.1: *The helicopter body frame, the TPP angles, and the thrust vectors of the main and tail rotor [47].*

The TPP is the plane in which the tips of the blades lie. Since the thrust vector is considered normal to the TPP, by controlling the TPP inclination the pilot indirectly controls the direction of the propulsion forces. The inclination of the TPP can be seen in Figure 4.1.

Let $\vec{h}_M = \begin{bmatrix} x_m & y_m & z_m \end{bmatrix}^T$ and $\vec{h}_T = \begin{bmatrix} x_t & y_t & z_t \end{bmatrix}^T$ be the position vectors of the main and tail rotors shafts respectively (expressed in the body coordinate frame). Since the thrust vector is assumed to be perpendicular to the TPP from simple geometry, the torque equations follow

$$\vec{T}_M = \begin{bmatrix} X_M \\ Y_M \\ Z_M \end{bmatrix} = \begin{bmatrix} S_a C_b \\ -S_a C_b \\ C_a C_b \end{bmatrix} T_M \approx \begin{bmatrix} a \\ -b \\ 1 \end{bmatrix} T_M. \tag{4.7}$$

The above equation (4.7) is simplified by assuming small angle approximation ($\cos(\cdot) = 1$ and $\sin(\cdot) = (\cdot)$) for the flapping angles. The small angle assumption is adopted by [18, 23, 32]. For the tail rotor,

$$\vec{T}_T = \begin{bmatrix} 0 & Y_T & 0 \end{bmatrix}^T = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T T_T. \tag{4.8}$$

Therefore by including the helicopter weight, with mass m and standard acceleration due to

gravity $g = -9.82\,\mathrm{m/s^2}$, the complete force vector is

$$f^B = \begin{bmatrix} X_M \\ Y_M + Y_T \\ Z_M \end{bmatrix} + R^T \begin{bmatrix} 0 \\ 0 \\ mg \end{bmatrix}. \tag{4.9}$$

A common simplification practice followed in [23, 17, 30] is to neglect the effect of the lateral and longitudinal forces produced by the TPP tilt and the effect of the tail rotor thrust. Therefore,

$$f^B = \begin{bmatrix} 0 & 0 & T_M \end{bmatrix}^T + R^T \begin{bmatrix} 0 & 0 & mg \end{bmatrix}^T. \tag{4.10}$$

The generated torques are the result of the above forces and the moments of the rotors. Let $\tau_M^B = \vec{h}_M \times \vec{T}_M$ and $\tau_T^B = \vec{h}_T \times \vec{T}_T$ be the torques generated by $\vec{T}_M$ and $\vec{T}_T$, respectively. The complete torque vector is

$$\begin{aligned} \tau^B &= \tau_\beta^B + \tau_Q^B + \tau_M^B + \tau_M^B \\ &= \begin{bmatrix} -K_\beta b \\ -K_\beta a \\ 0 \end{bmatrix} + \begin{bmatrix} S_a C_b \\ -S_a C_b \\ C_a C_b \end{bmatrix} Q_M + \vec{h}_M \times \vec{T}_M + \vec{h}_T \times \vec{T}_T, \end{aligned} \tag{4.11}$$

where τ_β^B is a torque on the helicopter body due to a spring-like force from the main rotor hub stiffness and K_β is the stiffness of the main rotor blades; τ_Q^B is the anti-torque due to spinning of the main rotor, with $Q_M = C_M \left|T_M\right|^{1.5} + D_M$ and C_M, D_M are positive constants. A more compact form of the torque can be given as

$$\tau^B = A\left(T_M\right) \begin{bmatrix} a \\ b \\ T_T \end{bmatrix} + B\left(T_M\right), \tag{4.12}$$

where

$$A(T_M) = \begin{bmatrix} Q_M & z_m T_M - K_\beta & -z_t \\ z_m T_M - K_\beta & -Q_M & 0 \\ -y_m T_M & -x_m T_M & x_t \end{bmatrix}, \tag{4.13}$$

$$B(T_M) = \begin{bmatrix} y_m T_M \\ -\ x_m T_M \\ Q_M \end{bmatrix}. \tag{4.14}$$

4.2.4 COMPLETE RIGID BODY DYNAMICS

Using the force simplification assumption given in Eq. (4.10) and the applied torque by Eq. (4.12) the helicopter dynamics in the configuration space will have the form

$$\dot{p}^I = v^I, \tag{4.15}$$

$$\dot{v}^I = \frac{1}{m} R e_3 T_M - e_3 g, \tag{4.16}$$

$$\dot{R} = R\hat{\omega}^B, \tag{4.17}$$

$$\mathcal{I}\dot{\omega}^B = \omega^B \times \left(\mathcal{I}\omega^B\right) + A(T_M)\, v_c + B(T_M), \tag{4.18}$$

where $e_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$ and $v_c = \begin{bmatrix} a & b & T_T \end{bmatrix}^T$.

4.2.5 DISCRETE SYSTEM DYNAMICS

In this section, the discrete dynamics of the helicopter are provided by using Euler's implicit method for the approximation of the continuous derivatives. The motivation to shift the controller design to discrete time is twofold. First, control algorithms are computed by microprocessors so the discretization effect should be incorporated into the controller. And second, the ultimate purpose of the helicopter controller is to autonomously maintain a desired pose relative to a moving target, and data on the position of this target is being generated at discrete points in

time. The following set of equations are the discrete system to be controlled:

$$p^I_{k+1} = p^I_k + \Delta T v^I_k, \tag{4.19}$$

$$v^I_{k+1} = v^I_k + \alpha_1 R_k e_3 T_{M,k} + \alpha_2 e_3, \tag{4.20}$$

$$R_{k+1} = R_k + \Delta T R_k \hat{\omega}^B_k, \tag{4.21}$$

$$\omega^B_{k+1} = \omega^B_k + \Pi\left(\omega^B_k\right) + A\left(T_{M,k}\right) v_{c,k} + \bar{B}\left(T_{M,k}\right), \tag{4.22}$$

with the compact terms defined as

$$\alpha_1 = \Delta T \frac{1}{m} \qquad \alpha_2 = \Delta T g$$

$$\bar{A}(T_{M,k}) = \Delta T \mathcal{I}^{-1} A\left(T_{M,k}\right) \qquad \bar{B}\left(T_{M,k}\right) = \Delta T \mathcal{I}^{-1} B\left(T_{M,k}\right)$$

$$\Pi\left(\omega^B_k\right) = \Delta T \mathcal{I}^{-1}\left(\hat{\omega}^B_k \mathcal{I} \omega^B_k\right).$$

4.3 CONTROLLER DESIGN

The controller design is based on a previously derived discrete time backstepping controller [41, 42, 32], with only minor changes to accommodate a change in the coordinate system orientation. The rotorcraft dynamics and controller are not integral parts of the target tracking and geo-location problem, but are used to simulate realistic UAV dynamics and uncertainties.

Consider a helicopter described by the difference equations Eqs. (4.19)-(4.22). The objective is to design a nonlinear controller stabilizing the position p^I_k and the yaw angle ψ_k to the reference values $p^I_{r,k} = \begin{bmatrix} p^r_{x,k} & p^r_{y,k} & p^r_{z,k} \end{bmatrix}^T$ and $\psi_{r,k}$, respectively. The discrete time backstepping design procedure is similar with its continuous time counterpart. The main difference is that differentiation is substituted by shifting one time step forward in time.

4.3.1 ANGULAR VELOCITY DYNAMICS

Considering Eq. (4.22), an obvious control choice for canceling out the nonlinear terms of the angular velocity dynamics is

$$v_{c,k} = \bar{A}\left(T_{M,k}\right)^{-1}\left(-\omega^B_k - \Pi\left(\omega^B_k\right) - \bar{B}\left(T_{M,k}\right) + \bar{v}_k\right), \tag{4.23}$$

where $\bar{v}_k = \begin{bmatrix} v_{1,k} & v_{2,k} & v_{3,k} \end{bmatrix}^T$ is a stabilizing term designed in 4.3.2 and 4.3.3. The angular dynamics become

$$\omega^B_{k+1} = \bar{v}_k. \tag{4.24}$$

4.3.2 TRANSLATIONAL DYNAMICS

The equation of translational velocity is given by Eq. (4.20). Let $R_k = \begin{bmatrix} \rho_{1,k} & \rho_{2,k} & \rho_{3,k} \end{bmatrix}$, where $\rho_{i,k}$ for $i = 1, 2, 3$ are the column vectors of the rotation matrix. The difference equation of the translational velocity can be written as

$$v^I_{k+1} = v^I_k + \alpha_1 \rho_{3,k} T_{M,k} + \alpha_2 e_3. \tag{4.25}$$

The column vector $\rho_{3,k}$ is a unit vector with changing direction depending on the Euler angles. The goal is to change the direction of $\rho_{3,k}$ and at the same time adjust the magnitude of $T_{M,k}$ to match a desired vector that will control the translational velocity dynamics. Therefore, the vector function $\rho_{3,k} T_{M,k}$ is stepped forward in time until the components of $\bar{v}_k$ appear. Let $T_{M,k+1} = \eta_k$, and by considering Eq. (4.21) and that $\hat{\omega}^B_k e_3 = -\hat{e}_3 \omega^B_k$ then

$$\begin{aligned} \rho_{3,k+1} T_{M,k+1} &= R_{k+1} e_3 \eta_k \\ &= R_k e_3 \eta_k - \Delta T R_k \hat{e}_3 \omega^B_k \eta_k \\ &= R_k \left(e_3 - \Delta T \hat{e}_3 \omega^B_k \right) \eta_k. \end{aligned} \tag{4.26}$$

Let $\eta_{k+1} = \zeta_k$ then, by stepping forward in time once more, Eq. (4.26) becomes

$$\begin{aligned} \rho_{3,k+2} T_{M,k+2} &= R_{k+1} \left(e_3 - \Delta T \hat{e}_3 \omega^B_{k+1} \right) \eta_{k+1} \\ &= R_{k+1} \left(e_3 - \Delta T \hat{e}_3 \bar{v}_k \right) \zeta_k \\ &= R_{k+1} \begin{bmatrix} \Delta T v_{2,k} \\ -\Delta T v_{1,k} \\ 1 \end{bmatrix} \zeta_k = \mathcal{X}_k, \end{aligned} \tag{4.27}$$

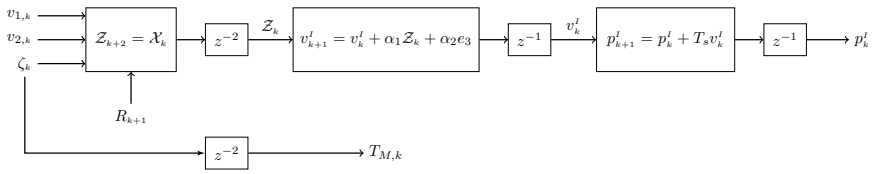


Figure 4.2: *Interconnection of the helicopter dynamics using Eqs. (4.28)-(4.32). The term z^{-1} denotes a unit time delay [47].*

where $\mathcal{X}_k$ is a vector as defined below. From Eq. (4.27), the following equalities hold:

$$\zeta_k = e_3^T R_{k+1}^T \mathcal{X}_k \tag{4.28}$$

$$\begin{bmatrix} v_{1,k} \\ v_{2,k} \end{bmatrix} = \begin{bmatrix} -\Delta T \zeta_k & 0 \\ 0 & \Delta T \zeta_k \end{bmatrix}^{-1} \begin{bmatrix} \rho_{2,k+1}^T \mathcal{X}_k \\ \rho_{1,k+1}^T \mathcal{X}_k \end{bmatrix} \tag{4.29}$$

The existence of the inverse matrix on the right hand side of Eq. (4.29) is guaranteed by the fact that the thrust magnitude $T_{M,k}$ should be always greater than zero (note that $\zeta_k = T_{M,k+2}$), since during typical flight conditions some thrust is needed to compensate for the weight force. Some rare maneuvers exist during which the desired thrust may be zero, requiring a small but non-zero ζ_ϵ such that $\zeta_k = T_{M,k+2} + \zeta_\epsilon$.

Let $\mathcal{Z}_{k+i} = \rho_{3,k+i} T_{M,k+i}$ with $i \in \mathbb{N}$. The associated equations related to the translational dynamics are

$$p^I_{k+1} = p^I_k + \Delta T v^I_k, \tag{4.30}$$

$$v^I_{k+1} = v^I_k + \alpha_1 \mathcal{Z}_k + \alpha_2 e_3, \tag{4.31}$$

$$\mathcal{Z}_{k+2} = \mathcal{X}_k. \tag{4.32}$$

Using the above equation, the interconnection of the helicopter dynamics is shown in Figure 4.2. The error dynamics of the p^I, v^I and $\mathcal{Z}$ state variables are

$$e_{p,k+1} = p^I_{k+1} - p^I_{r,k+1} = -p^I_{r,k+1} + p^I_k + \Delta T v^I_{d,k} + \Delta T e_{v,k}, \tag{4.33}$$

$$\begin{aligned} e_{v,k+1} &= v^I_{k+1} - v^I_{d,k+1} \\ &= -v^I_{d,k+1} + v^I_k + \alpha_1 \mathcal{Z}_{d,k} + \alpha_2 e_3 + \alpha_1 e_{\mathcal{Z},k}, \end{aligned} \tag{4.34}$$

$$e_{\mathcal{Z},k+2} = \mathcal{Z}_{k+2} - \mathcal{Z}_{d,k+2} = -\mathcal{Z}_{d,k+2} + \mathcal{X}_k. \tag{4.35}$$

Let the desired values be chosen as

$$v_{d,k}^I = \frac{1}{\Delta T}\left[p_{r,k+1}^I - p_k^I + K_1 e_{p,k}\right], \tag{4.36}$$

$$\mathcal{Z}_{d,k} = \frac{1}{\alpha_1}\left[v_{d,k+1}^I - v_k^I + K_2 e_{v,k} - \alpha_2 e_3\right], \tag{4.37}$$

$$\mathcal{X}_k = \mathcal{Z}_{d,k+2} + \Lambda_1 e_{\mathcal{Z},k+1} + \Lambda_2 e_{\mathcal{Z},k}, \tag{4.38}$$

where $K_1, K_2, \Lambda_1, \Lambda_2$ are diagonal gain matrices. After applying the desired values of Eqs. (4.36)-(4.38) to the translational dynamics described in Eqs. (4.33)-(4.35). the following is obtained:

$$\begin{bmatrix} e_{p,k+1} \\ e_{v,k+1} \\ e_{\mathcal{Z},k+2} \\ e_{\mathcal{Z},k+1} \end{bmatrix} = \begin{bmatrix} K_1 & \Delta T & 0 & 0 \\ 0 & K_2 & 0 & \alpha_1 \\ 0 & 0 & \Lambda_1 & \Lambda_2 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} e_{p,k} \\ e_{v,k} \\ e_{\mathcal{Z},k+1} \\ e_{\mathcal{Z},k} \end{bmatrix} \tag{4.39}$$

Provided that the eigenvalues of the above linear system lie inside the unit circle, the translational helicopter dynamics will be globally asymptotically stable. Therefore, the convergence rate of the error variables can be determined by the designer. Due to the fact that small-scale helicopters are very sensitive to control inputs, regulating the convergence rate of each state improves the flight behavior.

4.3.3 YAW DYNAMICS

The yaw dynamics are obtained from Eq. (4.2). More specifically,

$$\psi_{k+1} = \psi_k + \Delta T e_3^T \Psi\left(\Theta_k\right) \omega_k^B. \tag{4.40}$$

Let $e_{\psi,k} = \psi_k - \psi_{r,k}$ be the error in the yaw, then the yaw error dynamics are

$$e_{\psi,k+1} = \psi_{k+1} - \psi_{r,k+1} = -\psi_{r,k+1} + \psi_k + \Delta T e_3^T \Psi\left(\Theta_k\right) \omega_k^B. \tag{4.41}$$

The above equation is shifted forward in time in order for the control commands to appear. This

leads to

$$\begin{aligned} e_{\psi,k+2} &= -\psi_{r,k+2} + \psi_{k+1} + \Delta T e_3^T \Psi\left(\Theta_{k+1}\right) \omega_{k+1}^B \\ &= -\psi_{r,k+2} + \psi_{k+1} + \Delta T \left(\frac{S_{\phi_{k+1}}}{C_{\theta_{k+1}}} v_{2,k} + \frac{C_{\phi_{k+1}}}{C_{\theta_{k+1}}} v_{3,k} \right). \end{aligned} \tag{4.42}$$

An obvious choice for the selection of the value of $v_{3,k}$, that will cancel out the nonlinear terms and stabilize the yaw error dynamics, is

$$v_{3,k} = \frac{C_{\phi_{k+1}}}{C_{\theta_{k+1}}} \left[-\frac{S_{\phi_{k+1}}}{C_{\theta_{k+1}}} v_{2,k} + \frac{1}{\Delta T} \left(\psi_{r,k+2} - \psi_{k+1} + M e_{\psi,k+1} \right) \right], \tag{4.43}$$

where M is a diagonal matrix of gains with the absolute value of each diagonal entry being smaller than unity. Applying the above value for $v_{3,k}$, the yaw error dynamics become $e_{\psi,k+2} = M e_{\psi,k+1}$, which implies the asymptotic convergence of $e_{\psi,k}$ to zero.

4.4 CAMERA CONTROL AND TRAJECTORY GENERATION

4.4.1 GIMBAL CAMERA CONTROL

Gimbal devices are a common choice for camera positioning, allowing aiming of a directional sensor [33, 6]. Gimbal orientation may be set through two command types: position commands or angular velocity commands. For simplicity, a gimbal that accepts position commands only is chosen; design and use of an angular velocity controller is beyond the scope of this research. The gimbal aims the camera at the measured or estimated location of the target, attempting to place the target in the center of the camera image. A precise estimate is not necessary; only the estimate error bounds, projected onto the camera image frame, must lie within the field-of-view (FoV) of the camera.

The camera gimbal is assumed to orient an attached camera sensor in the desired direction, with the observation vector towards the target, by accepting a pitch and yaw command. Let $\bar{\eta} = \begin{bmatrix} \bar{\eta}_x & \bar{\eta}_y & \bar{\eta}_z \end{bmatrix}^T$ denote the unitary directional vector from the helicopter CoG to the target, expressed in the body-fixed frame. The position of the target in the inertial frame is denoted by

the vector $p_t^I = \begin{bmatrix} p_x^t & p_y^t & p_z^t \end{bmatrix}^T$. The desired pitch and yaw commands for the camera are

$$\bar{\eta} = R^T \frac{p_{t,k}^I - p_k^I}{\left\| p_{t,k}^I - p_k^I \right\|}, \tag{4.44}$$

$$\psi_{cam} = atan2\left(\bar{\eta}_y, \bar{\eta}_x\right), \tag{4.45}$$

$$\theta_{cam} = atan2\Big(\bar{\eta}_z, cos\left(\psi_{cam}\right)\Big), \tag{4.46}$$

where $atan2\left(\cdot,\cdot\right)$ is the four-quadrant $arctan\left(\cdot\right)$ function.

4.4.2 TRAJECTORY GENERATION

Trajectory generation is a broad problem with many solutions available, each dependent on the application goals [40, 26, 19]. For this application, the rotorcraft UAV is intended to be following a ground target with an estimated location in the inertial frame, or in Earth-fixed coordinates. The UAV will need to both seek the target to retain proximity and orbit the target for multiple observation vantages as the situation changes.

According to this scenario, the target is assumed to be traveling at an altitude below the UAV, with a variable speed and variable heading. There exists on-board equipment for measuring and estimating the position and velocity of the target with a degree of accuracy, particularly the camera sensors and geo-location particle filter. It is therefore assumed that the position, p_t^I, and velocity, v_t^I, of the target are estimated with finite error bounds.

The UAV is tasked with keeping the target within range of its sensing equipment. Depending on the type of sensors on-board, this operative range may vary, and will depend on criteria such as sensor coverage footprint, resolution, and error, as well as uncertainties in the position and orientation of the UAV. The range is thus described as a hemisphere above the terrain with a radius, r, determined by said criteria.

During all operation, the UAV is restricted to a single altitude, denoted by h, to ensure airspace deconfliction with multiple UAVs and to simplify the operative range to a circle centered over the location of the target at the altitude of the UAV, with radius $\mathcal{R} = \sqrt{r^2 - (h + p_z^T)^2}$. Therefore, the desired position of the UAV will always be the nearest point of this circle.

The vector function $\mathcal{P}_{xy} : \mathbb{R}^3 \to \mathbb{R}^2$ is defined such that for every $\lambda = \begin{bmatrix} \lambda_x & \lambda_y & \lambda_z \end{bmatrix}^T \in \mathbb{R}^3$, $\mathcal{P}_{xy}(\lambda) = \begin{bmatrix} \lambda_x & \lambda_y \end{bmatrix}^T$. The directional vector of the target with respect to the helicopter in the $x-y$ plane of the inertia frame is denoted by $\pi = \begin{bmatrix} \pi_x & \pi_y \end{bmatrix}^T$. Because the velocity of the target is either estimated or known, the desired yaw angle is calculated as the heading to the target:

$$\pi_k = \mathcal{P}_{xy}\left(p^I_{t,k} - p^I_k\right), \tag{4.47}$$

$$\psi_{r,k} = atan2\left(\pi_{y,k}, \pi_{x,k}\right). \tag{4.48}$$

Adding the altitude restriction, the reference position vector is given by $p^I_{r,k} = \begin{bmatrix} \pi_r & h \end{bmatrix}^T$, where

$$\pi_{r,k} = \mathcal{P}_{xy}\left(p^I_{t,k}\right) - \frac{\pi_k}{\|\pi_k\|}\zeta\mathcal{R}, \tag{4.49}$$

and $\zeta \in \mathbb{R}, 0 \leq \zeta < 1$ is a scaling factor to ensure the reference position is within the target circle. The ζ factor is chosen to maximize the orbit radius while keeping the resulting circle inscribed within a region described by the absolute target position estimate error bound, ϵ, so that $(1-\zeta)\mathcal{R} \leq \epsilon$.

There are benefits to keeping a UAV in motion while it is tracking a target, either by flying in a specific pattern relative to a moving target, or by orbiting a stationary target. The general goal is to minimize the geometric dilution of precision (GDOP) between any UAVs in the scenario and to maximize the line-of-site rate (LOS-rate) for each UAV.

The measurement uncertainty of a single observer covers a volume of the state space defined by the measurement error bounds and the geometry of the observation function. For the case of a video camera mounted on a UAV, this volume is a cone extending from the camera and encompassing the actual location of the target. For multiple volumes defined by multiple observers, the overlap of these volumes contains the majority of the probability density of the target location and represents the target's most likely location, and the GDOP is a measure of the overlapping volume between observations. Therefore, the smaller the overlap between volumes, the smaller the region in which the target may be found and the smaller the GDOP. For the problem of geo-location using video sensors and the case of using two UAVs, it is ideal to separate the UAVs by 90° around the target in any direction to minimize the GDOP. For more than two UAVs, other geometric configurations

may be used, such as spreading UAVs out evenly in a circle around the target, or placed around a sphere surrounding the target.

For individual UAVs, the LOS-rate is maximized to continuously produce succesively new measurements of the target. A stationary UAV is only viewing a target from a single angle at all times, whereas an orbiting UAV is viewing the target from new angles over time. Ideally, the LOS-rate should be maximized at all times, however, it is desired for the UAVs to close the distance to the target as fast as possible to first minimize estimation errors due to the range ot the target (as seen in Chapter 6). After reaching the target, the UAV then performs an orbiting maneuver to maximize the LOS-rate. It is desired that once the UAV enters this circle, it begins to orbit around the circumference of the circle until the target and the circumscribed circle above it move out of range. The reference values when the UAV is in orbital mode are

$$\dot{\bar{\theta}} = 2sin^{-1}\left(\frac{d}{2\zeta\mathcal{R}}\right), \tag{4.50}$$

$$\pi_{r,k} = \zeta\mathcal{R}\begin{bmatrix} cos\left(\dot{\bar{\theta}}T_s + \bar{\theta}\right) \\ sin\left(\dot{\bar{\theta}}T_s + \bar{\theta}\right) \end{bmatrix} + \mathcal{P}_{xy}\left(p^I_{t,k}\right), \tag{4.51}$$

$$\psi_{r,k} = atan2\left(p^r_{y,k} - p^I_{y,k}, p^r_{x,k} - p^I_{x,k}\right), \tag{4.52}$$

$$\bar{\theta} = atan2\left(\left(p^I_{y,k} - p^t_{y,k}\right), \left(p^I_{x,k} - p^t_{x,k}\right)\right). \tag{4.53}$$

Although the controller is designed to handle UAV overturning gracefully, it is practically undesirable to allow this behavior. A maximum allowable acceleration, a_m, is set to prevent large values for tilt angle contorls a and b. This desired acceleration translates to subsequent reference positions

$$p^I_{r,k+i} = p^I_k + i\Delta T v^I_k + \frac{1}{2}a_m\left(i\Delta T\right)^2. \tag{4.54}$$

Additional heuristic commands are used to account for three possible cases: 1) the UAV is currently travelling away from $p^I_{r,k}$ ($-e_{p,k}\cdot v^I_k < 0$), 2) the UAV is within deceleration range ($\|e_{p,k}\| < r_d$), and 3) the UAV is beyond deceleration range ($\|e_{p,k}\| \geq r_d$). The deceleration range,$r_d = \frac{\left\|v^I_k\right\|^2}{2a_m}$, is the distance required for the UAV to reach a stop, $\left\|v^I\right\| = 0$, from current velocity at fixed deceleration rate $a = -a_m$. For each of the three cases, the acceleration is contextually chosen as either $a = a_m$ or $a = -a_m$ to accelerate the UAV towards the reference position $p^I_{r,k}$ or decelerate the UAV to a stop.

Table 4.1: *UAV system parameters.*

C_M	$0.004452\ {}^{\mathrm{m}}/\sqrt{\mathrm{N}}$	h_M	$\begin{bmatrix} -0.015 & 0 & 0.235 \end{bmatrix}^T$ m
D_M	$0.6304\ \mathrm{N \cdot m}$	h_T	$\begin{bmatrix} -0.91 & 0 & 0.08 \end{bmatrix}^T$ m
m	$8.2\ \mathrm{kg}$	K_β	$52\ {}^{\mathrm{N \cdot m}}/_{\mathrm{rad}}$
$\mathcal{I}$	$\mathrm{diag}(0.18, 0.34, 0.28)\ \mathrm{kg \cdot m^2}$		

Table 4.2: *UAV controller gains.*

K_1	$diag\,(0.99, 0.99, 0.98)$
K_2	$diag\,(0.97, 0.97, 0.90)$
Λ_1	$diag\,(0, 0, 0)$
Λ_2	$diag\,(0.9, 0.9, 0.85)$

4.5 SIMULATION RESULTS

The simulation uses rotorcraft model parameters similar to those of a small-scale model helicopter developed at U. C. Berkeley and found in Table 4.1. For simplicity, all control signal gains are set to 1 and the UAV starts at the origin with a yaw angle of 0^o. The control gains are shown in Table 4.2. The UAV is restricted to an altitude of 80 meters, with $\zeta = 0.5$ and $d = 90$ meters chosen through trial and error.

Some care must be taken in selecting controller gains. To reduce the likelihood of the UAV overcorrecting for large position errors, the gains are set close to positive one, resulting in more conservative control overall. Due to the fact that the control for the position is two sample periods behind control over the orientation, the thrust control, which is primarily responsible for rectifying position errors, is set to be stronger than the tilt control by setting z axis gains to smaller values.

The simulated target is assumed to be a ground vehicle that starts 100 meters east and 150 meters south of the UAV. The target trajectory is as follows: for 200 seconds it travels northeast at 40 kph, for another 200 seconds it travels southeast at 30 kph, for 100 seconds it remains stationary, for 50 seconds it travels west at 54 kph, then it stops for the remainder of the simulation. The ground is a flat plane at $z = 0$, so the target travels over level terrain for the duration of the simulation and the UAV never conflicts with the ground.

Figure 4.3 shows the resulting UAV position in the $x-y$ directions, along with the target position, at various points in time during the simulation. Figure 4.4 shows the UAV altitude, set to be at the desired altitude of 80 meters.

It is shown that the rotorcraft UAV can be controlled to maintain a desired altitude while tracking a moving target on the ground. While the target is in motion, the UAV follows the target closely

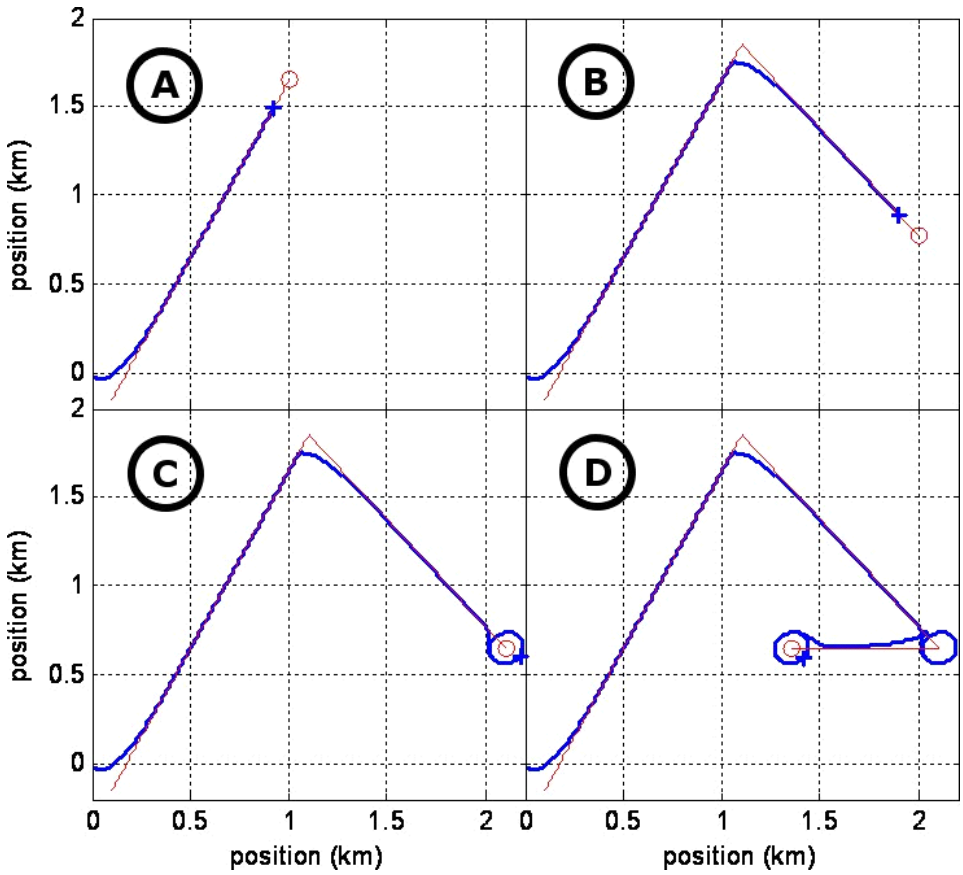


Figure 4.3: *Position of the target (thin, red line) and the UAV (thick, blue line) at times 180 seconds (A), 380 seconds (B), 480 seconds (C), and 800 seconds (D).*

behind, attempting to remain within sensor range. Once the target has come to a stop, the UAV switches modes and begins to orbit the target, as can be seen in Figure 4.3C and 4.3D. The UAV controller is shown to keep the UAV in a stable condition throughout the simulation.

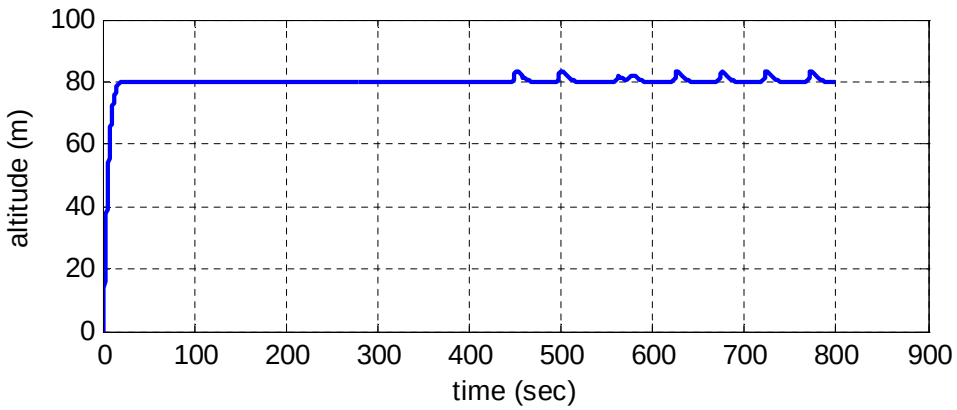


Figure 4.4: *Altitude of the UAV during simulation. Desired altitude is 80 meters.*

CHAPTER V

SIMULATION AND PERFORMANCE METRICS

5.1 SIMULATION

The testing procedure is split into two parts: simulation and performance metrics extraction. The simulation is restricted to its own environment to ensure clean measurements of processing time and real-time visualization. Raw data is extracted and saved during simulation, and then imported into MATLAB for processing and performance metrics extraction. MATLAB provides tools for fast vector data processing and plotting.

The simulation is written entirely in the C++ programming language. The C++ language was chosen for its flexibility in all areas of the design process. The object-oriented design philosophy of the language allows easier creation of modular systems, *i.e.* separate UAVs, UAV subsystems, and target system. Public code libraries are widely available for complex data structures, system APIs, and computational algorithms relating to the geo-location problem. The 3D environment is processed using the OpenGL API, which communicates two-dimensional and three-dimensional model and drawing information to the graphics processing unit (GPU) of the computer. Because the geo-location problems naturally involves three-dimensional constructs and transforms, taking advantage of the processing capabilities of the GPU is a step towards optimizing the geo-location algorithm.

The simulation consists of one target, supported by a behavioral model expressed in Chapter 3 and specifically Section 3.6. One or more UAVs may be added or removed from the simulation at any time, with UAV consisting of a dynamics model, a controller and trajectory generator, and a camera sensor, as defined in Chapters 3 and 4. The geo-location particle filter accepts measurement data from the UAVs, processes these data according to the geometric transformations and information about the UAVs, and estimates the target location in the world coordinate system. Sensor noise and uncertainty is modeled during the simulation as well.

The simulation updates at a maximum rate of 40 Hz. In cases where the processing requirements of the system do not allow an update rate of 40 Hz, such as during high system load, the system may update at a slower rate. All systems are designed to update at a variable rate during simulation to allow for this case. Some sensitive systems are negatively impacted should this occur, but this is restricted to the UAV controller and trajectory generator. During the testing cases related in

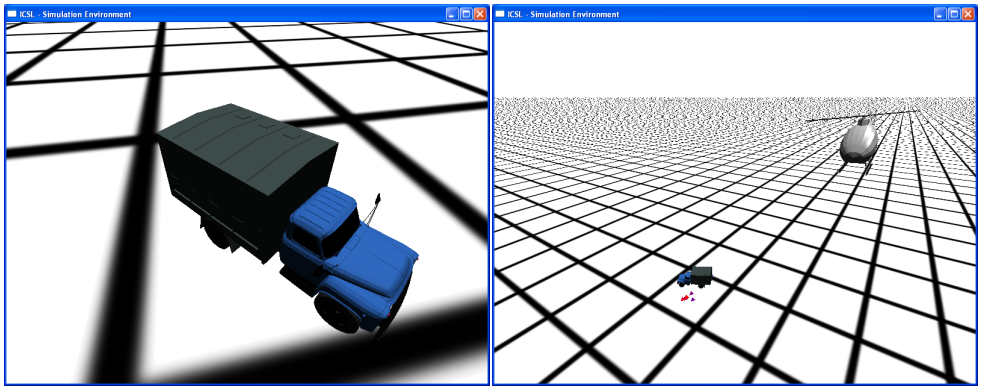


Figure 5.1: *Example screenshots of the simulation environment.*

Chapter 6, the visualization module is deactivated to provide maximum processing power to the target, UAV, and particle filtering modules.

Some example screenshots of the visualization environment are shown in Figure 5.1. The black grid is for distance reference and is oriented along the $x - y$ plane at $z = 0$. The grid does not take any part in helping to estimate the location of the vehicle in the image frame or in the environment, it is for visualization only. The target is represented by a model of a truck, and the UAV is represented by a model of a helicopter. Both models are designed to scale, but increased by a factor of ten for visualization purposes. In the screenshot on the right in Figure 5.1, several colored objects are visible near the target. Each is a triangle located in the world-space coordinates, and represents one particle in the geo-location particle filter.

5.2 PERFORMANCE METRICS

Test runs are designed to examine different aspects of the geo-location particle filter and UAV control. Each test run is performed a number of times, usually 50. Performance metrics are extracted after the completion of a full set of test runs to allow for a wider range of test configurations and random behavioral patterns. The following metrics are extracted.

5.2.1 TARGET LOCATION ESTIMATE

The target position is the desired output of the geo-location particle filter, although the full particle state may be estimated, including the velocity, acceleration, and angular velocity when appropriate. Because the particle state is weighted according to the likelihood of the particle position matching the sensor measurements, the target position is most likely to be correct. Other

target dynamics may be derived from the estimated target position instead of extracted from the particle state if desired.

The target position estimate, $\bar{p}$, is calculated using a weighted mean of the particle state. The weighted sum is the first moment expectation of the PDF formed by the particles (Eq. (2.17)). The location estimate is used to evaluate the particle filter error or accuracy.

5.2.2 ESTIMATOR ACCURACY

The estimator accuracy is determined through analysis of the absolute error of the target location estimate. The estimator accuracy is inversely proportional to the absolute estimator error, and is an abstract term meant to convey how close the estimator may be to the actual target location. A reliable sense of the accuracy of the estimator may be obtained through statistical analysis of the absolute and relative error measures.

The formula for the absolute error, ϵ, is $\epsilon = \|p^\tau - \bar{p}^\tau\|_2$, where $\bar{p}^\tau$ is the first triplet of the estimated particle state, or the estimated position of the target. To better represent the geo-location accuracy over numerous runs, the error may calculated at all times for each run and averaged at each time instant over all runs. The overall error may be averaged across all time and all runs.

5.2.3 ESTIMATION PERCENT ERROR

The distance of the UAVs to the target is shown in Chapter 6 to play a role in the accuracy of the target location estimation accuracy. The estimation percent error metric accounts for the distance to the target to better show the behavior of the geo-location particle filter at any distance.

This is a relative metric, calculated by dividing the absolute estimation error by the average distance from each UAV to the target, or as $\eta = \frac{\epsilon}{\text{mean}(\|p^\tau - p^U\|)}$. The relative metric may be more easily recognizable as a percent, or when $\eta^{\%} = \eta * 100$. Like the previous metrics, the percent error is averaged at each time instant over all runs.

5.2.4 ERROR VS. DISTANCE

This exmaines the same dependence as the percent error metric, plotting ϵ versus mean $\left(\left\|p^\tau - p^U\right\|\right)$. However, this method of examining the data emphasizes the role of the distance between the UAVs and the target in determining the geo-location error.

5.2.5 CPU PROCESSING TIME

The amount of time spent processing a specific section of the geo-location particle filter algorithm is called the CPU time. This is calculated by subtracting the time, t_s, when the geo-location process enters the section of code from the time, t_e, when the process leaves the section of code.

To acheive high precision time measurements on the Windows machine the internal Windows process timer is used, which has a precision on the order of microseconds and is accessed through the QueryPerformanceCounter function. This function returns the time, t', in the number of processing ticks since process initiation. To translate this to seconds, this is divided by the number of ticks per second, f, as reported by the QueryPerformanceFrequency function. Therefore, measuring the amount of time spent processing a section of the geo-location algorithm is done by

$$t = \frac{t'_e - t'_s}{f}$$

CHAPTER VI

TESTS AND RESULTS

6.1 BASE TEST CASE

The initial tests serve as a baseline case from which to analyze the behavior of the particle filter itself. The random target maneuvers and UAV control have been removed to reduce testing variety. This test shows some basic issues with the particle filter geo-location algorithm. The accuracy and error metrics described in Section 5.2 are the most useful methods of visualizing the performance of this initial test.

Only one mode is activated in the geo-location particle filter, matching the single behavior of the target: constant velocity motion. Although the target behavior is known, the target position and velocity is unknown to the UAVs at the start of the test. The use of a single target mode is intended to restrict the convergence time of the particle filter to acquisition of the target position only, ignoring the time required to determine the correct target mode. Given the initial placement of the particles is relatively near ot the target location, convergence time only appears as a factor during test runs where the particle filter loses track of the target entirely. For this run, the particle count is fixed at 50 particles.

The target is placed at $p^{\tau}(0) = \begin{bmatrix} -30m & 10m & 0m \end{bmatrix}^T$ at the start of the test run, with velocity $v^{\tau} = \begin{bmatrix} 1.15^m/_s & 0.96^m/_s & 0^m/_s \end{bmatrix}^T$, or a speed of $1.5^m/_s$ and a heading of 40° north of east. Each run lasts for 120 seconds (at 40 computation cycles per second), resulting in a final target position of about $p^{\tau}(120) = \begin{bmatrix} 110 & 125 & 0 \end{bmatrix}^T$.

The target is geo-located by three stationary UAV observers. The UAVs are placed at random positions $p^U = \begin{bmatrix} u_1 & u_2 & 40 \end{bmatrix}^T$, where $u_{1,2} \sim U(-100, 100)$. There is a high probability that the UAVs will surround the target over part of its path, and a low probability that the UAVs will never surround the target, for instance being placed all to one side of the target path or, in the worst case, behind the target entirely. The particle filter uses 50 particles for each run. The test is repeated 50 times, and the average performance of the geo-location particle filter accounts for cases of good and poor UAV placement relative to the target and its path.

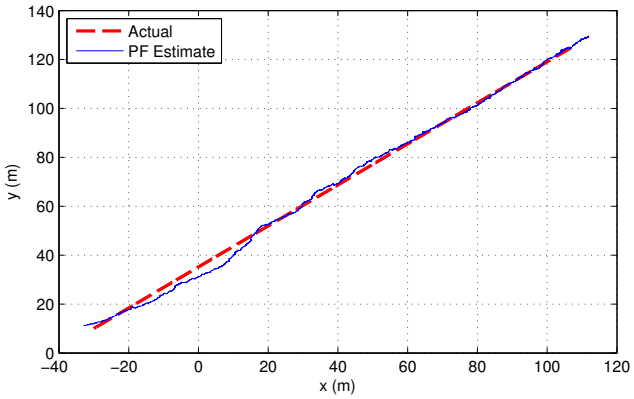


Figure 6.1: *Actual and estimated target paths.*

Target Location Estimate — The target location estimate produced by the geo-location particle filter over the 120 second test run forms a time-dependent track that is desired to follow the target as precisely as possible. The tracks from all 50 runs are averaged together and shown in Figure 6.1. This average track, although not representative of a single geo-location run, shows the general capabilities of the geo-location particle filter to track the target with a minimum set of assumptions, the one assumption being that the target moves with a constant but unknown velocity.

In Figure 6.1, the dashed line represents the steady path that the target takes during the test run, and the solid line is the average estimated track. The target begins in the bottom left of the map, and ends at the top right at the end of the run. The three UAVs are placed at random positions for each run as described above, which are not shown here. Also not shown is the z dimension, or the altitude, as neither the target nor the UAVs change altitude during the test run. A more in-depth examination of the geo-locator performance is given farther below.

Besides giving a sense of how the particle filter estimator tracks the target, there are a couple of points of interest in Figure 6.1. The first point, though difficult to see, is the extension of the estimated track beyond the actual path at the end of the test run. It is desired that the estimated track is as close to the actual track as possible, minimizing estimator error. However, the estimated track begins to deviate further from the actual path in the opposite direction from the location of the UAVs (restricted to x and $y \in [-100, 100]$). This is a limitation of the range-limited geo-location system. The range of a target that is moving directly away from an observer becomes diffcult to impossible to estimate. The target is estimated as farther away from the UAVs (Figure 6.2a) than nearer because the particle filter favors higher speeds on the target

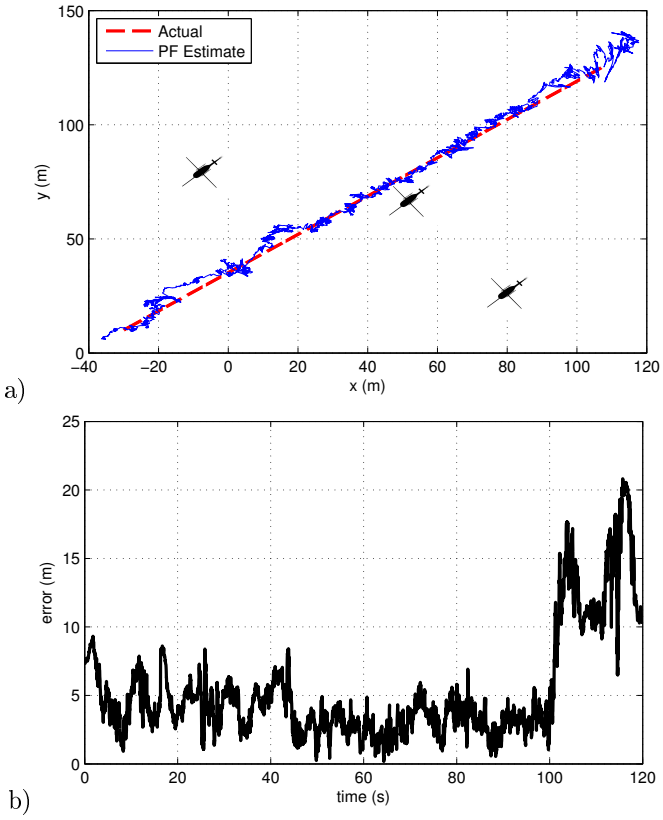


Figure 6.2: *Example of target geo-location with good UAV placement.*

motion vector.

The second point of interest is the deviation apparent towards the beginning of the test runs. Analysis of the 50 test runs shows that this is due to the particle filter losing track of the target for one or more runs because of poor UAV placement. After averaging, the exceptional error during these iterations produce a larger error in the average track.

For an example of good geo-location tracking resulting from good UAV placement, see Figure 6.2. For this 13th iteration of the base test, the UAVs are placed to either side of the target path and are relatively spaced apart within the space (Figure 6.2a). This provides three distinct vantages for observing the target during the test, including two at nearly 180° for a large portion of the target path. Notably, the geo-location error (Figure 6.2b) decreases as the target moves between the UAVs. It is during this time, from 45 to 100 seconds into the test run, that the observers are closer to the target and better surrounding the target.

For an example of poor UAV location causing the geo-location particle filter to lose track of the target, see Figure 6.3. A noticeable deviation in the geo-location estimate occurs towards the

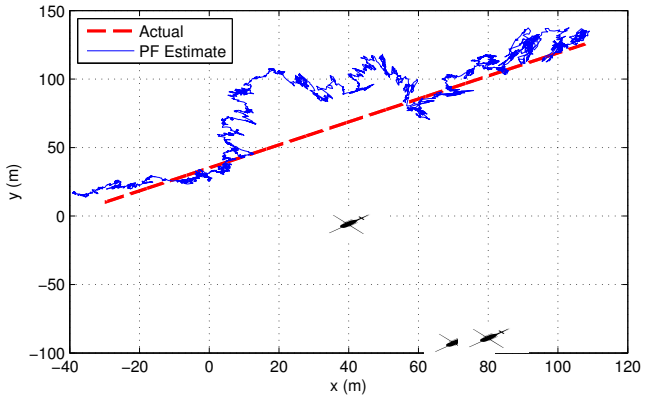


Figure 6.3: *Example of target geo-location with poor UAV placement.*

middle of the 16th test iteration. The placement of the three UAVs on only one side of the target path, along with the placement of two UAVs near each other, results in a poor geo-location accuracy throughout the entire test run.

At the point where the geo-location track begins to deviate ($p^{\tau}(t) \approx \begin{bmatrix} 10 & 40 & 0 \end{bmatrix}^{T}$), the observation vector for all three UAVs are nearly parallel, effectively providing only a single vantage of the target. The geo-location particle filter continues to roughly track the direction of the target motion, but can not estimate the range of the target with any accuracy, pushing the estimate far to one side of the actual target path. As the target continues, the observation vectors between the two southern UAVs and the northern UAV begin to separate, increasing the ability to determine range and improving the geo-location accuracy.

These tests cases and their different results highlight the need for deliberate UAV placement near and around the target of interest. Although the majority of the test cases provide a good geo-location accuracy, as shown in Figure 6.1, certain UAV placements and edge cases may reduce the ability of the geo-location particle filter to estimate the all or part of the target state.

Estimator Accuracy — Figure 6.4 shows the average estimator error for all 50 runs of the base test.

Before 100 seconds, the estimator error steadily increases from around 5 meters to 9 meters. The exceptional increase in error around 30 seconds is due to the test iteration where the geo-location particle filter loses track of the target due to poor UAV placement. Besides this singular event, the steady increase in error (or decrease in accuracy) is attributed to the target moving farther

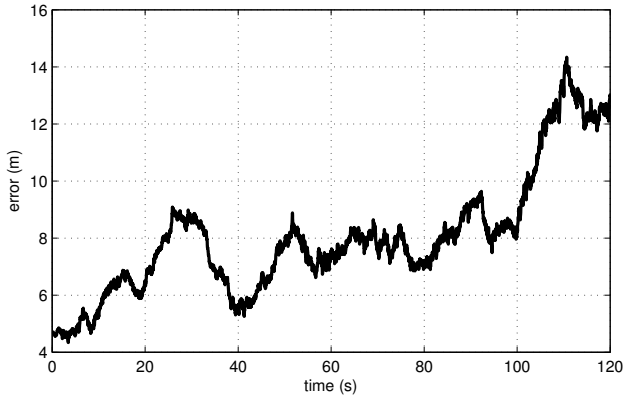


Figure 6.4: *Geo-location particle filter estimation error.*

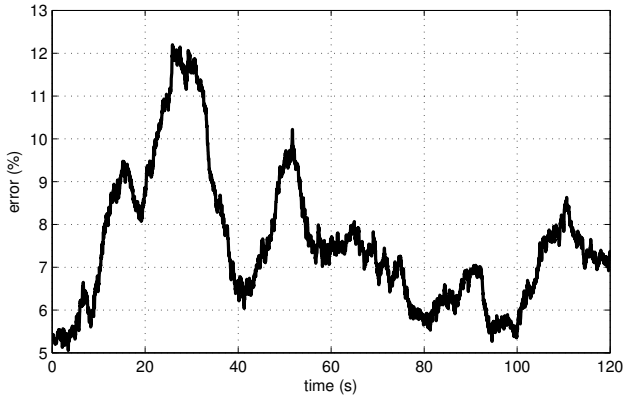


Figure 6.5: *Geo-location particle filter percent error.*

away from the centroid of the region of UAV placement. It is expected that the accuracy of the estimator decreases as the target moves farther from its observers.

Also of note is the behavior of the estimator error after 100 seconds. It is around this time that the target is forced to leave the convex hull formed by the UAV placement. Within this region, there is a high probability that the UAVs may surround the target. Outside this region, the target is not contained by the UAVs, and the ability to estimate the target location decreases significantly. This is eveident in the sharp increase in geo-location estimator error.

Estimation Percent Error — In Figure 6.5, it is shown that the percent error is roughly between 6 and 9 percent for the base test. The large anomaly around 30 seconds is due to the poor UAV placement during few test runs.

The low estimation percent error is equivalent to the statement that for every 100m between the UAVs and the target, there is a roughly 6-9m error in the target geo-location.

Table 6.1: *Transition probability matrices* Π.

Derived mode change probability	Markov switching transition matrix
$\Pi_d = \begin{bmatrix} 0.995 & 0.005 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.992 & 0.002 & 0.004 & 0.001 \\ 0.003 & 0.001 & 0.991 & 0.002 & 0.002 \\ 0.000 & 0.002 & 0.002 & 0.995 & 0.001 \\ 0.000 & 0.003 & 0.001 & 0.002 & 0.993 \end{bmatrix}$	$\Pi_p = \begin{bmatrix} 0.990 & 0.010 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.986 & 0.004 & 0.008 & 0.002 \\ 0.006 & 0.002 & 0.984 & 0.004 & 0.004 \\ 0.000 & 0.004 & 0.004 & 0.990 & 0.002 \\ 0.000 & 0.006 & 0.002 & 0.004 & 0.988 \end{bmatrix}$

6.2 MANEUVERING TARGET CASE

In this test case, target maneuvers are added according to the driving car scheme in Section 3.6. The initial target position is set to a random location for each test run, chosen as $p^\tau \sim \begin{bmatrix} U(-25,25) & U(-25,25) & 0 \end{bmatrix}^T$, with initial velocity $v^\tau \sim \begin{bmatrix} U(-10,10) & U(-10,10) & 0 \end{bmatrix}$ and a yaw angle matching the velocity vector (*i.e.* the car is travelling forward, not in reverse), and no acceleration.

The initial driving mode for the target is chosen randomly from the five available modes: STOP, ACCEL, DECEL, COAST, and TURN. The associated parameters for the chosen driving mode, such as the acceleration magnitude or the turning rate, are also chosen at random. During simulation, the car will continue driving in this chosen mode until a new mode is chosen under two possible circumstances: 1) a new mode is chosen at random from the set of allowed modes at the current time, or 2) specific conditions are met that force a mode change, such as decelerating to a complete stop. These modes and mode changes simulate a car maneuvering according to a set of driving rules. The mode switching behavior is not Markovian, as a mode may alter itself without requiring a mode switch, such as a right turn changing to a left turn over time, although set probabilities are supplied for the purposes of simulating decisions.

The geo-location particle filter is accordingly augmented to simulate these multiple modes of driving behavior. However, the multi-modal behavior of the particles strictly follows a Markovian switching system design. A 5x5 transition probability matrix Π is designed based on the probablities of mode changes derived from the previously described behavior. To faciliate mode switches among the relatively small number of particles (50), the probabilities that a particle will switch to a new mode are double their derived values. In this way, particles are more likely to seek alternative mode solutions than the actual existence of those solutions, without completely abandoning a correct solution when found.

Table 6.1 shows both the derived mode change probabilities, and the Markov switching prob-

abilities in the form of transition matrices. Each row contains probabilities the current mode (sorted by [STOP, ACCEL, DECEL, COAST, TURN]) will switch to a new mode according to the column the probability value lies in (sorted the same as the rows). For example, if the target is believed to be operating in mode ACCEL, or row 2, it has a 0.8% probability that it will switch to mode COAST, or row 2 column 4.

Note that the rows sum to one only in the matrix Π_p associated with the particle filter. For the derived probabilities, an extra column is not shown containing the probabilities that a mode changes internally without switching to another mode. This is inconsistency is not allowed in the Markov switching system for the target behavior, so it is rolled into the matrix elements that govern the probability a mode will not switch. The random particle behavior within each mode generates solutions for such internal mode changes.

As noted in Section 2.1.5, particles may not be restricted to operating in the same mode as the actual mode of the vehicle. For instance, while the vehicle is turning, not all particles must be updated according to the turning behavior. Because the mode transitions enforce C^1 continuity, and most target behaviors appear linear for short periods of time, most target modes appear indistinguishable over short periods. Over longer periods of time, this is not true, and particles that have switched to a correct mode will tend to be favored by the particle filter resampling, and this in turn will correct the full estimated target state over successive iterations. Therefore, the multi-modal particle filter improves target state estimation for maneuvering target behavior, but particles do not necessarily estimate the correct target mode.

Along with the mode additions, the same test parameters are used for the multi-modal tests as for the base test case: three stationary observers placed randomly and a particle filter using 50 particles. A total of 50 test runs are performed, with 180 seconds per run at 40 computation cycles per second.

Typical Multi-Modal Run — Figure 6.6 shows one of the 50 test runs for the maneuvering target test case. In this case, the UAVs are placed relatively far apart, but not in a manner that provides optimal vantage for the target throughout the run. However, the location estimate tracks the actual target location roughly well, with a relative estimator error that ranges between 2-12%.

Of importance is the number of turns that the target performs during a typical run. Each turn visually represents both a non-linearity in the target state space model and a sudden maneuver

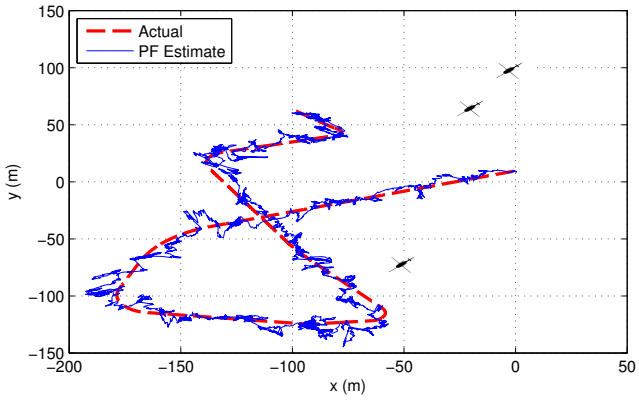


Figure 6.6: *Example maneuvering target test run.*

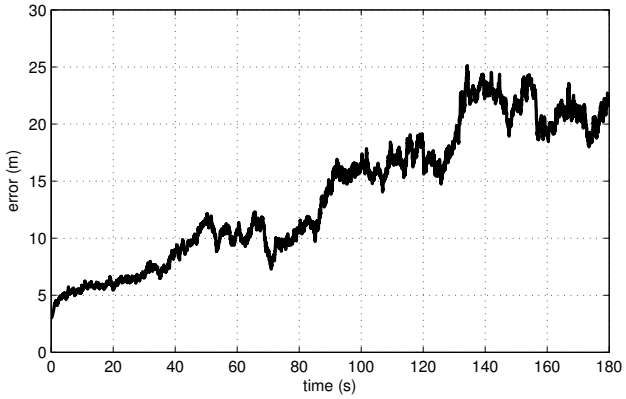


Figure 6.7: *Average estimator accuracy for maneuvering target case.*

requiring an alteration to the assumed target behavior. Other mode changes such as stops and linear acceleration exist but are not easily visually identifiable, nor is it possible to show the altitude of the vehicle from this vantage, which is parallel to the z-axis. However, the performance metrics account for all three axes and will substitute for a visual identification of estimator behavior.

Although the following performance metrics are averaged across all 50 test runs, the non-deterministic state of the target across the runs precludes examining the average behavior of the target location estimate itself.

Estimator Accuracy — Unlike the base test case, the maneuvering target runs for 60 seconds longer and runs at much faster speeds for stretches of time. Therefore, the target travels much farther from the starting position and the UAVS. The estimator accuracy, shown in Figure 6.7, is averaged over all 50 runs. As expected, the accuracy decreases (location error increases) as the target moves farther away.

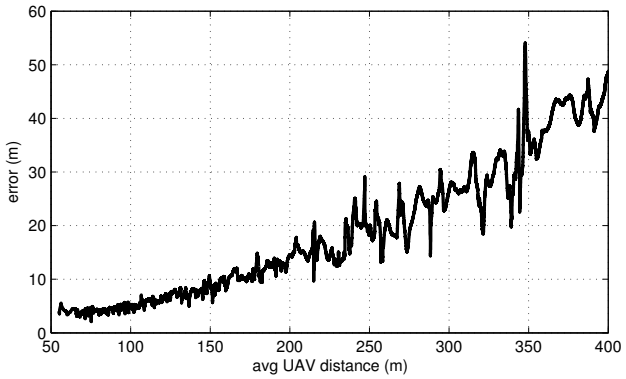


Figure 6.8: *Error vs. distance for maneuvering target case.*

This, along with the error versus distance plot below, highlights the necessity for a UAV control scheme that attempts to follow the target, keeping it within a specific distance. While the relative location error remains relatively steady over increasing distance between the UAVs and the target, a maximum location error can only be enforced by restricting the UAV distance.

Error vs. Distance — Figure 6.8 plots the location estimate error as a function of the average distance from the three UAVs. The distance is computed as the average distance from all UAVs to the target at each point in time. A hard lower limit of roughly five meters estimation error exists below 100 meters. This is a limit of the process noise required by the particle filter to acquire state estimate solutions and to retain track of the target at all times. Beond 100 meters, this process noise is overcome by the observation noise, which linearly increases in proportion to the distance of the target from the observer(s). It is apparent that the location estimate error follows this linear trend, with about 5-10 meters of location error for every 100 meters distance from the UAVs to the target. This is also shown in the percent error metric.

It must be noted that there is fewer collected data for average distances above 300 meters, as few targets managed to reach such distances. However, the roughly linear trend is still apparent. For certain targets, it may be infeasible to design an image detection system that can reliably detect a target within an image frame at such distances.

Estimation Percent Error — Like in the base test case, the relative location error, given as percent error, is quite steady over all runs (Figure 6.9). On average, the relative error is between about 5-8% error, or 5-8 meters in location estimate error for every 100 meters between the target and UAVs.

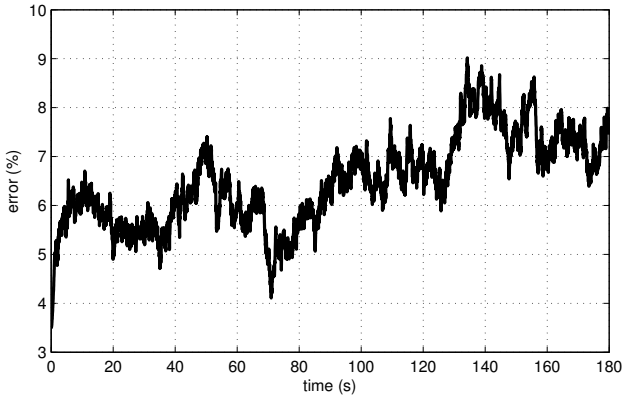


Figure 6.9: *Relative error for maneuvering target case.*

The initial low estimation error (near 0 meters) during the first few cycles of each simulation is due to a unique situation in the simulation when the observation measurements have yet to show noticeable uncertainty. The simulation is reliable after at most one second of operation, or 40 cycles.

Here, a comparison may be made between the location error and the observation error, which is assumed to be the major contributor to uncertainty in calculating the target location. The observer error amounts to an uncertainty in the direction vector from the center of observation to the target.

In a two-dimensional problem, the geometric location may be reduced to $p = d * sin(\psi)$, where p is the location of the target in the y-axis, d is the distance to the target from the observer along the x-axis, and ψ is the yaw to the target from the observer. A fixed uncertainty in ψ translates to a fixed error in p that is linearly dependent on the distance d. With a fixed compound uncertainty assumed for ψ, a hard upper bound on the relative error may be derived from $p/d = sin(\sigma_\psi)$. For percent error (error relative to 100 meters), the upper error bound for an assumed $\sigma_\psi = 0.3$ is $p/d = sin(0.3) * 100 \approx 29.5\%$. The resulting percent error for the maneuvering target case, which is roughly between 5 and 8 percent, shows a location estimation improvement of 400-600% over the upper bound of 29.5%.

Target Mode Estimation Accuracy —

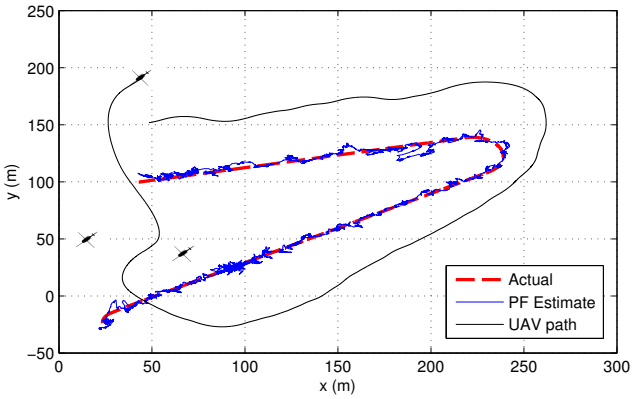


Figure 6.10: *Example mobile UAV test run.*

6.3 MOBILE UAV PLATFORM

As the previous test results make clear, the accuracy of the geo-location particle filter is dependent on the distance from the UAVs to the target. It is necessary for the UAV to attempt to stay within a maximum distance from the target to minimize the estimator location error. This presents no problem for the highly mobile rotorcraft UAV platform.

For the tests in this section, the UAV trajectory generator and controller described in Chapter 4 have been activated. The UAV parameters are the same as those described in Tables 4.1 and 4.2, which match the model parameters for a small-scale model helicopter developed at U. C. Berkeley. Three UAVs are included from each test run, each operating at a distinct fixed height, beginning at 30 meters above the ground and increasing in increments of ten meters per UAV. The outer target radius is $\mathcal{R} = 80m$ and the inner orbiting radius is $\zeta\mathcal{R} = 50m$. Without a more sophisticated placement algorithm, it is not guaranteed the UAVs will reach unique vantages, so each UAV is assigned to orbit either clockwise or counter-clockwise according to the height it operates at, with successively higher UAVs alternating orbit directions as a naive scheme to minimize the GDOP for at least part of the orbit cycle. This increases the likelihood of unique vantages, or viewing angles, throughout the simulation run.

All other testing parameters remain the same as previous tests. A total of 50 test runs are performed with 50 particles used in the geo-location particle filter. Each test is run for 180 seconds at 40 computation cycles per second.

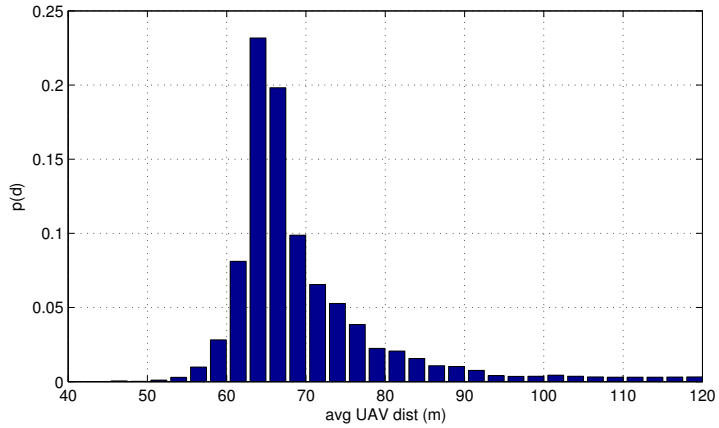


Figure 6.11: *Histogram of average UAV distance, d, to target.*

Typical Mobile UAV Run — For clarity, only one UAV path is shown in Figure 6.10. The UAV path shown follows the estimated target location during the run. Initially, the UAV begins about 200 meters away from the target and attempts to close this distance. Once the UAV is within the prescribed range, it attempts to orbit the target in a counter-clockwise pattern. The UAV follows the target while it is in motion, which is for the majority of this test run.

UAV Distance from Target — Figure 6.11 is a histogram of the average UAV distance to the target over all runs. For the majority of the runs, the UAVs maintain a distance to the target between 60-80 meters. Time spent above 80 meters distance is due to the UAVs approaching the target at the beginning of each run. Conversely, very little time is spent below 50 meters to target, which is considered too close according to the test configuration.

The histogram confirms the desired behavior of the UAV control system and trajectory generator, which is to keep the UAVs below a maximum distance to the target at all times, if possible.

Estimator Accuracy —

Estimation Percent Error — The geo-location estimation absolute error and relative error shown in Figures 6.12 and 6.13 validate the advantage of restricting the range of the UAVs to the target. After 40 seconds and once the UAVs have approached the target, the absolute estimation error is limited to less than five meters. This is confirmed by the relative estimation error, which shows that for UAV-to-target distances of around 70 meters a relative error of around seven percent is maintained.

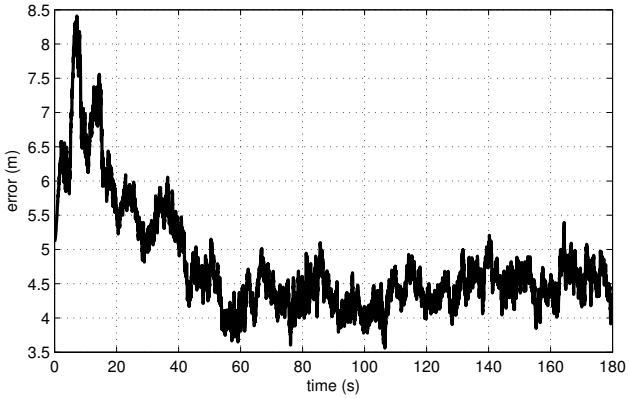


Figure 6.12: *Average estimator accuracy for mobile UAV case.*

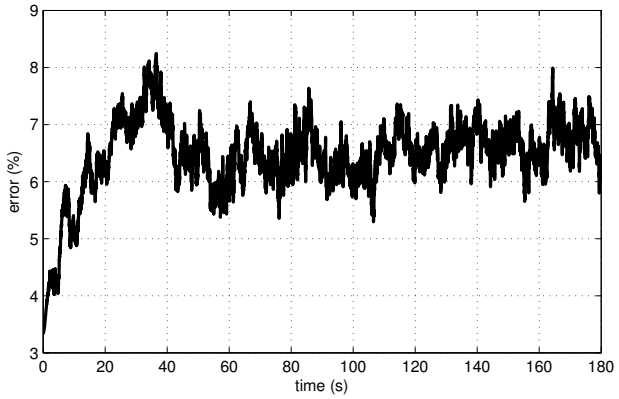


Figure 6.13: *Relative error for mobile UAV case.*

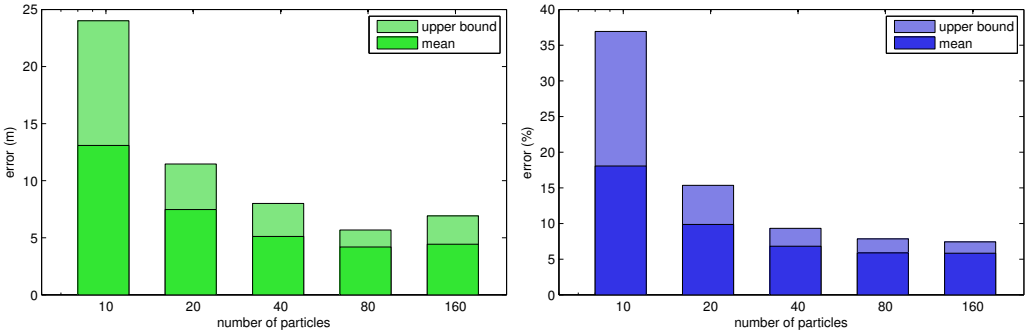


Figure 6.14: *Geo-location estimator aboslute error (left) and relative error (right) vs. number of particles in use.*

6.4 EFFECT OF NUMBER OF PARTICLES

Estimator Accuracy and Percent Error —

Estimator Percent Error — Figure 6.14 relates how the location estimation error changes with respect to the number of particles in use by the geo-location particle filter. As the number of particles decreases, the location estimation error increases exponentially. This result is expected and is due to the reduced amount of space that the particles can cover in numbers. As the number of particles increases, the probability that one or more particles will achieve the correct state increases to one. However, above a certain number of particles–between 40 and 80 particles according to the experiments–the minimum possible estimator error is reached and the accuracy of the particle filter no longer improves.

Through similar analysis of estimator error, simulation data, and real flight test data, it is possible to optimize the number of particles in use to minimize excessive computation cycles and memory allocation. Also, through experimentation, it is possible to train a reasoning system to alter the number of particles in use according to mission, flight, and environmental conditions [26]. These optimizations are beyond the scope of this research.

Average UAV Distance — Figure 6.15 verifies that the UAV control algorithm is minimally affected by the number of particles in use by the geo-location particle filter. Although the location estimation error increases as the number of particles decreases, and the UAV trajectory generation and control is dependent on the accuracy of the location estimate, the UAV maintains the desired distance to the target, on average during the duration of the tests. The robustness of the UAV

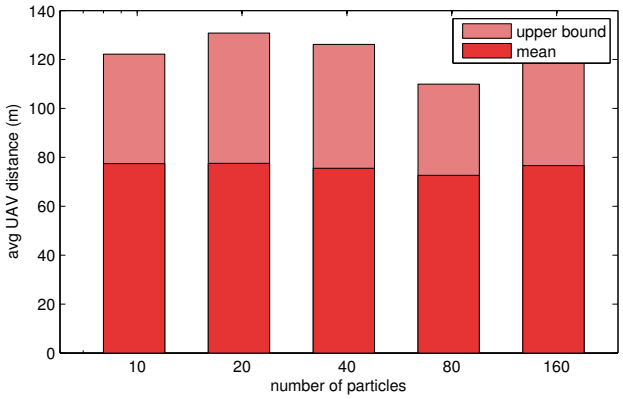


Figure 6.15: *Average UAV distance to target vs. number of particles.*

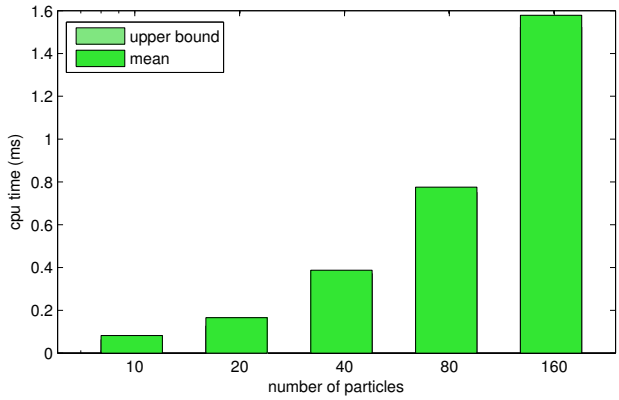


Figure 6.16: *Algorithm CPU time for geo-location particle filter algorithm.*

control and the generated trajectory prevent the UAV from deviating too far from the estimated location of the target, however poor that estimation may be.

CPU Time — Figure 6.16 shows the total CPU processing time required per iteration by the geo-location particle filter in relation to the number of particles in use. Because each particle is manipulated independently, there is a linear relationship between the processing time and the number of particles. According to the experiment data, a single particle requires about 10 µs of processing time per iteration. Extrapolated linearly to 160 particles, this translates to around 1.6 ms of processing time, which coincides with the experimental results seen in Figure 6.16.

For a large number of particles, here 160 particles, the total processing time per iteration remains well below the upper limit for allowed processing time–25 ms allowed by the 40 Hz system update rate. By extrapolating to the upper bound of 25 ms, it is found that the total number of particles allowed is around 2500, a very large number of particles. This extremely high limit allows for

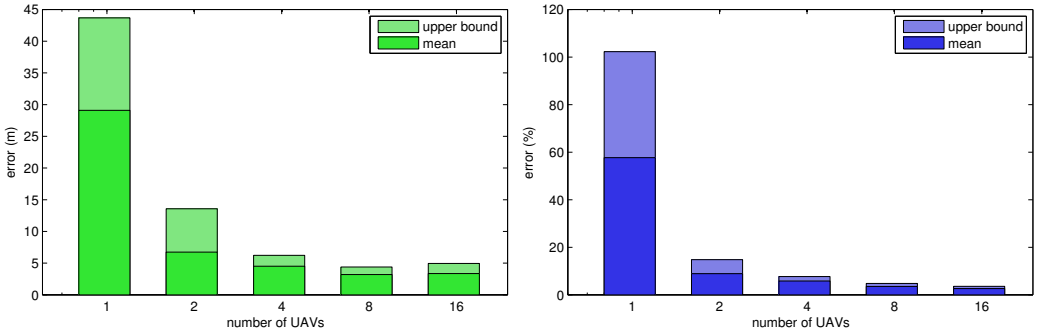


Figure 6.17: *Geo-location estimator absolute error (left) and relative error (right) vs. number of UAVs.*

more complex and sophisticated refinements of the geo-location particle filter algorithm.

6.5 EFFECT OF NUMBER OF UAVS

The variable number of UAVs is tested in this series of experiments. Five separate cases are run, with one, two, four, eight, and sixteen UAVs simultaneously geo-locating and tracking the maneuvering target per case. Each UAV is operating at maximum capacity: all target modes and mode switching behavior are estimated, and the UAVs are controlled and restricted to remain within a fixed radius of the target in the $x - y$ plane. Each test case is run ten times, for a total of 50 test runs, and performance metrics are extracted and collated for each case independently.

Estimator Accuracy and Percent Error — Figure 6.17 shows both the absolute error and the relative error in relation to the number of UAVs present during the test. As expected, the accuracy of the geo-location particle filter increases as more UAVs are added to the mission. A single UAV may only view the target from a single vantage, and thus has limited information from which to estimate the location of the target. As more UAVs are added, more vantages are available and thus more information is added to the location estimation system.

A single UAV performs poorly, but the estimation rapdily improves for each UAV added to the system, wth diminishing returns for larger numbers of additional UAVs. The improvement in estimation performance–in relation to location error–is roughly 600% when the total number of UAVs is increased to two. The minimum location error for this configuration, an error of around five meters, is achieved when the total number of UAVs is between four and eight.

As with the variable number of particles, it is possible to optimize the number of UAVs participating in the geo-location of a single target. Mission and environemntal parameters play an

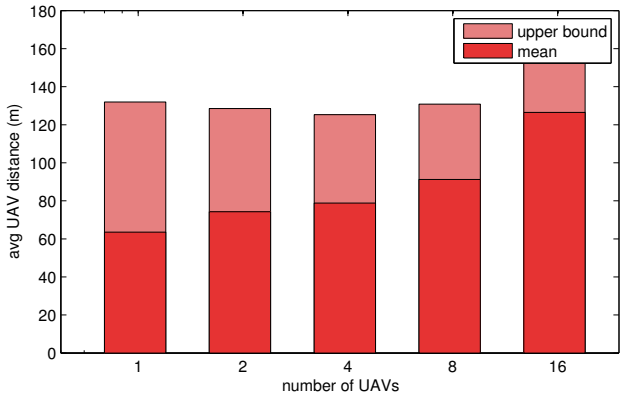


Figure 6.18: *Average UAV distance to target vs. number of UAVs.*

important role in the ability of each UAV to detect, locate, and track a target, and must be taken into account to achieve the right balance between too many and too few UAVs. For scenarios requiring tracking of multiple targets, the ability to optimize the number of UAVs tracking each target enables efficient resource allocation and improves the capabilities of the entire multi-UAV system.

Average UAV distance — The average distance of the UAVs to the target is maintained for both low and high numbers of UAVs, as shown in Figure 6.18. The robustness of the particle filter geo-locator and the UAV control and trajectory generator and these systems ability to work in concert are key to maintaining the desired distance to the target.

The anomaly of the average UAV distance increasing as the number of UAVs increases is an artifact of the trajectory generator. In particular, the desired height of each UAV is restricted to a single value to reduce the chances of collisions between UAVs. As more UAVs are added to the system, more UAVs are restricted to higher altitudes and are farther away from the target because of their height restriction. A more advanced airspace deconfliciton algorithm may be incorporated into the trajectory generator to allow more UAVs to operate at lower heights, closer to the target.

CPU Time — Figure 6.19 shows the CPU time spent processing the weight update function algorithm, and the CPU time spect processing the remaining portion of the particle filter geo-location algorithm. The sum of these two CPU times is the total processing time for the entire geo-location algorithm.

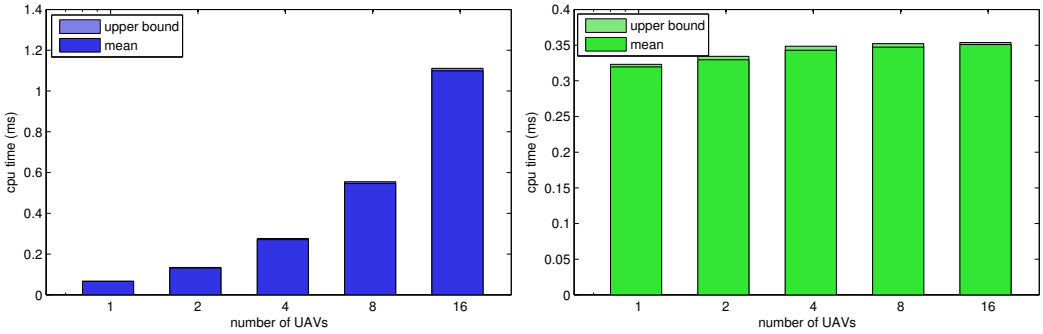


Figure 6.19: *Algorithm CPU time for particle filter weight update (left) and remaining particle filter algorithm (right).*

It is seen that only the weight update function is affected by the nubmer of UAVs present in the mission. The observation function for the camera system, independently computed for each UAV, is the only interaction between the UAVs and the geo-location particle filter.

More importantly, it is seen that the weight update function processing time is linearly dependent on the number of UAVs present. As explained in Section 3.5, because all UAVs operate independently of each other, the likelihood function may be calculated as a product of the individual likelihoods derived for each UAV. The computation time for this product is negligible compared to the computation time for each likelihood function, and so the total computation time for the weight update function is linearly dependent on the number of UAVs, as is evident from the experiment data.

Also of note is how much processing time is spent during the weight update versus the rest of the geo-location particle filter algorithm. Minus the weight update function, the particle filter algorithm completes in 0.3-0.35 ms per iteration. As the number of UAVs increases from one to sixteen, the processing time for the weight update function increases from 0.1 to 1.1 ms, or 25% to 75% of the total geo-location particle filter processing time. Even though these tests remain well below the upper limit of 25 ms of total processing time per iteration–25 ms allowed by the 40 Hz system update rate–the total number of UAVs is limited by this upper bound. By extrapolating these results, it is found that the limit of the number of UAVs is in the hundreds, or specifically 300 UAVs. This extremely high limit allows for more complex and sophisticated refinements of the geo-location particle filter algorithm.

CHAPTER VII

CONCLUSIONS

The result of this research is a geo-location particle filter that accepts video sensor and image frame data to determine the location of a target in world-space coordinates. The geo-location particle filter reliably locates the target with no measurements of or assumptions on the distance from the UAV to the target. This required modeling the projection of three-dimensional data to two dimensions according to the structure of the video sensor, as well as incorporating uncertainty of the position and orientation of the UAV, and subsequently the on-board video camera. Additionally, approximations of the target motion are used to improve the target location estimate, which necessitated construction of a Markov switching model to describe the target motion. And, a scalable architecture supports the addition of multiple UAVs to the geo-location problem with minimal cost addition in processing time. This research attempts to close the loop in target tracking architecture by providing a solution for target geo-location. Small-scale UAVs can inexpensively and reliably locate a target in world-space coordinates using cheap sensing technologies and limited sensor information.

7.1 CONTRIBUTIONS

- A robust particle filtering algorithm capable of geo-locating a target from a UAV platform using limited video sensor information.
- A multi-modal state estimation solution for target behavior and location estimation.
- A real-time geo-location solution with flexibility in design complexity and robustness to maximize probability of mission completion.
- A scalable framework for the addition and removal of multiple UAVs, with minimal computation cost requirements.
- A closed-loop target tracking system, using the geo-location particle filter and a UAV control and trajectory generator to minimize the geo-location estimation error.

7.2 FUTURE WORK

This research focuses primarily on tracking a target modeled after a car. Further work is needed to refine the target behavior model used by the geo-location particle filter to further improve location estimation results. Additionally, other models may be derived to approximate other types of targets, such as aerial vehicles, underwater vehicles, and humans, and to accommodate for different reactions and varied behavior modes.

The geo-location particle filter is designed to accept video sensor and image frame data as measurements of the target location, but the particle filter is flexible enough to accept many different sensor and data types. Further work may examine heterogeneous sensing strategies, including non-UAV observation platforms.

BIBLIOGRAPHY

[1] "Predator rq-1 / mq-1 / mq-9 reaper uav, united states of america." airforce-technology.com, Net Resources International, 2011. <http://www.airforce-technology.com/projects/predator-uav/>.

[2] Arulampalam, M. S., Maskell, S., Gordon, N., and Clapp, T., "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," in *IEEE Transactions on Signal Processing*, vol. 50, February 2002.

[3] Bar-Shalom, Y. and Li, X.-R., *Estimation and Tracking: Principles, Techniques, and Software*. Artech House, 1993.

[4] Bar-Shalom, Y. and Li, X.-R., *Multitarget-Multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, 1 ed., 1995. ISBN 0-9648312-0-1.

[5] Bar-Shalom, Y., Li, X.-R., and Kirubarajan, T., *Estimation with Applications to Tracking and Navigation*. New York: Wiley-Interscience, 1 ed., June 2001. ISBN 0-471-41655-x.

[6] Barber, D., Redding, J., McLain, T., Beard, R., and Taylor, C., "Vision-based target geo-location using a fixed-wing miniature air vehicle," *Journal of Intelligent and Robotic Systems*, vol. 47, no. 4, pp. 361–382, 2006.

[7] Braybrook, R., "Three "d" missions–dull, dirty and dangerous!," *Armada International*, vol. 28, pp. 10–12, Feb-Mar 2004.

[8] Conte, G., Hempel, M., Rudol, P., Lundström, D., Duranti, S., Wzorek, M., and Doherty, P., "High accuracy ground target geo-location using autonomous micro aerial vehicle platforms," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, (Honolulu, HI), August 18-21, 2008.

[9] Craig, J. J., *Introduction to Robotics: Mechanics and Control*. Pearson/Prentice Hall, 3rd ed., 2004.

[10] Devroye, L. and Györfi, L., *Nonparametric Density Estimation : The L_1 View*. Wiley Series in Probability and Mathematical Statistics, New York: John Wiley & Sons, 1985.

[11] Doucet, A. and Gordon, N., "Efficient particle filters for tracking manoeuvring targets in clutter," in *IEE Colloquium on Target Tracking: Algorithms and Applications*, (London), pp. 4/1–4/5, November 11-12, 1999.

[12] Fiume, E., *Mathematical Foundations for Computer Graphics.* Academic Press, 1989.

[13] Gibbins, D., Roberts, P., and Swierkowski, L., "A video geo-location and image enhancement tool for small unmanned air vehicles (uavs)," in *Proc. 2004 IEEE Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 469–473, December 14-17, 2004.

[14] Gordon, N. J., Salmond, D. J., and Smith, A. F. M., "Novel approach to nonlinear/non-gaussian bayesian state estimation," in *IEE Proc. F. Radar and Signal Processing*, vol. 140, pp. 107–113, April 1993.

[15] Hastings, W., "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[16] Hogg, R. V., McKean, J. W., and Craig, A. T., *Introduction to mathematical statistics.* Upper Saddle River, New Jersey: Prentice Hall, 6th ed., 2004.

[17] Isidori, A., Marconi, L., and Serrani, A., "Robust nonlinear motion control of a helicopter," in *IEEE Transactions on Automatic Control*, vol. 48, pp. 413–426, 2003.

[18] Johnson, W., *Helicopter Theory.* Princeton University Press, 1980.

[19] Jones, P., *Cooperative Area Surveillance Strategies Using Multiple Unmanned Systems.* Ph.d. dissertation, Georgia Institute of Technology, Atlanta, GA, 2009.

[20] Kamel, H. and Badawy, W., "Comparison between smoothing and auxiliary particle filter in tracking a meuverable target in a multiple sensor network," in *Proc. SPIE* (Masten, M. K. and Stockum, L. A., eds.), vol. 5810, pp. 74–81, June 29, 2005.

[21] Karlsson, R. and Bergman, N., "Auxiliary particle filters for tracking a maneuvering target," in *Proc. of the 39th IEEE Conference on Decision and Control*, vol. 4, (Sydney, NSW), pp. 3891–3895, December 12-15, 2000.

[22] Kenney, J. and Keeping, E., *Mathematics of Statistics, Part 2.* Princeton, NJ: Van Nostrand, second edition ed., 1951.

[23] Koo, T. J. and Sastry, S., "Output tracking control design of a helicopter model based on approximate linearization.," in *Proceedings of the 37th IEEE Conference on Decision and Control*, vol. 4, pp. 3635–3640, Dec 16-18 1998.

[24] Lepage, G., *VEGAS: An Adaptive Multi-dimensional Integration Program.* Cornell, 1980.

[25] Li, Y.-Q., Shen, Y., and Liu, Z.-Y., "A new smoothing particle filter for tracking a maneuvering target," in *Proc. of the 2nd International Conference on Machine Learning and Cybernetics, 2003*, vol. 2, (Xi'an, Shaanxi), pp. 1004–1008, November 2-5, 2003.

[26] Ludington, B., *Particle Filter Tracking Architecture for use Onboard Unmanned Aerial Vehicles.* Phd dissertation, Georgia Institute of Technology, Atlanta, GA, 2006.

[27] Ludington, B., Johnson, E., and Vachtsevanos, G. J., "Augmenting uav autonomy: Vision-based navigation and target tracking for unmanned aerial vehicles," in *IEEE Robotics and Automation Magazine*, vol. 13, pp. 63–71, 2006.

[28] Manolakis, D. E., "Efficient solution and performance analysis of 3-d position estimation by trilateration," in *IEEE Transactions on Aerospace and electronic Systems*, vol. 32, pp. 1239–1248, October 1996.

[29] Mao, G., Drake, S., and Anderson, B. D., "Design of an extended kalman filter for uav localization," in *Information, Decision and Control, 2007*, (Adelaide, Qld), pp. 224–229, February 12-14, 2007.

[30] Marconi, L. and Naldi, R., "Robust full degree-of-freedom tracking control of a helicopter," *Automatica*, vol. 43, pp. 1909–1920, 2007.

[31] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E., "Equations of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[32] Mettler, B., *Identification Modeling and Characteristics of Miniature Rotorcraft.* Kluwer Academic Publishers, 2003.

[33] Monda, M., Woolsey, C., and Reddy, C., "Ground target localization and tracking in a riverine environment from a uav with a gimbaled camera," in *AIAA Guidance, Navigation, and Control Conference*, no. 6747–6750, (Hilton Head, SC), August 2007.

[34] Morelande, M. R., Challa, S., and Gordon, N., "A study of the application of particle filters to single target tracking problems," in *Proc. SPIE*, vol. 5204, pp. 211–222, January 21, 2003.

[35] Murphy, K. and Russell, S., *Rao-Blackwellised particle filtering for dynamic Bayesian networks*, ch. 24, pp. 499–515. Springer, 2001.

[36] Musso, C., Oudjane, N., and Gland, F. L., *Sequential Monte Carlo Methods in Practice*, ch. Improving Regularized Particle Filters, pp. 247–271. New York: Springer, 2001. ISBN 0-387-95146-6.

[37] Newman, M. and Barkema, G., *Monte Carlo Methods in Statistical Physics*. Clarendon Press, 1999.

[38] Oudjane, N. and Musso, C., "Multiple model particle filter," in *Actes du 17ème Colloque GRETSI*, (Vannes), pp. 681–684, 13-17 septembre 1999.

[39] Plett, G. L., DeLima, P., and Pack, D. J., "Target localization using multiple uavs with sensor fusion via sigma-point kalman filtering," in *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, (Rohnert Park, CA), May 7-10, 2007.

[40] Quigley, M., Goodrich, M., Griffiths, S., Eldredge, A., and Beard, R., "Target acquisition, localization, and surveillance using a fixed-wing mini-uav and gimballed camera.," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, (Barcelona, Spain), pp. 2600–2605, April, 2005.

[41] Raptis, I. A. and Valavanis, K. P., *Linear and Nonlinear Control of Small-Scale Unmanned Helicopters*. Springer, 2011.

[42] Raptis, I. A., Valavanis, K. P., and Moreno, W. A., "A novel nonlinear backstepping controller design for helicopters using the rotation matrix," in *IEEE Transactions on Control Systems Technology*, vol. 19, pp. 465 – 473, 2010.

[43] Ristic, B., Arulampalam, S., and Gordon, N., *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Boston, London: Artech House, 2004. ISBN 1-58053-631-x.

[44] Sathyan, T., Kirubarajan, T., and Sinha, A., "Geolocation of multiple emitters in the presence of clutter," in *Proc. SPIE* (Kadar, I., ed.), vol. 5429, pp. 66–76, September 20, 2004.

[45] Sathyan, T., Sinha, A., and Kirubarajan, T., "Passive geolocation and tracking of an unknown number of emitters," in *Proc. SPIE* (Kadar, I., ed.), vol. 5809, pp. 1–11, May 31, 2005.

[46] Savage, C. O., Cramer, R. L., and Schmitt, H. A., "Tdoa geolocation with the unscented kalman filter," in *Proc. 2006 IEEE Networking, Sensing and Control*, (Ft. Lauderdale, FL), pp. 602–606, April 23-25, 2006.

[47] Sconyers, C., Raptis, I. A., and Vachtsevanos, G. J., "Rotorcraft control and trajectory generation for target tracking," in *19th Mediterranean Conference on Control & Automation*, (Corfu, Greece), pp. 1235–1240, June 20-23, 2011.

[48] Sworder, D. D. and Boyd, J. E., *Estimation Problems in Hybrid Systems*. Cambridge, UK: Cambridge University Press, 1999.

[49] York, G. and Pack, D., "Comparative study on time-varying target localization methods using mutliple unmanned aerial vehicles: Kalman estimation and triangulation techniques," in *Proc. Networking, Sensing and Control, 2005*, (Tucson, AZ), pp. 305–310, March 19-22, 2005.