UNIVERSITY OF CALIFORNIA
SANTA CRUZ

MULTI-UAV COORDINATION FOR UNCERTAINTY SUPPRESSION OF NATURAL DISASTERS

A dissertation submitted in partial satisfaction of the
requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER ENGINEERING

by

Sharon Rabinovich

June 2018

The Dissertation of Sharon Rabinovich
is approved:

Professor Gabriel Elkaim, Chair

Professor Renwick Curry

Professor Qi Gong

Professor Vladimir Dobrokhodov

Tyrus Miller
Vice Provost and Dean of Graduate Studies

Table of Contents

List of Figures

Abstract

Multi-UAV Coordination for Uncertainty Suppression of Natural Disasters

by

Sharon Rabinovich

Containing a wildfire requires an efficient response and persistent monitoring. A crucial aspect is the ability to search for the boundaries of the wildfire by exploring a wide area. However, even as wildfires are increasing today, the number of available monitoring systems that can provide support is decreasing, creating an operational gap and slow response in such urgent situations. Many natural and national security phenomena have a similar need for monitoring the propagating periphery of a hazardous substance or security threat.

The objective of this thesis is to estimate a propagating boundary and create a system that works in real time. It focuses on an autonomous system that suppresses uncertainty, investigating to what extent binary measurements away from the periphery can be used to predict the boundary and its spread rate. It proposes a coordination strategy with a new methodology for estimating the periphery of a propagating phenomenon using quantized observations. The method is tested in a simulation of an autonomous system with multiple unmanned aerial vehicle models that gather local measurements and share the information with a centralized ground control system. The estimate the system generates then incorporated into an allocation algorithm, weighing uncertainty and the rate of change of the uncertainty, reassign the UAV trajectory in order to suppress the uncertainty.

The complete system design, tested on the high-fidelity simulation, demonstrates that steering the vehicles towards the highest perpendicular uncertainty generates the best predictions. The results indicate that the new coordination

scheme has a large beneficial impact on uncertainty suppression. By using the developed coordination algorithm and the adopted flight control system, the vehicles can follow the desired trajectory to reduce the uncertainty and the errors in predicting the periphery across a range of wind and terrain conditions.

To my brothers

Shay

&

David

Acknowledgments

My research began with uncertainty followed by confusion, and it ended up with joy and satisfaction. Learning, for me, was always a combination of success and struggle, a mix of diligence with incompetence, a solution of friendship and competition. I started that journey with people that I love and ended up with more of them. I have met new people that have built-in goodness, where their integrity comes first, and their personal interests come last. I was concerned about the language barrier, and I was surprised to find out that friendship could still emerge and becomes meaningful.

That study track is not a one-person recognition, I have been supported by many, and all have influenced me. My Mom, who has pleaded for our education. My Dad, who fought his lack of confidence by gaining an informal knowledge. My family, with their drive to compete against the ignorance.

My wife, Maya, without whom I would never have had the opportunity to proceed persistently.

The ladder that we use to climb up is not high enough for any roof. That shouldn't prevent us from rising. There will always be someone who reaches his hands to us and pulls us higher.

I wish to thank my adviser, Gabriel Elkaim, for his support, for giving me the chance to see his cleverness and his sensitivity. I am grateful to the members of my dissertation committee: Ren Curry, Vlad Dobrokhodov, Dejan Milutinovic, and Qi Gong, for their excellent assistance and professional advising. This dissertation would not have been possible without the endless effort, priceless advice and intellectual contribution of Ren.

Thank you very much, for pulling me higher.

Chapter 1

Introduction

1.1 Overview

This thesis details the design of a multiple unmanned autonomous vehicle (multi-UAV) monitoring system. The design includes processes for both monitoring particular missions as well as for testing and validating multi-UAV systems. The original autopilot developed for a single aircraft mission to support the new framework has been redesigned, tested with newly integrated hardware components, and implemented a new approach for high-level control on top of the pre-existing Autonomous Systems Lab (ASL) navigation and control software.

Integrating this complex systems required developing a simulation for multi-UAV as a verification and validation step before deploying the software onto real hardware. Thus there was a need for designing multi-UAV software in the loop simulation that could also be used to develop different mission designs.

Although some academic and commercial research has introduced related design aspects, there is no known work that looks specifically at the multi-UAV problem with the rerouting approach developed in this work.

Beyond the theoretical contribution of this work, we developed a multi-UAV

monitoring system prototype. It provides all the functionality of operational periphery monitoring and enables testing of the design. The flight tests and simulation results demonstrate effectiveness and the robustness of the design. A real-time mock-up that includes a conceptual approach facilitates the primary setup needed for developing fully operational system in the future.

In this work, we designed a system prototype with an adaptive multi-UAV high-level controller and an associated simulation framework. Its design was based on the following operational criteria: low cost, long endurance, low altitude, scalability, and robust uncertainty suppression.

1.2 Motivation

In any region undergoing some form of environmental distress, it is very important to detect changes occurring on the ground. In some cases, the environmental incident has spatial changes, and the incident spreads steadily. In other cases, it becomes difficult to follow the incident without knowing how it is evolving. Having a system that follows the event helps rescue human lifes, monitor the incident, and allow the human responders to take better actions (as well as deploy assets in an optimal manner to mitigate the incident).

It is of great importance to monitor and respond to natural phenomena (e.g., fires) and national security disasters (e.g., emitting source). One needs to be able to explore a wide area and search for the source of hazardous substance emmisions or the expansion of a fire front. In 2016 alone, Federal agencies reported 67,595 fires and an estimated cost of fire suppression of approximately 2 billion U.S. dollars [68]. In addition to financial loss and significant damage to the environment, wildfires threaten the lives of firefighters during these fire extinguishing operations [70].

There are two main reasons why a solution to fire tracking needs to be found. The first relates to modeling; it is very challenging to predict the fire frontier as a stochastic phenomenon dependent on weather conditions, terrain, and fuel (flammable materials) [7]. Secondly, operational aspects are exposed to severe limitations and constraints. The resources to respond to and monitor disasters are still quite limited. In the aviation section of the National Interagency Fire Center's annual summary of wildfire activity in 2013, there were many requests for large air tankers, which were Unable To be Filled (UTF). The number of cases of firefighters needing air tankers that were unavailable reached a high of 48 percent in 2012 [69] [1].

An incident with a dangerous spread area requires immediate exploration. Some examples are distributed fire spots and chemical threats, however there are way others. This type of scenario requires surveillance to search for threats, but human observers are likely difficult to use, because the task is dull, dirty, and/or dangerous. Wildfire monitoring missions are a perfect example of why a solution needs to be developed. Wildfires (and all natural and national threats phenomena) require urgent attention and an efficient response to monitor and contain their spread.

1.3 Existing Monitoring Systems

Previous studies have examined two different solutions: one on the ground and one in the air. In the first case, ground vehicles are used to explore the area. Use of ground vehicles depends on how passable is the area which needs to be explored. Ground vehicle capability is not necessarily suitable for scenarios with

[1] This means that in 2012, almost half of all requests for tankers to bomb fires were unanswered due to limited resources

difficult terrain. In the second case, the deployment is in the air, the motion is smooth and the area can be observed much more efficiently. In addition, most of those scenarios have a critical time factor. The systems phenomena is dealing with time, and because the existing system capabilities are limited, they cannot collect all spatial/temporal information at once. Whenever the observing vehicle is positioned in any one place, the system necessarily misses events in other places within the search area.

[1] describes existing projects that support disaster management in real time and mainly explore systems that are space based (e.g., GlobalStar), or high altitude and long duration (e.g., Global Hawk).

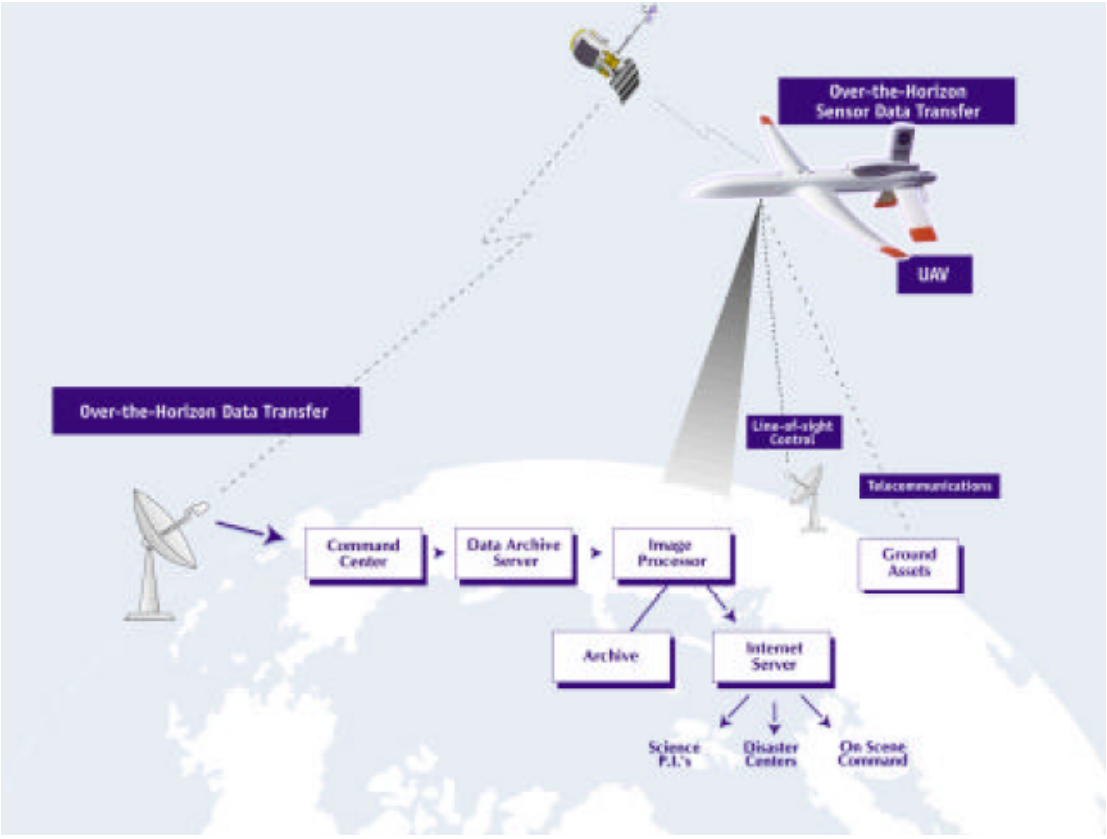


Figure 1.1: Demonstration Elements for UAV Over-the-Horizon projects. An example of an existing system from [1].

The project in Fig. 1.1 reinforces the importance of tracking events like floods and earthquakes, and how the tracking events help to monitor the incidents and handle them effectively from the ground control segment.

[99] describes remote sensing techniques and sensor packages that have been used on UAVs. The author argues that these techniques can serve as the main solution for various disasters. Reviewing the literature on the development of UAVs, including projects with different types of sensors (IR/Visual) and platforms (fixed-wing/rotary-wing), he concludes that Multi-UAVs can be used to avoid the drawbacks of approaches that are based on either land or space.

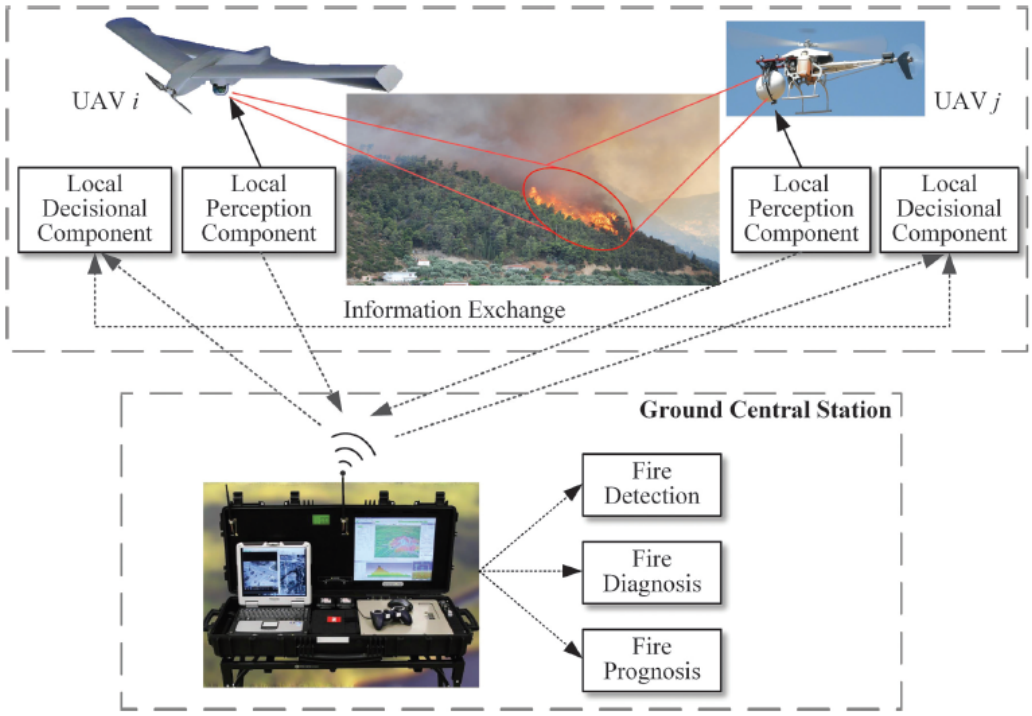


Figure 1.2: A conceptual UAV-based forest fire monitoring system (from the survey [99]). The concept covers the functions of monitoring: detection, diagnosis, and prognosis

1.4 Multi-UAV

The ideal mission for a UAV is monitoring a wide area and searching for the source of emission of a hazardous substance or expansion of a fire front. There has been a rising interest in increasing UAV efficiency and reliability. Autonomous vehicles have been used in a variety of applications for surveillance, search, and exploration [73]. In surveillance problems, the target space needs to be surveyed continuously. It differs from a coverage problem, which involves optimal deployment of the sensors for complete coverage of the target area. It also differs from an exploration problem, which deals with gaining more information about the bounded area [25]. This exploration research moves in two directions. The first focuses on how to pinpoint the source of an odor [56], [83]. In this area of research, robots are tasked with detecting, tracking and seeking the odor source efficiently.

The second direction of this exploration research focuses on how to establish the boundary or perimeter of a spreading phenomenon in order to monitor the area and prevent human exposure to risk [97]. Because of the spatial limitations of a single UAV, most research currently focuses on how to monitor large areas by operating multiple inexpensive simple UAVs simultaneously [15].

Though the studies mentioned above are significant, they focus on exploring the environment using clues (e.g., aerosol diffusion) for tracing emission sources. Moreover, the techniques used to detect the plume or periphery are strongly dependent on the spatial gradient change of the underlying tracked phenomenon. The research presented in this work proposes to explore the area by using approximate inference methods [85] and statistical reasoning [80]. The developed method takes into consideration the operational aspect of the mission in addition to the statistical characteristics of the underlying phenomenon.

1.4.1 Coordination

Most of the multi-UAV systems are designed to address problems related to specific research in a particular environment of interest. The UAVs cooperate and share data to obtain information on a certain aspect of the environment. Regardless of the number of UAVs and size of the AOI (Area Of Interest), cooperative systems deliver an improved overall picture of the environment through coordination.

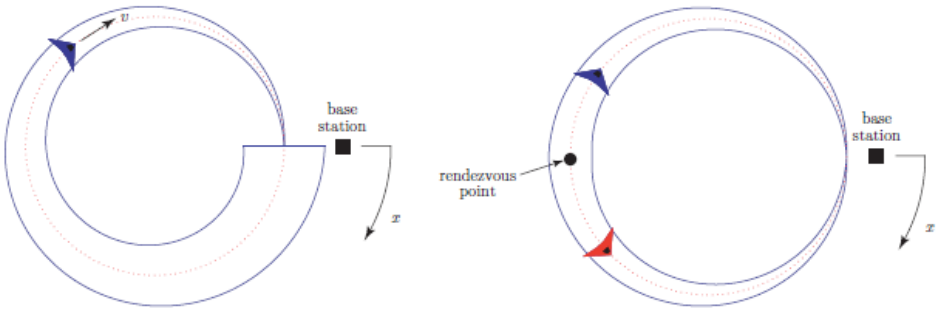
There are many studies on multi-UAV cooperative control systems that address coordination issues. These focus on designing a system to control and monitor a region. One of the earliest studies proposed using aerial photographs to monitor fires in order to combat them [5]. The objective was to use aerial photographs to map the fire and then coordinate the team on the ground. In the past few years, the literature has included more and more research of systems utilizing a team of small cooperating UAVs to get better surveillance; that is, better response time in missions where time is critical.

Recent studies have focused on special missions that can be efficiently performed with multi-UAV systems. Some address the problems of formation flight and some the problems of coordination. Fewer studies have been done on reconfiguring the coordination [81], or on coordination where the assigned tasks have uncertainty. This thesis demonstrates that if the guidance system accounts for real-time events and is able to adjust the flight formation to incorporate changes, then the trajectories are optimal in time.

Closely related is the work that has been done on multi-UAV coordination for tracking missions for search and rescue or surveillance [21]. Fig. 1.3 presents a concept that relies on low altitude and short endurance (UAVs). The work explores tracking a fire line by using a team of UAVs following the perimeter of

the wildfire area. The UAVs return periodically to the ground station location for downloading the collected data. The research focused on how to minimize the latency associated with the fire perimeter measurement when it is transferred to the ground station.

In [3], the design includes a coordination scheme to control a rotary wing platform (Quadrotor) for a similar mission to the one above. Essentially the motion of the UAV patrols the propagating perimeter. Whenever one UAV approaches another UAV (rendezvous point), deconflict the rendezvous and resolve each UAV next flight direction. This research assumes, however, that the perimeter of the fire is circular.



(a) Single UAV. (b) Two UAVs.

Figure 1.3: Conceptual diagram from [21]. (a) Latency profile for a single UAV monitoring a static circular fire. The thickness of the path denotes the latency of information at that point when it is transmitted to the base station. (b) Latency profile with a pair of UAVs monitoring a static circular fire in opposite directions.

An additional application [3] focuses on patrolling a fire front using quadrotors. They suggest that the path of the quadrotor can be updated by planning rendezvous points for sharing tracked data and then assigning the next segments of the perimeter. In a similar type of mission, [21] suggests a decentralized multi-UAV approach to monitor the perimeter of a fire. These studies (and similar ones), examine a specific scenario where the focus is on directly tracking the pe-

riphery point. This is limited however, by focusing on the connection between the uncertainty of the spreading perimeter and the maneuverability of the fleet needed to maintain knowledge of the complete perimeter.

Figure 1.4: UqH- (2 member) cooperation for forest fire surveillance from [3]. The goal of the cooperation is to track every point of the evolving fire perimeter and update the location of the fire with the minimum latency.

1.4.2 Observation

In coordination, one of the basic operations is observation sharing. Most of the recent studies in multi-UAV address the problem of partial information. It reflects the "real-world" problem where the UAV has limited communications (range or bandwidth). One UAV can communicate with one that is close by, but not with another that is far away. [78] presents a variety of research problems in which multi-vehicle systems agree on the value of observed data (consensus), and explores control strategies and a set of solutions for implementing them. [14] includes a chapter that suggests various deployment algorithms. They consider a distributed algorithm to address the physical limitations of the communication

system for observation sharing.

If a coordination algorithm for an environment with uncertainty is available, the overall system still relies on individual sensing capabilities. Even if the system uses the best or most advanced sensors, the sensors can be restricted by environmental conditions, i.e., the sensors carried by the UAV do not have suffecient range [30], and the data measured can only be local and quantized.

Quantization is a well-studied topic for a broad range of applications, most importantly overcoming the limited precision of electronics and computational processes. When sensors have insufficient precision, the situation can lead to ineffective decision-making; Quantized estimation was developed in order to reconstruct the data effectively [44].

The theorem of quantization [44] implies that the likelihood function can be reconstructed from quantized sensor observations through various methods [8]. Studies also show that sensor networks sharing quantized data can reach a consensus for estimating the state mean [12]. Multiple sensors, data sharing, and data fusing are examples of a class of applications that try to solve the reconstruction problem [8].

This thesis investigates data reconstruction with a new coordination scheme where missions are facing uncertainty about the periphery in the AOI. The coordination scheme takes into account the UAVs' state, their observations, and the overall mission, and allocates each UAV to a specific task. This coordination scheme allows the multi-UAV system to act in a coordinated manner. In this research, the coordination scheme is based on the assumption that there is no communication limit and therefore no need to visit the ground station often. The coordination algorithm considers the observations in the two phases of a traditional mission: the discovery phase and the tracking phase. Using the observed

information in both phases generates a better prediction of the periphery, reducing time lost and maximizing performance.

Current multi-UAV high-level-control systems have an inefficient use of resources (i.e., number of UAVs to AOI) for observations [73]. By wasting time and resources (e.g., hovering, loitering or long period of searching), the tracking mission performance is low, the system has a low-rate of significant detected data, and the current coordination algorithm makes the AOI less predictable.

The inefficiency of current systems with high level control creates significant timing difficulties for achieving the mission objectives. The ongoing mission can leave one vehicle loitering, resulting in a high latency of updates. Based on different studies [21], this represents a big time loss during a mission, with fewer updates, which in some cases can cause the mission to fail in its tracking objective.

1.4.3 Propagated Periphery Modeling

Disaster phenomena growth models, which predict the spatial and temporal dynamic spread rate, may help in evaluating the situation and deciding on a suitable response in a real-time deployment [7]. Appropriate representation and estimation of the spatial uncertainty can improve the prediction or help in developing a simplified model [89]. A mission with an uncertainty model for the AOI stands to benefit substantially from the predicted confidence envelope approach. For example, in expected high rate of spread (ROS) segments along the AOI perimeter, the allocation can use the availability and priority of the segment to get better results than if it were to assume that all segments along the perimeter are identical. Available UAVs can be redirected to new areas instead of merely

loitering.

In one of the biggest wildfire research projects done by the *Joint Fire Science Program*, the researchers developed fire behavior models for operational use. Their main objective was to develop a detailed dynamic model to predict the physical behavior of the ground phenomenon. They considered two simple fire modeling approaches. In both models, the assumption was that the local spread at a point on the perimeter is perpendicular to the fire perimeter into an unburned environment, and that the fire has a local rate of spread (ROS) normal to the fire line.

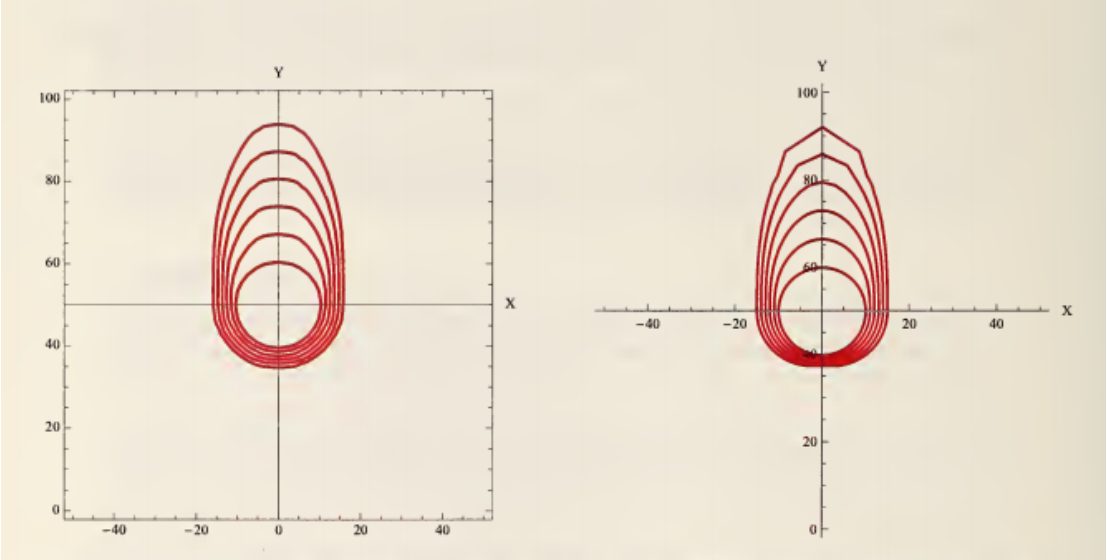
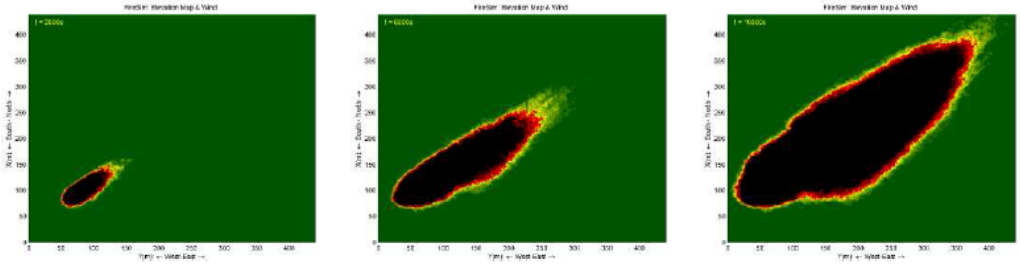


Figure 1.5: The progression of a wind-blown fire front from an initially circular fire. The model is formulated as a two-dimensional Eulerian partial differential equation (PDE) and solved with a level-set methodology (imported from [84]).)

There could be a variety of phenomena on the ground. This thesis deals with uncertainty from different sources, so that the detected phenomenon is represented as unknown dynamics, which is almost impossible to predict. However, the motion of the fire can be limited to reasonable boundaries with randomly changing motion. The future position of the measured object on the ground depends only on its

current position and not on its history; thus a simplified model is feasible.



(a) $t = 2800$sec (b) $t = 6800$sec (c) $t = 10800$sec

Figure 1.6: Model results from [21]. Fire simulation of high wind conditions with an elevation gradient. The fire is spreading in the direction of the wind. Ecological Model for Burning in the Yellowstone Region (EMBYR)

1.5 Contributions

This thesis contributes new results on mission-level control of multiple agents. The primary objective of the research is to minimize the uncertainty of a monitored closed boundary with dynamical changes in a large-scale area with stochastic environmental factors.

Much work has been done on patrolling missions of a stochastic propagated boundary [21]. Some researchers reported on utilizing variational methods for multi-agent guidance systems. This paper proposes a solution for deploying multiple agents for monitoring a propagating boundary in sub-optimal conditions. The results of this research introduce a new factor, Uncertainty Rate of Change (URoC), which can serve as a performance criterion for operational multi-agent systems. The URoC criterion considers the dynamical process model, number of agents, agent dynamics, and the agents' relative path orientation.

In particular, the contributions include the following developments:

- A methodology to estimate a propagated boundary with quantized measurements.
- A scheme for optimal deployment of a fleet of agents in an exploration mission.
- Methodology to dynamically assign a fleet of agents based on uncertainty distribution in a monitoring mission.
- Design and implementation of experimental demonstration for quick deployment of a low-cost, low-power fleet of UAVs with a high-level ground control system.
- Design and implementation of rapidly reconfigurable multi-UAV simulation (MSIL & MHIL).

1.6 Structure of the Dissertation

This research presents a multi-UAV coordination approach for reducing the uncertainty of a propagating periphery. Chapter 2 develops and presents our novel approach to periphery monitoring and compares its key performance characteristics with some existing methods. Chapter 3 develops the system architecture based on past projects. Chapter 4 describes the simulation as part of the system development process. Chapter 5 demonstrates a case study utilizing the complete developed algorithms, the implemented software, and the new MSIL simulation. Chapter 6 presents the conclusions of the research and suggests future work on multi-UAV high level control.

Chapter 2

Periphery Estimation Methods

2.1 Introduction

A necessary condition for monitoring a propagating boundary is precise estimation. That is, minimizing the deviation from the actual periphery in the presence of disturbances such as wind, 3D terrain features (such as slopes and vegetation), and sensor noise. Precise estimation is achieved through the development of a mathematical model that accurately responds to external observations so as to converge upon a predictive model [35]. The process of estimation is conditioned on having a mathematical model of what is being estimated (referred to as the dynamical phenomenon). This is because the estimator must be able to predict the boundary of the propagated phenomenon and adjust that prediction based on available sensor data. Estimation techniques are used to extract the statistical properties required to improve the model accuracy from the raw measurements gathered by the sensors.

2.1.1 Overview of Estimation Approaches

In dynamic system models there are often unknown or uncertain parameters that should be estimated along with the state itself. For example, in stochastic propagated phenomena, the spread rate of the boundary might be unknown. Similarly, the variances might only be known approximately, or not at all.

The estimation problem is not feasible in a situation where there are no available sensors. It is practical only where a dynamics model exists. Modeling the propagation of a ground phenomenon is complex because the model needs to include multiple dynamic and environmental factors [91]. Deriving an accurate model that considers all of those factors leads to an extremely complicated model. Usually, the solution is to utilize a dynamic model that is computationally feasible and is well-supported by observation.

The purpose of estimation is to extract the required information and often missing or unobserved measurements from the incoming measurement data. The estimator performs this task by using a cost function. That cost function can be an error over a period of time or more rigorous derivation of the system properties using probability tools.

Bayes' Rule

In a Bayesian setting, the proper way to estimate a parameter is by setting a prior distribution on the parameter $P(A)$ and treating it as an additional random variable in the model ([88]).
Bayes' theorem describes the probability of an event based on initial knowledge (also called the *prior*) and by accounting for evidence (the *measurement*) [63].

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)} \tag{2.1}$$

Equation (2.1) is the simple form of Bayes' rule where, according to the common interpretation, the event A has been observed and its impact on event B is being evaluated. $P(B|A)$ is the *posterior* probability and it is proportional to the *prior*, $P(B)$, times the conditional probability (or *likelihood*), $P(A|B)$.

The Bayesian interpretation for the case study in this thesis can be made by applying Bayes' rule. In equation (2.2), z describes the measurement of relative location of each agent in a binary representation, IN or OUT. The probability that the observation is in a bounded region A is $P(z \in A)$, and the conditional sampled event is, therefore:

$$P(B|z \in A) = \frac{P(z \in A|B)P(B)}{P(z \in A)} \tag{2.2}$$

Maximum Likelihood Estimation

The Gaussian distribution has two adjustable parameters: the mean value, μ, and the standard deviation σ. The question is: What choice of these parameters makes the observations "most likely"?

In principle, an optimal estimate of a state variable may be obtained from the probability density function. The maximum likelihood optimization method relies on the measured set of observations. Maximum Likelihood Estimation (MLE) can be utilized to estimate the state in any time-step when there is a new observation (Sequential method) or after collecting a number of observations (Batch method). The formal notation of the likelihood function is a conditional probability of the measurement z given a particular value of x:

$$\mathcal{L} \triangleq p(z|x) \tag{2.3}$$

Since the definition of the likelihood focuses on an observation and an associated *pdf*, it is a function of x. To find the maximum likelihood one should vary x until the probability is maximized:

$$\hat{x}_{MLE} = \arg\max_{x} p(z|x) \tag{2.4}$$

Maximum A-Posteriori Estimation

The Maximum A-Posteriori (MAP) estimator differs from the MLE estimator. The a-priori information can be regarded as an additional measurement. When there is prior knowledge of the probability distribution of x, the estimator incorporates that information in the estimation process to get better accuracy. In MLE the parameters are assumed to be unknown. The Bayesian inference used to determine MAP formally considers the parameters to be random variables. By using the Bayes rule to come up with the *posterior pdf* that incorporates the observation and the *prior* information, we get the following formal formula:

$$p(x|z) = \frac{p(z|x)p(x)}{p(z)} \tag{2.5}$$

where $p(x)$ is the prior distribution that denotes the prior beliefs of the parameter before we consider the observation, and $p(z)$ is a normalization term that is independent of the parameter and therefore is often left out of the estimator. To find the MAP estimator one should vary x until the probability $p(z|x)p(x)$ is maximized:

$$\hat{x}_{MAP} = \arg\max_{x} p(z|x)p(x) \tag{2.6}$$

Kalman Filter

The Kalman filter is regarded as the optimal linear solution to signal tracking and data estimation. The Kalman filter estimator is a practical use of some of the statistical techniques and is constructed as a mean squared error (MSE) minimizer. For minimization of the MSE, it employs a model for the Gaussian distribution errors. The process model is:

$$x_{k+1} = \Phi\ x_k + w \tag{2.7}$$

and the observation model,

$$z = H\ x + v \tag{2.8}$$

where x is the state, Φ is the state transition matrix, z is the measurement, H is the noiseless connection between the state vector and z, w and v are the uncorrelated associated process and observation noise accordingly. Both models of noise are assumed to be Gaussian:

$$x = N\{E(x|z), P^{(0)}\} \tag{2.9}$$

$$v = N(0, R) \tag{2.10}$$

where, R is the covariance, $P^{(0)}$ is the error covariance matrix and the state and the measurement noises are independent:

$$E(xv^T) = 0 \tag{2.11}$$

The estimator process follows a well known recursive method resulting in a minimum variance unbiased estimator (Linear Kalman Filter). The conditional mean

time propagation relation can be written as:

$$\bar{x} = \Phi \cdot x^{(-)} \tag{2.12}$$

$$P^{(-)} = \Phi \cdot P_{k-1}^{(+)} \cdot \Phi^T + Q \tag{2.13}$$

where Φ is the transition matrix, Q is the process noise covariance, $x^{(-)}$ is the prior state vector and $P_{k-1}^{(+)}$ is the process *a-priori* covariance. The conditional mean, (*a-posteriori*), of x is:

$$E(x|z) = \bar{x} + K(z - H\bar{x}) \tag{2.14}$$

where, H is the observation matrix and K is time-varying weighting matrix (the minimum variance gain) for Gaussian random variables:

$$K = P^{(-)}H^T(HP^{(-)}H^T + R)^{-1} \tag{2.15}$$

The conditional covariance of $\boldsymbol{x}$ is:

$$P_k^{(+)} = P^{(-)} - P^{(-)}H^T(HP^{(-)}H^T + R)^{-1}HP^{(-)} \tag{2.16}$$

and after substituting the variance gain, K:

$$P_k^{(+)} = [I - KH]P^{(-)} \tag{2.17}$$

Unscented Kalman Filter

The Unscented Kalman Filter (UKF) has become a standard technique used in many nonlinear applications [49]. It introduces improvements where the Extended

Kalman Filter (EKF) fails, and the estimator outcome diverges [95]. The original Kalman Filter approach proposes to propagate the Gaussian random variable through linear system dynamics, where the Gaussian random variable represents the state probability distribution. Furthermore, the EKF offers an approximation method when the system model is nonlinear, in which the Gaussian random variable is being propagated through a linearized version of state equation.

The first-order linearization of a nonlinear system can introduce significant errors and may cause reduced performance for state or parameters estimation [51] (i.e., mean and covariance). The UKF addresses the problems of state distribution approximation and uses pre-calculated sample points that capture the probability distribution of the approximated state [50].

Truncation

In engineering, many quantities that cannot be modeled accurately are assumed to follow a normal distribution. In some applications, however, there are physical reasons to set the probability to zero on specific bounds of the sampled data. If the normal distribution fits the data strongly, it may be preferable to work with a truncated normal distribution. The data is truncated in cases when it is known that the data can never be out of certain bounds. An example is if one wishes to consider data within a particular range of interest, sometimes noted as $[a, b]$, or $[a, +\infty)$, or $(-\infty, b)$, depending on the truncation applied.

It is possible to establish a truncated normal distribution by first assuming the existence of a "prior" normal distribution, with mean μ and standard deviation σ. One may then derive a modified distribution which is zero outside the region of interest, and inside the region has the same "bell shape" as the original normal distribution. The derived distribution must support the fact that the integral of

the new region should be 1, so it should be scaled by a constant [43]. Following the definition of conditional probability, the truncated normal distribution is:

$$f(x|x > a) = \frac{f(x)}{Prob(x > a)} \tag{2.18}$$

where, $f(x)$ is the *pdf* (probability density function) of a random variable x, and a is the lower bound of the range. The denominator is the quantity needed to scale the density such that it will integrate to 1.

Censoring

In some applications, the data is not fully accessible or has been masked. The measured quantities are, therefore, censored where there is not enough information and the different values of these quantities (in a specific region) are assumed to have the same value. The new distribution is a mixture of discrete and continuous parts. The original distribution is assigned to the uncensored region, and the full probability of the censored region is assigned to the censoring point. That way there is no need to scale up the uncensored part [43].
The expected value, $E[y]$, is then a mixture of a discrete censored part and a continuous part:

$$E[y] = Prob(y = a) \times E[y|y = a] + Prob(y > a) \times E[y|y > a] \tag{2.19}$$

The truncation and censoring methods assume that a normal distribution can represent the uncertainty of a given state variable. Each incorporated sample would update the distribution. The truncation approach scales up the distribution. The censoring keeps the scale at the last censoring point. Nevertheless, the resulted distribution in both is associated with the last updated region.

For the sequential process, when there is a new observation in each time step, implementing those methods becomes difficult. The observations being incorporated are from different drones with different distances (along with local compass directions), and they do not necessarily have common regions, nor monotonically decreasing distances.

Quantization

Digital components are frequently incorporated into most modern systems. Digital devices are limited to a certain number of quantum intervals in their outputs. The quantization of measurement is a process where a digital device outputs the proper interval according to where the measured quantity lies. The information can be recovered accurately if digital devices are either upgraded or improve their computational performance, such as by a sophisticated data processing algorithm.

In the research problem presented herein, there are only two quantum intervals which are considered as coarsely quantized measurements. The sensor measures the status of its current location accurately. The measurements represent the *inside of boundaries* or *outside of boundaries* status relative to the explored phenomenon, which is also equivalent to a quantized measurement. Moreover, where improving the sensor capabilities is not a feasible solution an alternate approach is to use the information, that a measurement on the boundary has not yet occurred, to improve knowledge of its current expected value.

In [27], the author describes methods to evaluate the mean and covariance of the measurement vector conditioned on the fact that the components have been quantized. The method is combined with the Kalman filter predictor-corrector approach. It determines the conditional mean and covariance of an estimated

state with quantized measurements. For convenience, Quantized Kalman Filter will be noted as QKF. The following section lays out an analytic derivation of the expectation for the state vector conditioned on quantized measurements. It concludes with the general equations that have been adapted to the periphery estimation problem under discussion.

2.1.2 Problem definition

The problem is one of optimization with respect to time with sparse measurements detected by a fleet of UAVs. The UAVs have a dynamic process to monitor, as quickly as possible, a periphery represented by a set of Control Points (CPs). The region of the phenomenon is in R^2, and the payload carried by each UAV fleet member includes an onboard sensor to distinguish between inside and outside areas (the quantized measurements). The observations are binary:

$$z \in A \tag{2.20}$$

where A is a bounded region (inside or outside).

Figure 2.1 illustrates the approach taken to represent the boundary with a set of CPs connected by straight lines. Each CP has a nominal spread rate that is considered relative to the origin point of the propagated phenomenon. That is, the spread rate is always pointed outward. The information is being gathered by a UAV to provide the observations that are noted as IN or OUT relative to the enclosed periphery. The optimum policy is derived from the decision of which CP the UAV should approach first.

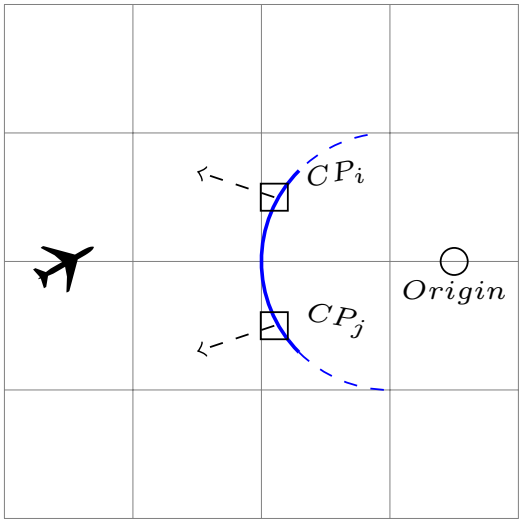


Figure 2.1: Setup of the boundary representation approach. CP_i and CP_j, (square), are two of many grid points representing the close predicted periphery (in blue). The Origin (circle) is the starting point of the propagated phenomenon, and the UAV is used to collect observations.

The estimation problem accounts for the dynamics properties of the problem, where vehicles are mobile and the environment domain changes.

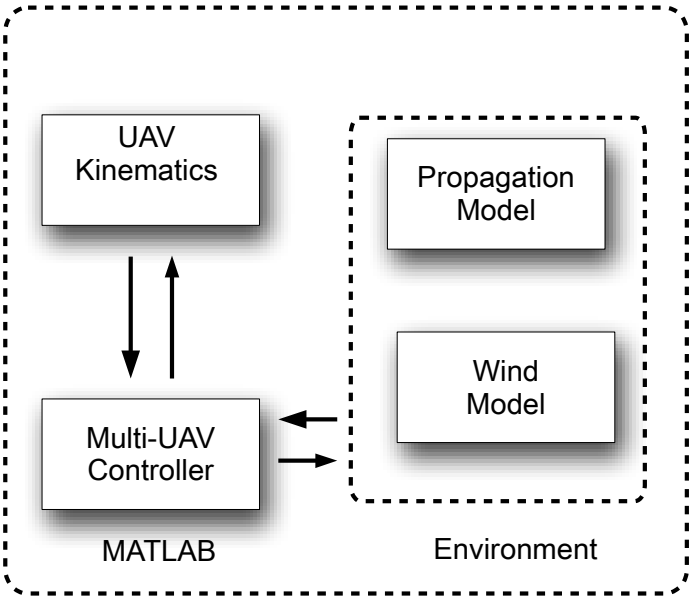


Figure 2.2: The block diagram describes the major components involved in the estimation problem.

Figure 2.2 illustrates the the major components involved in the estimation process. The propagation model accounts for the dominant factors of a fire line (i.e., wind and terrain slope).

The model used for the UAV assumes that the speed (v) and altitude are constants, and the UAV is limited to a minimum turn radius of R:

$$\begin{aligned} \dot{x} &= vcos(\theta) \\ \dot{y} &= vsin(\theta) \\ \dot{\theta} &= \omega \end{aligned} \tag{2.21}$$

where the angular velocity ω is bounded, ($\omega \leq |v/R|$), and x, y are the position

of the UAV.

The primary goal of the research is to design a method that achieves the minimal cumulative uncertainty of the periphery. The method should persistently monitor the uncertainty over all CPs and over a long span of time. The problem can be viewed as an evaluation of a ground phenomenon where the state periphery is propagated and sparse measurements are detected by a fleet of UAVs. The particular surface of the phenomenon is a large-scale area, and the payload on each UAV fleet member includes an onboard sensor to distinguish between inside and outside areas (quantized measurements).

The problem involves three research areas: i) representation of the continuous phenomenon, ii) model-based prediction of a propagated periphery, and iii) estimation of the front based on quantized measurements. The primary goal of the research is to design a method that achieves the minimal uncertainty of the front based on the quantized measurements, such that in the following stage of the mission the guidance logic updates the fleet members' tasks and reroutes the fleet members to further suppress the uncertainty.

2.2 QKF Method

This section presents a method to estimate the uncertainty along the periphery of a stochastic phenomenon. The technique analytically computes the frontier that is conditioned on UAV measurements and uses the Quantized Estimation method [27] to update the expected location and spread-rate of a propagated boundary. The methodology utilizes a Quantized Estimation to improve the predicted spread of a periphery and decrease the uncertainty of its current state. It follows the procedure to construct the boundary and estimate the rate of spread. After the Quantized Estimator is determined, the algorithm is incorporated in a UAV dynamic simulation to prove the concept of uncertainty suppression with various deployment schemes for a single UAV. The section concludes with a detailed numerical analysis of the derived algorithm and extensive simulation results.

2.2.1 Boundary Definition

In dynamic systems models, there are often unknown or uncertain parameters that should be estimated along with the state of dynamics. For example, in stochastic propagated phenomena, the spread rate of the boundary might be unknown, and the variances of the spread-rate could be known approximately or not known at all.

Figure 2.3 illustrates the approach taken to represent the boundary with a set of control points (CPs) connected by straight lines. Each CP has a nominal spread rate that is relative to the origin (O) of the propagated phenomenon. That is, the observed spread rate is always pointed outward. The information is being gathered by a UAV to provide the observations that are noted as IN or OUT relative to the enclosed periphery.

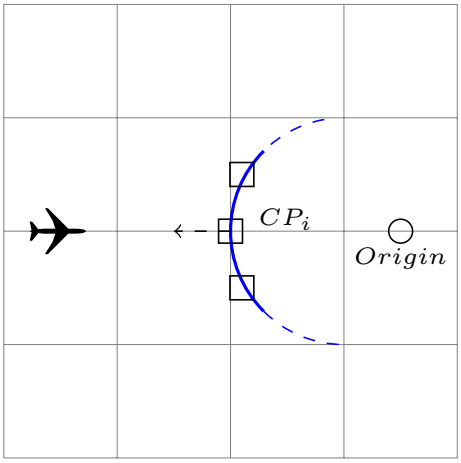


Figure 2.3: Setup of the boundary representation approach. CP_i, (square), is one of many points representing the close predicted periphery (in blue). The Origin, (circle), is the starting point of the propagated phenomenon and the UAV is used to collect observations.

2.2.2 QKF Estimator

In many estimation methods the principle solution is to obtain the *pdf* ($p(x_k|z_{1:k})$), where x_k is the state vector at time k, $z_{1:k}$ are all the measurements from initial time step to current time step (k) and $z_k = Hx_k + v_k$ is measurement at time k. The standard implementation is a two-step procedure: prediction and update. Assuming that the *pdf* from the last time-step $(k-1)$ is available the probability of the state vector (x) conditioned on measurements (z) is, $p(x_{k-1}|z_{1:k-1})$, the system model $x_k = f_k(x_{k-1}, v_{k-1})$ (where, v is the process noise) is used to obtain the predicted *pdf* $p(x_k|z_{1:k-1})$ in the prediction stage.

The update stage utilizes the measurement z_k to modify the prior *pdf* to obtain the posterior *pdf* $p(x_k|z_{1:k})$ of the current state x_k.

In [27] the author includes a full derivation of the expectation of state vector conditioned on quantized measurements, then concludes with the following equations:

$$E[f(x)|z \in A] = E\{E[f(x)|z]|z \in A\} \tag{2.22}$$

where the mean of some function of x, $f(x)$, conditioned on measurement z which is bounded in some region A, can be evaluated by finding the conditional mean of $f(x)$ given a measurement and then averaging this function, $f(z) = E[f(x)|z]$, conditioned on the bounded region ($z \in A$). These results are useful for quantized random variables and can be used to formulate the estimation problem with a proper interpretation of the type of dynamic system that is the focus of this research. The method that has been adopted and further developed in this research is based on the following [27]:

The conditional mean of the state,

$$E(x|z \in A) = \bar{x} + K(E(z|z \in A) - H\bar{x}) \tag{2.23}$$

and the conditional covariance of the state,

$$cov(x|z \in A) = P^{(+)} + Kcov(z|z \in A)K^T \tag{2.24}$$

where K is a time-varying weighting matrix (the minimum variance gain) for Gaussian random variables, $\bar{x}$ is the predicted state, $P^{(+)}$ is a-posteriori covariance, A ($A \subset R^2$) and $cov(z|z \in A)$ are the bounded region and the conditional covariance of the measurements, respectively.

The resulting formulas include two new terms, the mean and covariance of the measurement conditioned on the fact that the measurement vector has been quantized. Although the quantized region A can be more than one dimensional, the approximation methods are usually impractical. Tractable formulas, however, do exist for one dimension.

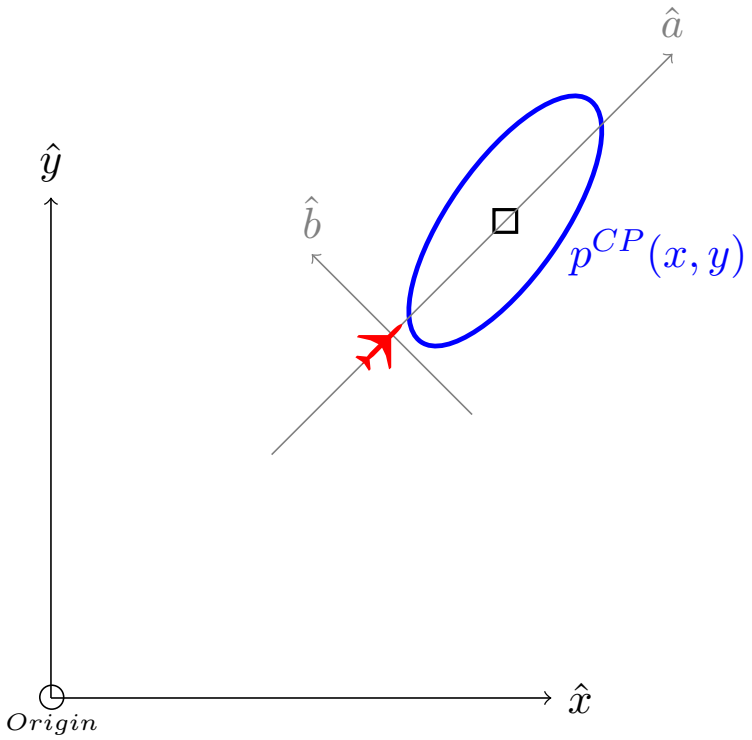


Figure 2.4: Basic coordinate-systems. The quantized measurement connected along 1-D coordinate line ($\hat{a}$) and the covariance of the prediction represented with an ellipse.

2.2.3 QKF - Implementation

The literature on this topic shows that there is no existing general analytic solution for the conditional mean and conditional covariance terms in equations 2.23 and 2.24. However, in the scalar case, there is a known relationship ([27]) if the a-priori measurement, z, is assumed to be Gaussian and lies in the region $r \leqslant z \leqslant s$ [27]:

$$E(z|z \in A) = \bar{z} + \frac{\sigma_z}{P(z \in A)} \times \frac{1}{\sqrt{2\pi}} \left(e^{-\frac{(r-\bar{z})^2}{2\sigma_z^2}} - e^{-\frac{(s-\bar{z})^2}{2\sigma_z^2}} \right) \tag{2.25}$$

$$\begin{aligned} cov(z|z \in A) = \sigma_z^2 [1 &+ (\frac{r-\bar{z}}{\sigma_z}) \frac{exp\{-\frac{1}{2}[r-\bar{z}/\sigma_z]^2\}}{(2\pi)^{1/2} P(z \in A)} \\ &- (\frac{s-\bar{z}}{\sigma_z}) \frac{exp\{-\frac{1}{2}[s-\bar{z}/\sigma_z]^2\}}{(2\pi)^{1/2} P(z \in A)} \\ &- \frac{\sigma_z}{P(z \in A)} \times (\frac{exp\{-\frac{1}{2}[r-\bar{z}/\sigma_z]^2\}}{(2\pi)^{1/2}} \\ &- \frac{exp\{-\frac{1}{2}[r-\bar{z}/\sigma_z]^2\}}{(2\pi)^{1/2}})] \end{aligned} \tag{2.26}$$

where

$$P(z \in A) = \frac{1}{2} \left[erf\left(\frac{r-\bar{z}}{\sqrt{2}\sigma_z}\right) - erf\left(\frac{s-\bar{z}}{\sqrt{2}\sigma_z}\right) \right] \tag{2.27}$$

where $\bar{z}$, σ_z are the mean and variance of the one dimensional measurement respectively, the quantities r and s define the region A. The scalar case yields the result for a one-dimensional system representation (see Fig. 2.4) and for that rea-

son, captures the distribution along a line that connects the measurement location with the predicted boundary (control points).

There is a complementary sequence of steps to set the approximated boundaries in the scalar closed form solution. It is first and foremost based on the UAV position and the estimated boundary. To compare the estimate of the IN or OUT of the periphery to the measurement, a relative coordinate system needs to be determined. That coordinate system should be oriented such that the measurement can directly influence the estimate and the measurement can be expected based on the prediction.

The virtual line being used to evaluate the region is always directed from the observation toward the CP. The bearing of that line is used to compute the transformation matrix (T) and to transform the covariance (Eq. 2.28), and derive the one-dimensional variance (σ_z). It then translates the origin of the coordinate system to the location of the observation. The mean value ($\hat{\mu_a}$) can then be evaluated based on probability reasoning for the region: $[0, \infty]$ if the measurement is as expected or $[-\infty, 0]$ if it differs from the expectation. The covariance of the predicted point is projected by using a transformation matrix to the line-of-sight coordinate system $[a, b]$:

$$P_{a,b} = \begin{bmatrix} \sigma_{aa} & \sigma_{ab} \\ \sigma_{ab} & \sigma_{bb} \end{bmatrix} = T \cdot P_{x,y} \cdot T^T \tag{2.28}$$

The evaluated variance associated with the explored CP is the same variance as at the measurement location (along the line-of-sight). Hence, the projected variance (σ_z) can be used as part of the evaluation step of the conditional covariance of the scalar measurement (in Eq. 2.26).

Figure 2.4 shows the basic transformation that the algorithm employs to project

the covariance and find the current variance for the current observation.

Algorithm 1 Estimation with QKF.

Given a CP state vector x_i^{CP} and observations y_i
repeat
1. *Predict.* $x_{x,y}^{CP(+)} := \Phi \cdot x_{x,y}^{CP(-)}$
2. *Transformation.* $T = T_{x,y}^{a,b}$.
3. *Translation.* $x_a^{CP} := Dist(x_{x,y}^{CP}, x_{x,y}^{DP})$.
4. *Covariance Evaluation.* $P_{a,b}^{CP} := T \cdot P_{x,y}^{CP} \cdot T^T$.
5. *1D Evaluation.* $\mu_a := x_a^{CP}, \sigma_a := P_{a,b}^{CP}(1,1)$.
6. *Set Bounds.* $(\lfloor a \rfloor, \lceil b \rceil)$.
7. *QKF (scalar evaluation).* $E(z|z \in A)$, $cov(z|z \in A)$.
8. *QKF Update.* $x_a^{CP} := E(x|z \in A), P_{a,b}^{CP} = cov(x|z \in A)$.
9. *Inverse Transformation.* $P_{x,y}^{CP} := T^T \cdot P_{a,b}^{CP} \cdot T$.
10. *Inverse Translation.* $\hat{x}_{x,y}^{CP} := x_{x,y}^{CP(+)} + T^T \cdot (\hat{\mu_a} - x_{x,y}^{CP(+)})$

until stopping criterion is satisfied.

Algorithm 1 presented here describes a step-by-step procedure to determine the estimated state of each CP. The superscript $^{(+)}$ describes the *a-posteriori* value, and $^{(-)}$ describes the *prior* value. Also, Φ is the transition function, representing the dynamic model. The first four steps are similar to the classic Kalman Filter. The following steps are part of the newly developed method. Step 5 and 6 are the core of the quantization interpretation. To set the boundaries of the interval, the method evaluates different configurations that the system would interpret in a nominal scenario. Before incorporating the measurement, it estimates what the expected state (IN/OUT) should be for the UAV's current location. The observations are either IN or OUT (relative to the periphery). If the observation is as the expected measurement, the interval is set to $[0, \infty]$. If there is a conflict between the observation and the expected measurement, the heuristic interpretation is that the predicted spreading rate is either slower than it should be or too fast. In the case where there is a conflict, the estimator sets the interval for evaluation to $[-\infty, 0]$, meaning that the predicted mean value will likely be behind the location

where the observation has been taken.

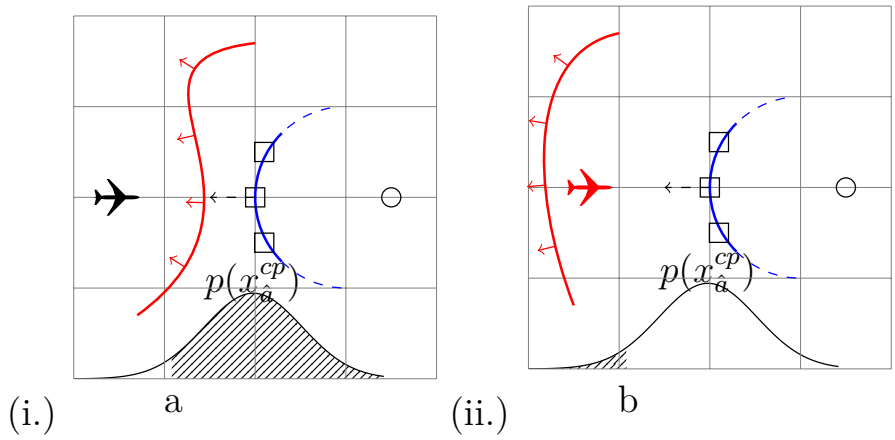


Figure 2.5: The two possible configurations for measurement. The plot to the left is for the case where observation is as expected. The plot to the right is the case in which the observation conflicted with expected measurement. The airplane in black means that the observation is labeled OUT. The airplane in red is for IN labeled observation. The control point (square) is within expected spreading direction. The fire origin is represented as a circle, and the actual propagated segment is in red with arrows toward the spread direction. Notice that the probability density function attached to each configuration consist interpreted boundaries (shaded in the graph) that should be considered in the estimation step.

Figure 2.5 presents the two possible configurations the measurements can be in (confirm or disagree with prediction). The integration limits are determine based on the interpretation of the current configuration: 'a' is lower integration limit for non-conflicted, 'b' is upper integration limit for conflicted observation.

2.2.4 Approximations Techniques

In the following sections, the closed form solution of the conditioned mean and variance is replaced with approximations. The new techniques combine QKF approach with classical estimators. It then modifies the update step to include approximated mean and covariance of the quantized measurements and examine its robustness.

MLE Approximation

This section addresses the general approach for the maximum likelihood (ML) with quantized measurements. The MLE estimator determines the most probable parameters $H_k x_k$. Consider the linear measurement equation:

$$z_k = H_k x_k + v_k \tag{2.29}$$

For convenience, the estimated measurement is denoted by $z_k^* \triangleq H_k x_k$, where the state vector x_k varies over time (subscript k) according to a known dynamic model, z_k is the measurement vector before quantization, and H is the linear observation model. Given an observation $z \in A$, the ML estimator finds the value of z^* that maximizes the likelihood

$$P(z \in A | z^*) \tag{2.30}$$

or,

$$z_{MLE}^* = \arg\max_{z^*} P(z \in A | z^*) \tag{2.31}$$

Ref. [27] describes the MLE estimates of z in Eqs. 2.4 and 2.5. The equations derivation assumes that the measurements are bounded ($[a, b]$) and that the noise components v^i are independent:

$$\begin{aligned} L(a, b, x) &= P(a \leq z < b) \\ &= P(a - H_k x_k \leq v < b - H_k x_k) \\ &= \prod_i P[a^i - (H_k x_k)^i \leq v^i < b^i - (H_x)^i] \end{aligned} \tag{2.32}$$

The probability for each quantized observation is given by integrating the noise

density function, $p_v(u)$:

$$P = P[a - z_k^* \leq v < b - z_k^*] = \int_{a-z_k^*}^{b-z_k^*} p_v(u)du \tag{2.33}$$

The integral is the probability, which is the area under the *pdf* curve. The likelihood is maximized for wider bounds. Assuming that the probability density function of the measurement noise is known, the MLE is that value of x_k that maximizes the likelihood of the measurement z_k.

To find the optimal parameters, the estimator sets arbitrary bounds based on statistical reasoning associated with the relative location of the measurement. Assuming that x can take any value and that $[a, b]$ are arbitrary bounds, there are two possible pairs of bounds, which leads to the integral lower and upper bounds. In both cases, the maximum likelihood obtained is as high as x goes, hence $P_i \approx 1$. A practical or engineering perspective, however, restricts the solution to a Gaussian random variable that is feasible and properly limited. Hence, referring to Fig. 2.5 the approximated state statistical properties used in the simulation for the MLE case are:

$$z_{MLE}^* = a + 2\sigma_v \tag{2.34}$$

and for the conflicted case:

$$z_{MLE}^* = b - 2\sigma_v \tag{2.35}$$

where a and b are the lower and upper integration limits for non-conflicted and conflicted measurement, (see Fig. 2.5).

Covariance Approximation The approximation for the conditional covariance, cov_{MLE}, is developed by analyzing the expected error and deriving the analytical formula. The detailed derivation is described for the MLE approxima-

tion; however it is also applied to the next two approximation methods (MAP and UKF).

$$cov^*_{MLE}(z|z \in A) = E\{(z - z^*)\}^2 \tag{2.36}$$

The covariance integral is by definition an integral of the product of squared errors and the PDF of the prior ($p(z)$):

$$cov^*_{MLE}(z|z \in A) = \int_a^\infty (z - z^*)^2 \cdot \frac{p(z)}{P(z \in A)} dz \tag{2.37}$$

The presented case in Eq. 2.37 is for the non-conflicted observation where a is the lower bound and the upper bound is ∞. The term $P(z \in A)$ is used for scaling the density so that it integrates to one over the range of A and is described in Eq. 2.27. By expanding the equation to three integrals :

$$\begin{aligned} cov^*_{MLE}(z|z \in A) = & \int_a^\infty z^2 \cdot \frac{p(z)}{P(z \in A)} dz \\ & - 2z^* \int_a^\infty z \cdot \frac{p(z)}{P(z \in A)} dz \\ & + z^{*2} \cdot \int_a^\infty \frac{p(z)}{P(z \in A)} dz \end{aligned} \tag{2.38}$$

Note that the integral of the second term is the conditional mean $E(z|z \in A)$ and that the integral in the third term is by definition one. By using basic exponential integral solutions:

$$\int z \cdot e^{-c \cdot z^2} dz = -\frac{1}{2c} e^{-c \cdot z^2} \tag{2.39}$$

$$\int z^2 \cdot e^{-c \cdot z^2} dz = \frac{1}{4} \sqrt{\frac{\pi}{c^3}} erf(z\sqrt{c}) - \frac{z}{2c} e^{-c \cdot z^2} \tag{2.40}$$

where, the variable of integration is z and c is a constant variable. Utilizing the formulas and substituting the limits of the first two terms in Eq. 2.38 is given by:

$$\begin{aligned} cov^*_{MLE} &= \frac{1}{P(z \in A)} \cdot d \cdot \left[\int_a^\infty z^2 \cdot e^{-\frac{z^2}{2\sigma_z^2}} dz - 2z^* \int_a^\infty z \cdot e^{-\frac{z^2}{2\sigma_z^2}} dz \right] + z^{*2} \\ &= \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4}\sqrt{\frac{\pi}{c^3}} \cdot (1 - erf(a\sqrt{c})) + \frac{a}{2c} e^{-c \cdot a^2} - z^*_{MLE} \cdot \frac{1}{c} e^{-ca^2} \right] + {z^*_{MLE}}^2 \end{aligned} \tag{2.41}$$

where, c, d are constant variables which represent the scaling factors for the normal *pdf*: $c \equiv \frac{1}{2\sigma_z^2}$ and $d \equiv \frac{1}{\sqrt{2\pi}\sigma_z}$.

In the conflicted case, where the expected measurement is not as been observed, the truncation is at the upper limit b, and the resulting covariance is:

$$cov^*_{MLE} = \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4}\sqrt{\frac{\pi}{c^3}} \cdot (1 + erf(b\sqrt{c})) - \frac{b}{2c} e^{-c \cdot b^2} + z^*_{MLE} \cdot \frac{1}{c} e^{-cb^2} \right] + {z^*_{MLE}}^2 \tag{2.42}$$

MAP Approximation

The theoretical formulas (2.4,2.6) can be hard to apply in real-time situations. Applying the formulas directly would require infinite samples and result in a high computational load.

The key in a real-time estimator implementation is to use a recursive estimator to incrementally update the posterior probability distribution of the state vector based on the most recent data. MAP estimation fuses both the a priori observation and the new one to come up with an estimate. It repeats that process by using the previous estimate as a priori and incorporating it with a fresh observation. Assuming the observations produced by the sensor sequentially, it would compute the estimate values in each time step.

One can find a closed form solution for the general MAP equations, but it cannot be employed here since the observations are bounded. It is discussed before that the measurements and the parameters be related through the linear measurement equation:

$$z = Hx + v = z^* + v \tag{2.43}$$

Where v is the measurement noise and z^* is the transformed state. MAP estimator finds the best measurement $z^* = Hx$ to use in place of $E(z|z \in A)$. The joint probability $P(z^*, z \in A)$, can be expanded by using the the chain-rule of conditional probabilities

$$P(z^*, z \in A) = P(z^*|z \in A) \cdot P(z \in A) = P(z \in A|z^*) \cdot P(z^*) \tag{2.44}$$

The probability is conditioned on measurement z, which is bounded on the region A ($[a, b]$). Following the general description in Eq. 2.44, the posterior probability is a product of the conditional probability and the prior:

$$P(z^*|z \in A) = \frac{P(z \in A|z^*) \cdot P(z^*)}{P(z \in A)} \tag{2.45}$$

The denominator is the normalizing factor from the formal Bayes rule which keeps the a-posteriori distribution normalized. It fuses both the a-priori observation and the new one to come up with an estimate:

$$z^*_{MAP} = \arg\max_{z^*} P(z \in A|z^*)p(z^*) \tag{2.46}$$

It is assumed that the measurements are Gaussian distributed:

$$p(z^*) \sim \mathcal{N}(H\bar{x},\ HP^{(-)}H^T) \tag{2.47}$$

where $E\{x\} = \bar{x}$, $cov\{x\} = P^{(-)}$ and for a special case of Eq. 2.47 the mean is zero and the variance is, $\sigma^2_{z^*}$,

$$p(z^*) \sim \mathcal{N}(0, \sigma_{z^*}) \tag{2.48}$$

The first case to consider is the non-conflicted case, where the expected measurement is the same as the observation. The lower bound is represented by the parameter a. For all scenarios where the measurement is bigger than the left side bound of the distribution it can be reformulated as follows:

$$p(z^*|z > a) \sim P(z > a|z^*)p(z^*) \tag{2.49}$$

The denominator term of Eq. 2.45 can be ignored. That term is a constant multiply function that does not change with z^*. Therefore, when computing the maximum a-posterior of the mean, the best z^* can be found by searching for the maximal probability:

$$z^*_{MAP} = \arg\max_{z^*} \int_{a-z^*}^{\infty} p_v(u)du \cdot e^{-\frac{1}{2}\frac{z^{*2}}{\sigma^2_{z^*}}} \tag{2.50}$$

where, $p_v(u)$ is the *pdf* of the measurement noise. The integral is re-scaled and reformulated to be in σ_z units. By using the substitution $\nu = \frac{u}{\sigma_z}$, convert $d\nu = \frac{du}{\sigma_z}$ and convert the limits, this transforms the integral term:

$$z^*_{MAP} = \arg\max_{z^*} \int_{lb}^{\infty} e^{-\frac{1}{2}\frac{\sigma^2_{z^*}\nu^2}{\sigma^2_v}} d\nu \cdot e^{-\frac{1}{2}\frac{z^{*2}}{\sigma^2_{z^*}}} \tag{2.51}$$

where, $lb \equiv \frac{a - z^*}{\sigma_z}$ is the integral lower bound for the non-conflicted case. By using the basic integrals with exponentials solution:

$$\int e^{-cx^2} dx = \frac{\sqrt{\pi}}{2\sqrt{c}} erf(x\sqrt{c}) \tag{2.52}$$

$$z^*_{MAP} = \arg\max_{z^*} [1 - erf(\frac{(a - z^*)}{\sqrt{2}\sigma_v})] \cdot e^{-\frac{z^{*2}}{2\sigma^2_{z^*}}} \tag{2.53}$$

A similar process may be derive for the MAP when the observation is conflicted with the expected measurement. The upper bound, b, appears in the equation and the lower limit of the integral is now $-\infty$. The derived formula for the conflicted case is then:

$$z^*_{MAP} = \arg\max_{z^*} [erf(\frac{(b - z^*)}{\sqrt{2}\sigma_v}) + 1] \cdot e^{-\frac{z^{*2}}{2\sigma^2_{z^*}}} \tag{2.54}$$

The approximation for the conditional covariance, $cov^*_{MAP}(z|z \in A)$, is treated in the same way as the derivation of the approximated covariance for the MLE.

$$cov^*_{MAP} = \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4}\sqrt{\frac{\pi}{c^3}} \cdot (1 - erf(a\sqrt{c})) + \frac{a}{2c} e^{-c \cdot a^2} - z^*_{MAP} \cdot \frac{1}{c} e^{-ca^2}\right] + {z^*_{MAP}}^2 \tag{2.55}$$

where, c, d are constant variables which represent the scaling factors for the normal *pdf*: $c \equiv \frac{1}{2\sigma_z^2}$ and $d \equiv \frac{1}{\sqrt{2\pi}\sigma_z}$.

In the conflicted case, where expected is not as been observed, thus the truncation is at the upper limit, b, and the covariance is:

$$cov^*_{MAP} = \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4}\sqrt{\frac{\pi}{c^3}} \cdot (1 + erf(b\sqrt{c})) - \frac{b}{2c} e^{-c \cdot b^2} + z^*_{MAP} \cdot \frac{1}{c} e^{-cb^2}\right] + {z^*_{MAP}}^2 \tag{2.56}$$

The major different of algorithm 2 compared with algorithm 1 lies in the fact that it utilizes the MAP approximation step. Eqs. 2.53 and 2.53 are employed in step 7 of the following detailed approximation method:

Algorithm 2 Estimation with MAP Approximation.

Given a CP state vector x_i^{CP} and observations y_i

repeat

1. *Predict.* $x_{x,y}^{CP(+)} := \Phi \cdot x_{x,y}^{CP(-)}$
2. *Transformation.* $T = T_{x,y}^{a,b}$.
3. *Translation.* $x_a^{CP} := Dist(x_{x,y}^{CP}, x_{x,y}^{DP})$.
4. *Covariance Evaluation.* $P_{a,b}^{CP} := T \cdot P_{x,y}^{CP} \cdot T^T$.
5. *1D Evaluation.* $\mu_a := x_a^{CP}, \sigma_a := P_{a,b}^{CP}(1,1)$.
6. *Set search Bounds.* $(\lfloor a \rfloor, \lceil b \rceil)$.
7. *MAP (scalar approximation).* z_{MAP}^*, cov_{MAP}^*.
8. *Update (QKF).* $x_a^{CP} := E(x|z \in A), P_{a,b}^{CP} = cov(x|z \in A)$.
9. *Inverse Transformation.* $P_{x,y}^{CP} := T^T \cdot P_{a,b}^{CP} \cdot T$.
10. *Inverse Translation.* $\hat{x}_{x,y}^{CP} := x_{x,y}^{CP(+)} + T^T \cdot (\mu_a - \hat{x}_{a,b})$

until stopping criterion is satisfied.

The first steps in the algorithm are the prediction and the transformation from the general coordinate system to the one-dimensional representation. The last three steps include the QKF update and the inverse-transformation.

UKF Approximation

The UKF approximation process involves determining a nonlinear mapping of a given observation that is parametrized by mean and variance variables. The nonlinear mapping is used to interpret the statistical properties of the unobserved state from the observed one. The goal involves solving for the parameters in order to minimize the expected square error, $e_k = \hat{z}_k - h(x_k)$, where $\hat{z}_k$ is the predicted measurement and $h(x_k)$ is the nonlinear observation transformation. Hence, the parameter estimation corresponds to estimating the Gaussian statistical proper-

ties, for example, μ_z and σ_z.

The complete derivation of the UKF estimator is described in [50]. This research adapts the UKF method to approximate nonlinear mapping functions (see A.1.9). To calculate the statistics of the nonlinear measurement z_k, the method determines a matrix Y that includes $2L + 1$ candidate points, where L is the dimension of the measurement z_k. The candidate points are also known as the sigma points, and each one of them is a scalar (ith entry) in the matrix $Y^{(i)}$. The sigma points are sent through the observation heuristic:

$$Y^{(i)}_{k|k-1} = z^{(i)}_{k|k-1}, \quad i = 1, ..., 2L \tag{2.57}$$

Note, the subscript $k|k-1$ means that this is the predicted value based on the information from the last time step.

The observation heuristic adapts the UKF technique. It is adjusted and extended based on the main idea of considering the a, b limits. The sigma points are chosen with the prior variance, however since the sigma points are statistical distribution representation of the bounded region they should be shifted to the updated region when measurements are being incorporated. The sigma points are set to be in-between the limits, for the conflicted case or the non-conflicted case. A sigma point is reevaluated and replaced with the upper or lower limit when it is out of the bounded region.

$$z^*_{UKF} = \sum W^m_i Y^{(i)}_{k|k-1} \tag{2.58}$$

where W^m_i are the sigma points corresponding weights.

The approximation for the conditional covariance, $cov^*_{UKF}(z|z \in A)$, is treated in the same way as the derivation of the approximated covariance for the MLE or

MAP.

$$cov^*_{UKF} = \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4} \sqrt{\frac{\pi}{c^3}} \cdot (1 - erf(a\sqrt{c})) + \frac{a}{2c} e^{-c \cdot a^2} - z^*_{UKF} \cdot \frac{1}{c} e^{-ca^2} \right] + {z^*_{UKF}}^2 \tag{2.59}$$

In the conflicted case, where expected is not as been observed, the truncation is on the upper limit b, and the covariance is:

$$cov^*_{UKF} = \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4} \sqrt{\frac{\pi}{c^3}} \cdot (1 + erf(b\sqrt{c})) - \frac{b}{2c} e^{-c \cdot b^2} + z^*_{UKF} \cdot \frac{1}{c} e^{-cb^2} \right] + {z^*_{UKF}}^2 \tag{2.60}$$

The final step in the UKF approximation is the update step. This involves calculating the estimated output based on the estimated covariance (cov^*_{UKF}). By applying the UKF approximation on the mean of the state conditioned with quantized measurements:

$$\hat{x}_{UKF}(x|z \in A) = \bar{x} + K(z^*_{UKF}(z|z \in A) - H\bar{x}) \tag{2.61}$$

where, $z^*_{UKF}(z|z \in A)$ is the mean value of the selected sigma-points, and the approximated conditioned state covariance is:

$$cov_{UKF}(x|z \in A) = P^{(+)} + K cov^*_{UKF}(z|z \in A) K^T \tag{2.62}$$

where, $cov^*_{UKF}(z|z \in A)$ evaluated by substituting the approximated mean value, $z^*_{UKF}(z|z \in A)$, in the general formulation of approximated covariance and $P^{(+)}$ and K are the state covariance and Kalman gain from QKF estimator.

Approximation Techniques Comparison

The previous section formulate the mean and covariance of the measurement conditioned on the fact that the measurement vector has been quantized: $E(z|z \in A)$, $cov(z|z \in A)$. Different known estimation methods were adopted to estimate the state conditioned on quantized measurement.

This section compares the estimators that come from a single measurement with quantization limits as shown in Fig. 2.5

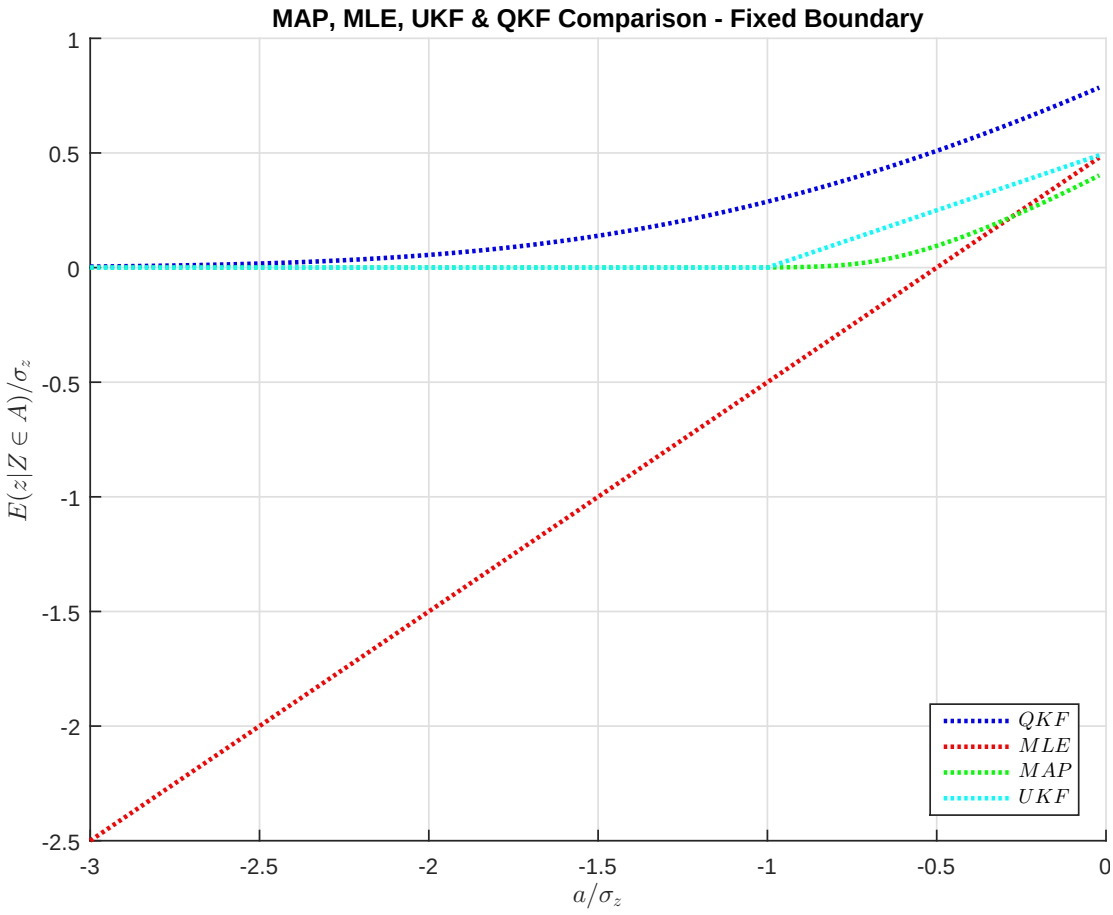


Figure 2.6: A comparison between the approximations methods and the QKF. The estimated mean of the quantized measurement is evaluated with distance for the non-conflicted case.

Fig. 2.6 compares the approximation methods to the QKF estimator. The

expectation is fixed for the presented sequence ($\mu = 0$), the variance remains the same for each distance evaluation, and the observations are sampled in the range of three standard deviations.

Fig. 2.7 illustrates the effectiveness of the different variants and how the statistical properties have been influenced by them. The observations impact the expectation more as the measurements get closer to the current predicted mean value. The MLE, MAP and QKF estimation methods incorporate the observations by truncating the distribution. The truncated part is relative to the distance from the mean value and the current known distribution (i.e., the variance). QKF introduces bigger displacements to the expectation value at a much closer distance, and the MAP estimator is bounded by the MLE which sets the expectation by the previous determine heuristic rule (Eq. 2.34).

Assuming that the observation is reliable, the expectation update is significant. MLE uses the truncation, but it is bounded by the size of the standard deviation of the observation noise. MAP relies on MLE and considers the prior which improves its effectiveness. UKF implementation is limited to two sigma points only. The attempt to describe the distribution with only two sigma-points induces less influence and worse performance than the other techniques.

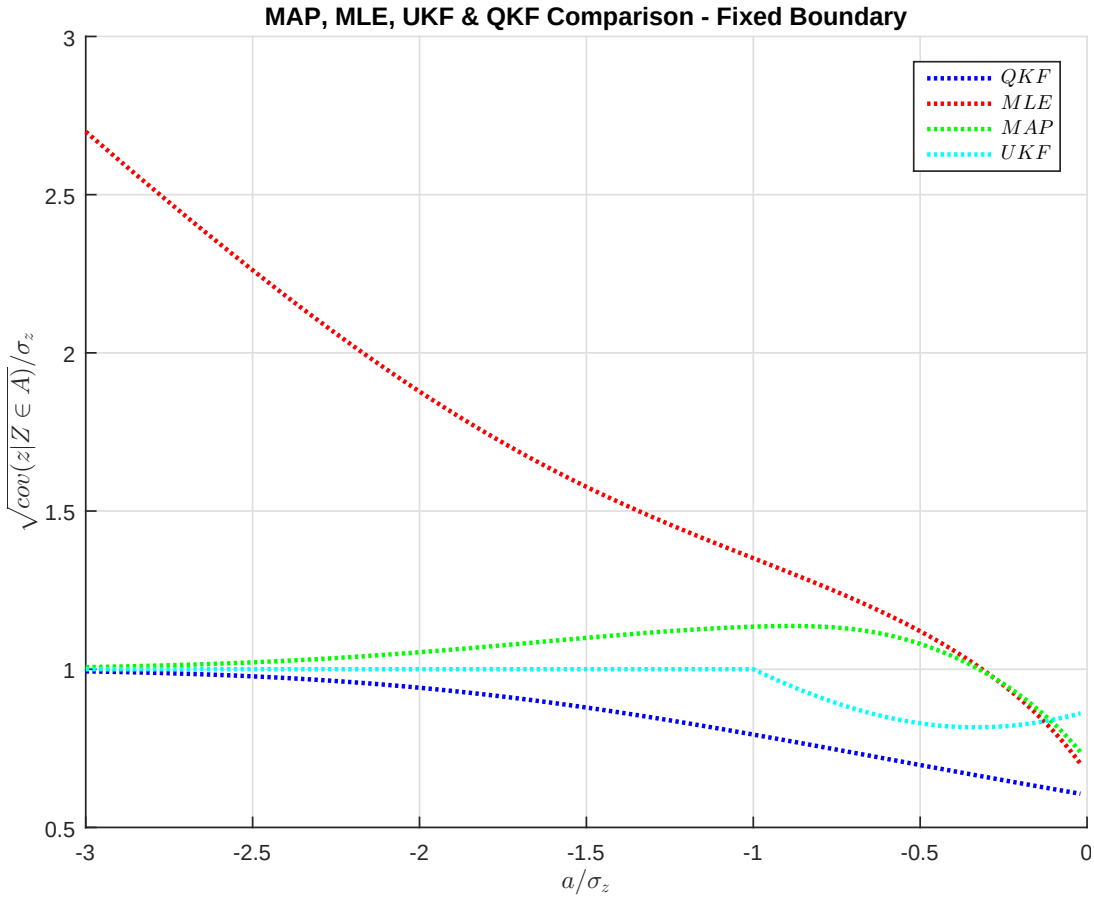


Figure 2.7: A comparison between the approximations methods and the QKF. The estimation of the quantized covariance is evaluated with distance for the non-conflicted case.

Fig. 2.6 shows that when the truncation is on the lower bound of the distribution, then the mean of the truncated variable increases compare to the mean of the original distribution. Moreover, Fig. 2.7 shows that the truncation reduces the variance compared with the variance in the untruncated distribution.

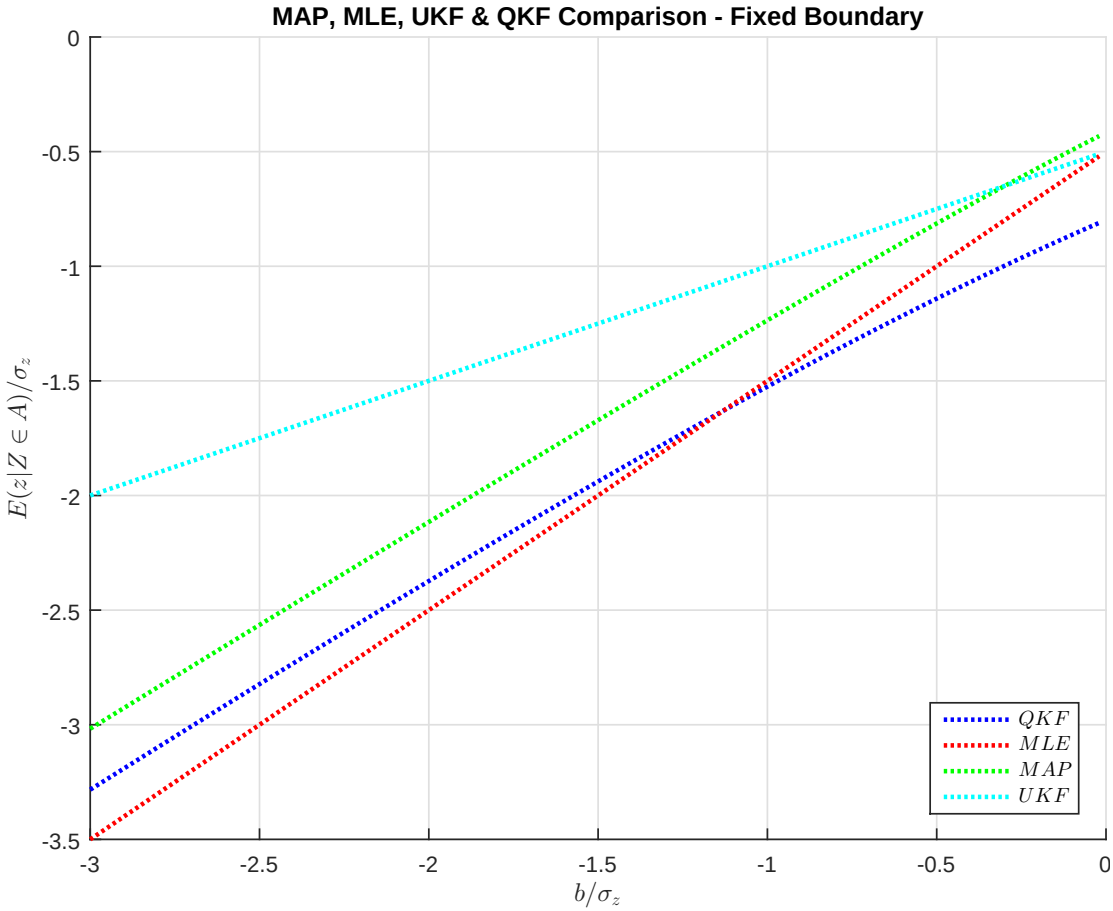


Figure 2.8: A comparison between the approximations methods and the QKF. The estimated mean of the quantized measurement is evaluated with normalized distance. Note that the case presented is the conflicted case where the conditional expectation is not the same as the incorporated observation.

Figure 2.8 and 2.9 illustrate the same running setting however in cases where the expected status is conflicted with the the observation (IN versus OUT). The truncation is bigger when the measurements are farther away from the mean value, and therefore the displacements are negative, which means that the expected value should be shifted behind the observer. The variance starts very small since the current expected value is conflicted from the beginning of that sequence of measurements. The variance gets larger as the measurements get closer. Hence the area under the density curve of the new bounds gets wider.

The UKF overestimate the results. The main reason is that the estimator additionally incorporates the observation error as can be seen in Eq. 2.59.

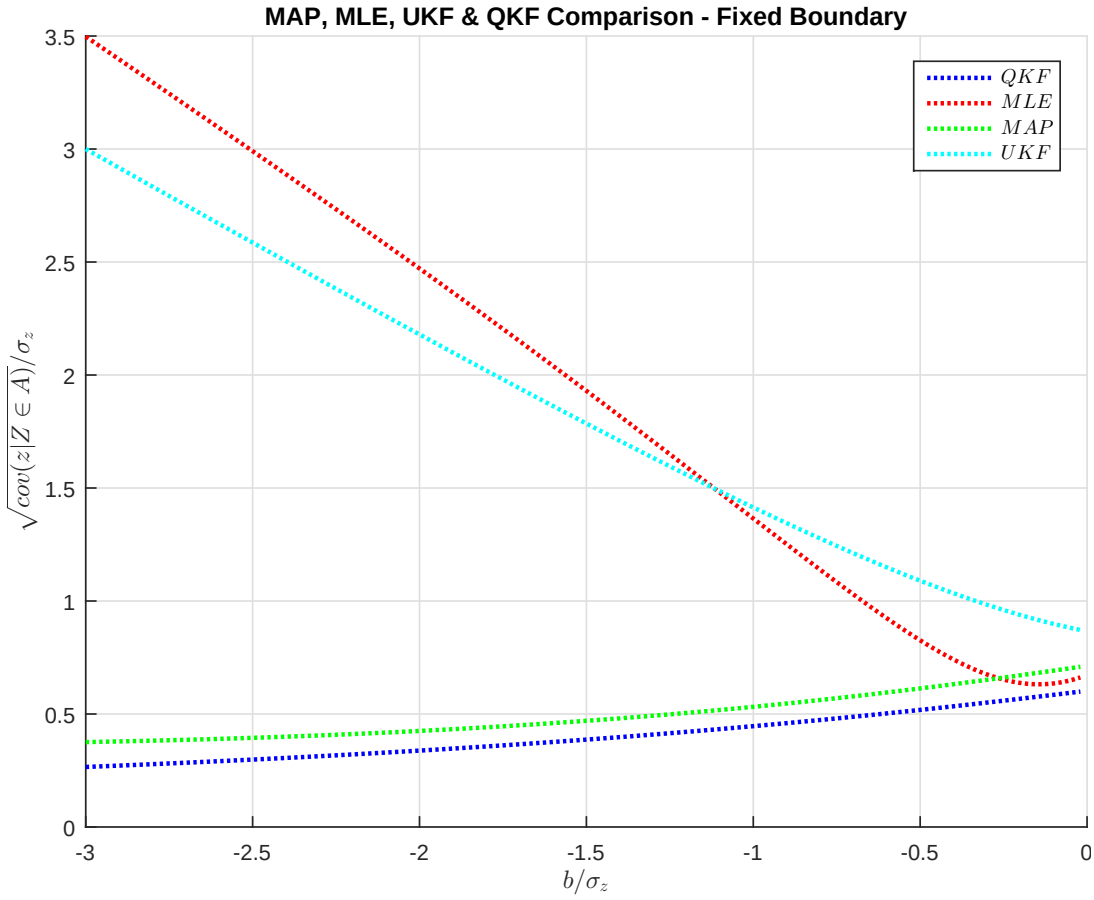


Figure 2.9: A comparison between the approximations methods and the QKF. The estimation of the quantized covariance is evaluated with distance. Note that the case presented is the conflicted case where the conditional expectation is not the same as the observation.

2.2.5 Simulation with a Single UAV

The simulation is designed to include all major components that are involved in the exploration mission. It simulates the propagation of a wildfire with a random and bounded spread rate along a wildfire periphery. The estimation of the periphery and its spread rate from quantized observations of the UAV is carried

out on a ground control system that fuses the observations based on the derived QKF method. A set of simple test cases are examined first. The performance is qualitatively examined with the help of the spatial uncertainty attached to each CP and is expected to improve as the number of measurements increases, or when the observations get closer and closer to the actual CP. Any detected observation, from inside or outside the periphery, could support the prediction. The new estimator interprets that situation as a conflicted observation or a non-conflicted observation. The initial setup chosen here is similar to an operational scale and is based on the recorded data received from the CAL FIRE (San Mateo Santa Cruz Unit for the Martin Incident). One can assume that the initial uncertainty is considerable at the time the UAVs are deployed ($\sim 1Km$). The variation of the initial location of the CP is due to the uncertainty of the predicted spread rate, ($\pm 180[m]$ after 30 *minutes* of when the phenomenon started).
The results are then based on an initial displacement of $\pm 180[m]$ between the actual and initial CP locations. Moreover, the spreading rate is noisy, with additional $\pm 0.1[m/sec]$ error.

Figures 2.10 and 2.11 illustrate the propagation of a single CP. As the observations get closer to the CP, the variance reduces along the line of sight, and the CP is shifted away from the UAV's position. The figure shows data only every ten time-steps along a pre-planned path, and for each time step Fig. 2.11 shows the ellipse that exists after incorporating the measurement. The ellipse represents the 50% uncertainty of a CP with the associated estimated covariance. In Fig. 2.10 the way the CP moves away from the observer is characteristic of any estimator of truncated data.

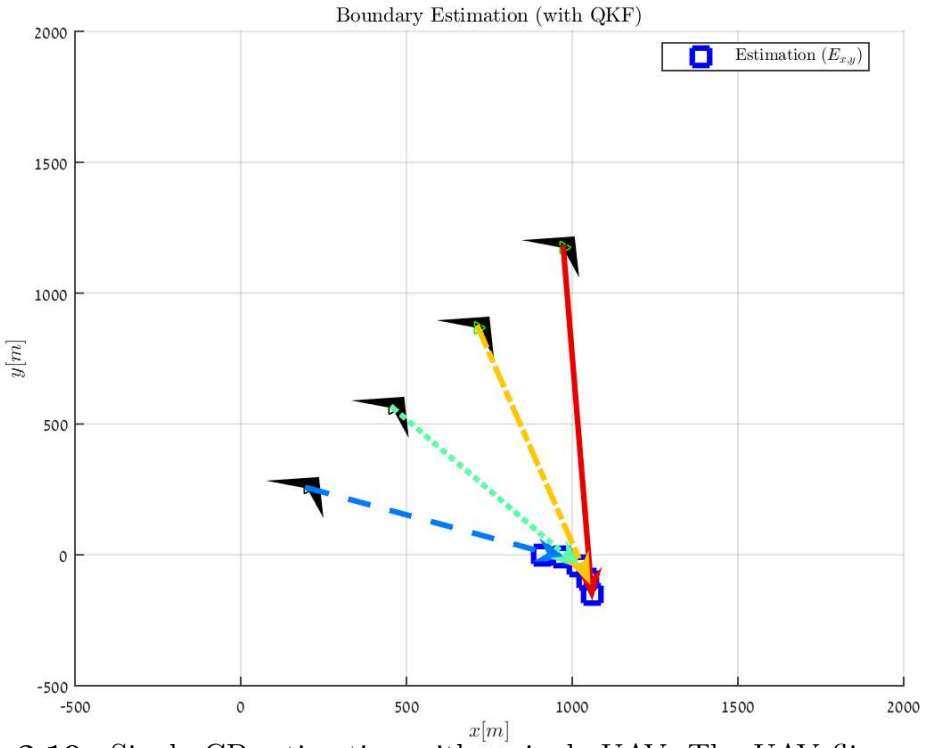


Figure 2.10: Single CP estimation with a single UAV. The UAV flies over the explored area on a pre-planned trajectory. The arrows pointing to the CP correspond to the UAV position.

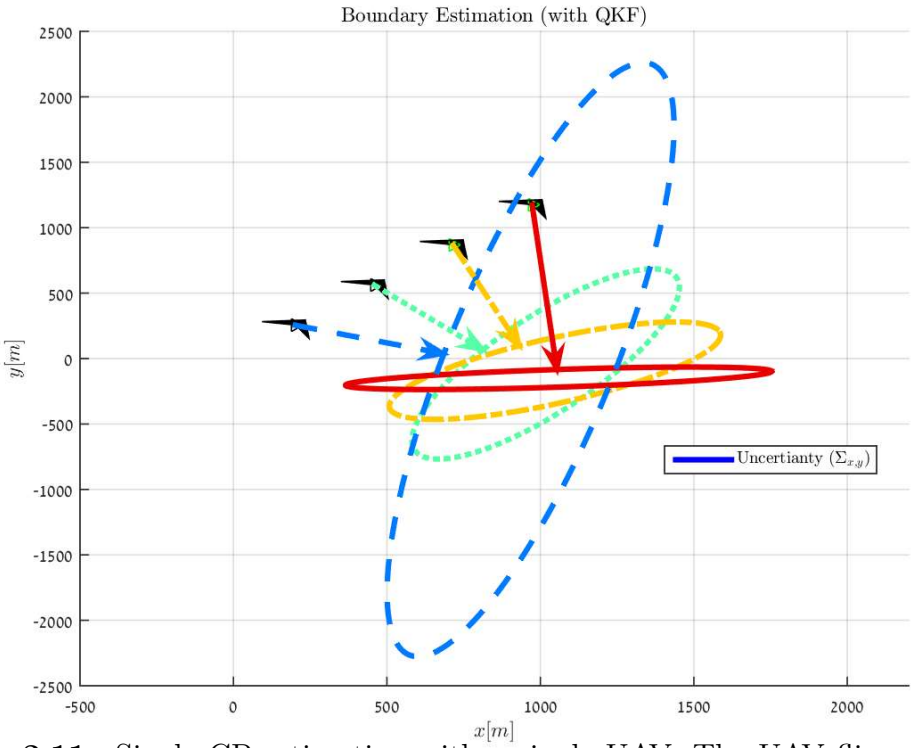


Figure 2.11: Single CP estimation with a single UAV. The UAV flies over the explored area on a pre-planned trajectory. The arrows pointing to the ellipses correspond to the UAV position and illustrate the directional effect of the covariance. Note how the ellipse starts large and is flattened with time and the UAV position

Figure 2.12 illustrates the propagated periphery and the predicted boundary. As the observations get closer to the CP, the variance reduces, and the CP shifts away from the UAV's observations. Notice that since the CPs start away from the actual periphery, there is a small displacement. When the UAV crosses the boundary, it enables an immediate correction to the closest CP to help reduce the displacement even more. The predicted boundary gradually converges to a value where the observations do not affect the CP since the projected variance is very tight and the evaluated probability is meaningless. The ellipse plotted in the figure is the last estimated outcome in the tested scenario.

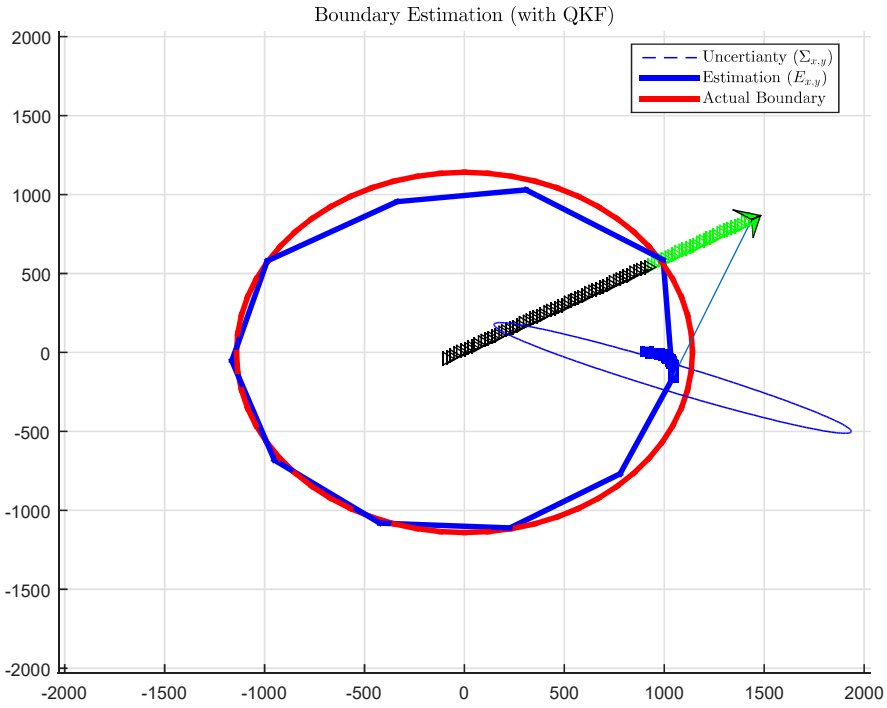


Figure 2.12: An example of periphery estimation with a single UAV. The red line represents the actual periphery, and the blue line represents the predicted one. The UAV flies over the explored area on a pre-planned trajectory. The line of sight to one CP illustrates the directional effect of the covariance (represented as an ellipse) with the QKF method.

2.3 UAV Strategies

2.3.1 Ellipse Steering

To improve the results even more, one can use a new approach to further suppress uncertainty. The UAV trajectory is changed to continuously reduce the uncertainty in the biggest covariance among all CPs, by flying directly to the tip of the major axis of that ellipse. Figure 2.13 illustrates the basic concept for reducing the uncertainty autonomously. Each associated uncertainty is represented as an ellipse (of 95% of confidence area).

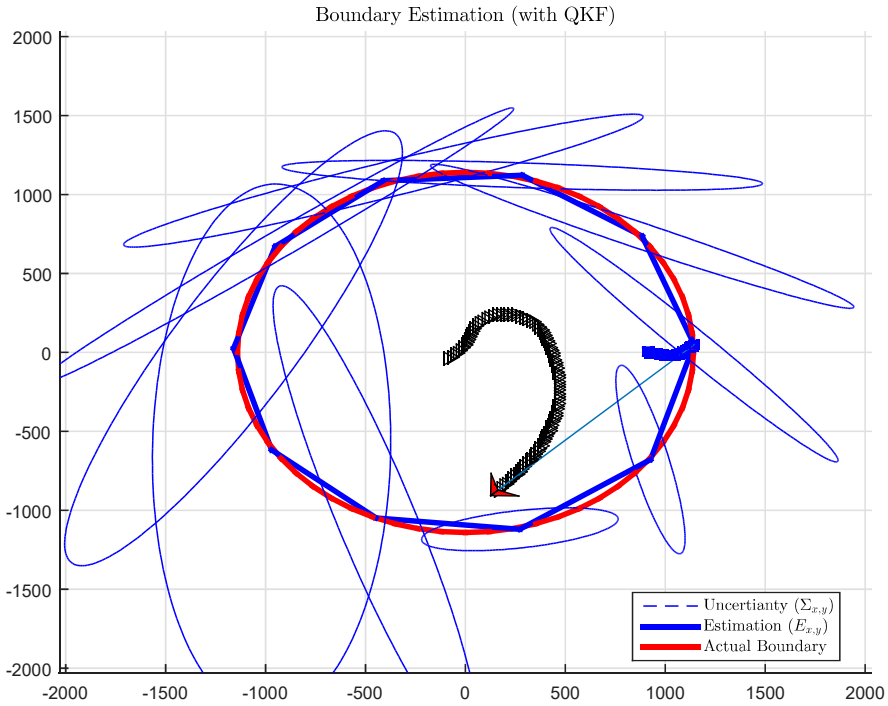


Figure 2.13: A periphery estimation with a single UAV for an autonomous mission is illustrated. The red line represents the actual periphery, and the blue line represents the estimated one. The UAV flies over the explored area autonomously. The line of sight to one of the CPs illustrates the directional effect of arbitrary CP. The QKF method is employed on all the CPs simultaneously, and the UAVs identify the current highest uncertainty to approach next.

If resources are not limited, the monitoring mission can be accomplished by

more than one UAV. Figure 2.14 illustrates the next level for reducing the uncertainty with two UAVs. The system coordinates their flight trajectories by assigning them to the closest and highest uncertainty. This method avoids creating a situation where it takes a long time to reach an unexplored area.

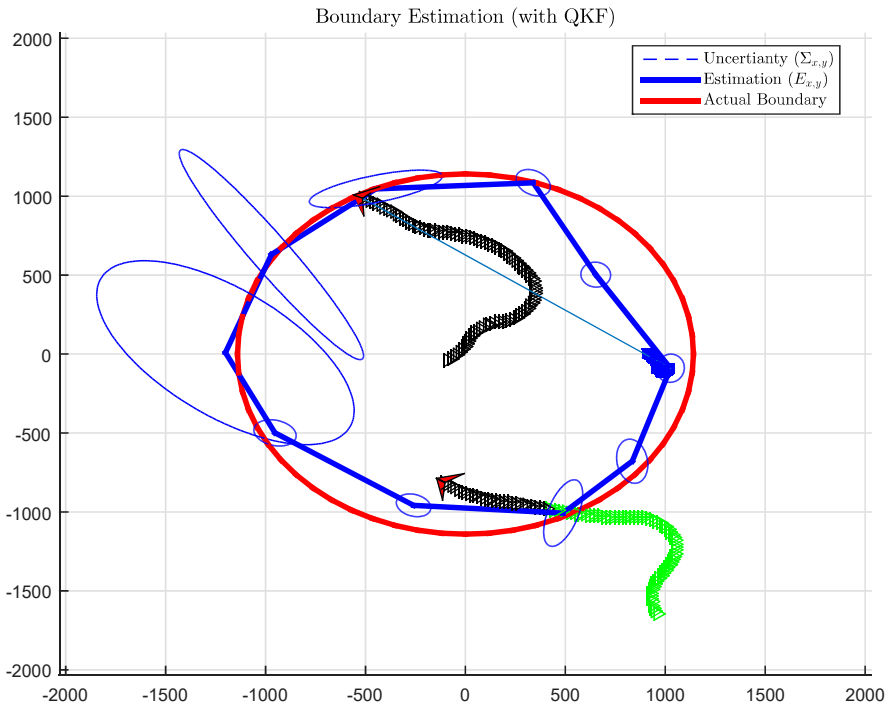


Figure 2.14: A periphery estimation with two UAVs. The UAVs fly over the explored area. The green trace represents trajectory that is outside of the periphery. The QKF method is employed on all CPs simultaneously, and the UAV identifies the current highest uncertainty to approach next. The final result of each CP uncertainty is represented here by an ellipse.

In addition to the QKF estimator employed in this example, the simulation includes a heuristic for crossing the actual boundary and switching the assigned CP. The logic of crossing enables updating the closest CP to the crossing point. The logic of switching is simple as well. The UAV switches from CP to CP any time the major axis of any other ellipse grows more than the one the UAV is currently assigned to. The concept of switching will be explored further.

2.3.2 Greedy Uncertainty Suppression Method

This section focuses on a methodology for monitoring the periphery of propagating phenomena with multi-UAVs. The methodology utilizes a coordination policy on top of a boundary estimator to decrease the aggregate uncertainty of the monitored periphery. The estimator reconstructs the boundary and improves the knowledge of its current expected value. The strategy to directly assign the UAVs to track a periphery is compared with a policy that considers directional uncertainties. After the coordination policy is determined, the algorithm is tested in a multi-UAV dynamic simulation. A detailed numerical analysis of the proposed algorithm and extensive simulation results are presented.

The new class of one-dimensional quantized estimator [76] lays the groundwork for the Greedy Uncertainty Suppression (GUS) strategy. The monitoring application involves large numbers of possibly randomly distributed inexpensive sensors, with limited sensing and processing. The estimator incorporates observations gathered by multiple observers and uses the QKF estimation method [27] to update the expected location and unobserved spread rate.

The main objective of the method is to find trajectories that improve the estimated boundary of a propagated phenomenon. The estimation is meaningless in a situation where the available sensors are located inefficiently (e.g., considerably far or colocated). Other methods for online look-ahead approaches to re-routing the UAVs are computationally intractable [73]. The GUS method utilizes a proactive monitoring approach supplemented by a QKF estimator and a simplified propagation model. The system relies on a rapidly deployable fleet of UAVs designed to detect limited information, generate an uncertainty map, and incorporate that information into new allocation tasks.

GUS - Coordination

Previous results in this chapter have suggested that reducing uncertainty is related to distance as well as the line of sight to a CP. Moreover, uncertainty depends on the availability of measurements. Hence uncertainty grows with time when no significant new observations have been incorporated. The UAV can approach a CP along the direction of its maximal uncertainty axis (direction of major axis of the covariance) and reduce the one-dimensional uncertainty.

The coordination addresses the monitoring problem by adopting common principles. The first principle is sorting. The mission controller keeps track of representative quantities of interest; the predicted variance and the variance rate of change. The QKF estimator evaluates the first and the second is a derivative of the two last evaluations (numerical gradient). These quantities are used later on to assign the UAVs to the target space (i.e., the CPs).

The GUS strategy accounts for the uncertainty by evaluating the uncertainty perpendicular to the boundary. The component along the periphery does not affect boundary position uncertainty to first order. The CPs are being sorted by their associated cost. The goal of the GUS strategy is to coordinate between UAVs and assign a UAV to one of the CPs to reduce the uncertainty accordingly.

$$J = \sigma_{CP_i} + \dot{\sigma}_{CP_i} \cdot t_{go} \tag{2.63}$$

Eqn. 2.63 is the cost function. It combines two components and gives the total value of any feasible assignment, the current associated uncertainty (σ_{CP_i}) and the uncertainty rate of change ($\dot{\sigma}_{CP_i}$). The higher the uncertainty, the higher is the cost. When the rate is taken into consideration, the dynamics of the UAVs are considered as well. The cost function accounts for the time it takes the selected

UAV to reach an arbitrary variance aim point and time-to-go (t_{go}) helps predict the size of the uncertainty when a UAV flies by an assigned CP. Time-to-go refers to the time it takes for a UAV to fly toward the perpendicular point along a 50% error ellipse (i.e., the target point).

Deploying the UAVs is based on the number of available UAVs. For example, for two UAVs, the deployment is to two different CPs, where one UAV is assigned to the highest look-ahead uncertainty, J, and the second direction is the highest cost (J) of the remaining CPs.

There are three benefits from this allocation policy. First, the solution avoids flyby trajectories and potential collisions. Second, UAVs are not allocated to the same or even a close area. Third, the trajectories are being evaluated for dynamic trajectory feasibility to be carried out by the assigned UAV.

GUS strategy tends to minimize the maximum uncertainty over all CPs by incorporating observations over a long period. The policy achieves longer look-ahead with an on-line re-routing logic for the fleet members' task.

GUS - Implementation

The implementation includes two main parts: coordination and allocation. The basic operation leading to coordination involves sharing information for the assigned tasks. The UAVs share their observations with a centralized entity. The observations are incorporated sequentially in the estimation process. GUS algorithm is a step-by-step procedure to determine the best task for each UAV. The notation uses superscript j to label the UAV and i as an index of an arbitrary CP.

The first step relies on a previously developed algorithm (QKF). This procedure includes system coordinates transformation, scalar probability evaluation,

and Kalman Filter to estimate the state ($\hat{x}_{x,y}^{CP_i}$) and covariance ($\hat{P}_{x,y}^{CP_i}$) of the CPs in the original coordinate frame.

The following steps determine the new policy. The second step runs Dubin's Vehicle algorithm to evaluate all the trajectory alternatives, which also provides the length of feasible trajectories and evaluates the time-to-go for each UAV. The associated trajectory for UAV_j, evaluated by the cost function in the fourth step, assigns the UAV to its best feasible task (that is, the CP with the highest cost). The values of the cost function address the need to consider additional restrictions or tasks (for example, deploying the UAVs to one side of the periphery).

The third step, sorts the CPs' estimation by their cost, J. After correcting the state ($\hat{x}_{x,y}^{CP_i}$) and updating the covariances ($\hat{P}_{x,y}^{CP_i}$), the procedure continues by evaluating the perpendicular component of the major axes of the uncertainties. It provides a list of variances with their associated waypoints. Each waypoint is a candidate target which lies along a different line of sight.

The propagation of the error between the predicted and actual boundary can increase without control. An additional objective of the GUS technique is to reduce the errors. It enforces boundary crossing where the UAV had not crossed the actual boundary for a set duration. In that special allocation mode, the UAV is rerouted by setting the origin point as the target, and after crossing the boundary, it switches back to default allocation mode.

2.3.3 Simulation and Results

The simulation is designed to include all major components which are involved in the GUS strategy. The environmental conditions are being simulated based on a model of a propagated wildfire with a random and bounded spread rate ($3 \pm 0.1[m/sec]$). The UAV allocation is being implemented in a separate component

that incorporates the observations gathered by the simulated UAVs.

The UAV dynamics model is subject to a constant speed of 20 $[m/sec]$, the approximate speed of the UAV that has been developed and examined for the experimental stage of this thesis. Moreover, the centralized controller is the QKF estimator, that fuses the observations and is based on a previously derived technique. The following simulated scenarios explore the efficiency of the GUS algorithm.

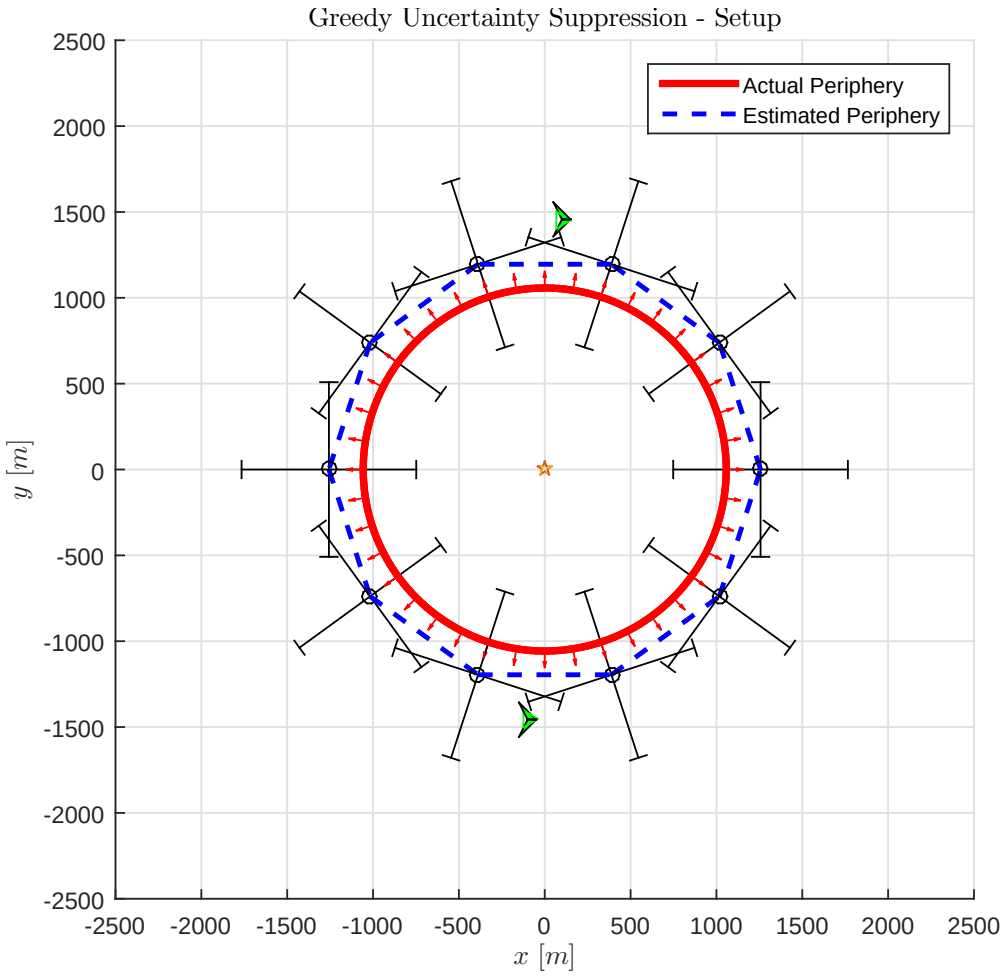


Figure 2.15: Initial setup. The UAVs are at the final stage of the deployment phase and located on the opposite sides of the boundaries. The actual periphery is a solid red line, and the predicted periphery is a dashed blue line. The error bar associated with an arbitrary CP represents its current perpendicular uncertainty (1σ). Note that the error bar is equal and results in a predetermined prediction that is based on a maximal spread rate.

The initial setup attempts to adhere to conditions similar to the real problem and therefore uses real-time data of a known wildfire incident (Martin incident 2008). For example, the initial AOI (area of interest) is large ($1Km \times 1Km$), and the time scale is long (i.e., hours). Fig. 2.15 shows the actual periphery with two UAVs deployed from both sides of it.

The propagation model used in the simulation is simplified. However, it still

allows investigating the major properties of fire spreading. The dynamic expansion of the boundary, the environmental effects (i.e., wind and slope) and the feasibility are all considered in the implementation and are utilized for different scenarios.

The simulation presents a qualitative evaluation of the GUS strategy compared to the traditional periphery tracking strategy as an obvious benchmark. In previous studies ([20],[21],[92]), the strategies explored are either that the UAV moves in a spiral pattern along the perimeter, or that the target space is divided between UAVs (Partition).

The core of the GUS strategy relies on the fact that the major influence of the uncertainty is along the line of sight. Fig. 2.16 demonstrates the relative orientation of the UAVs and the changes of magnitude and direction of the major axis. Conceptually, rerouting the UAVs based on the uncertainty, or the instantaneous largest uncertainty affects the rate of change of that uncertainty.

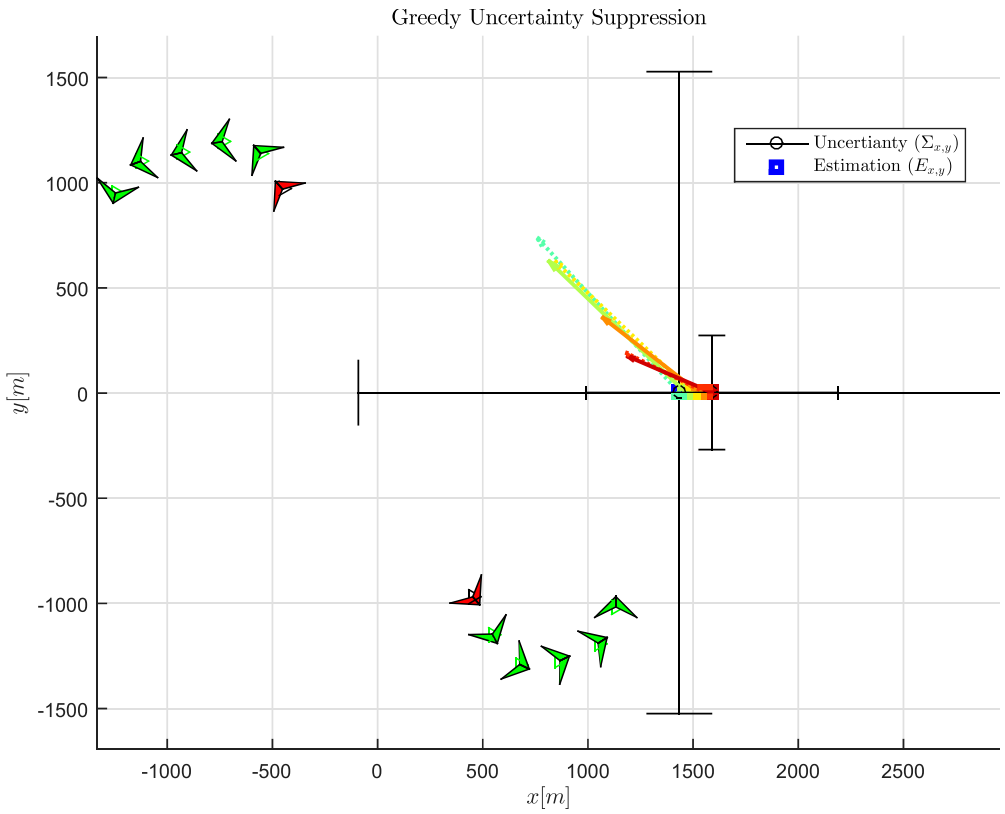
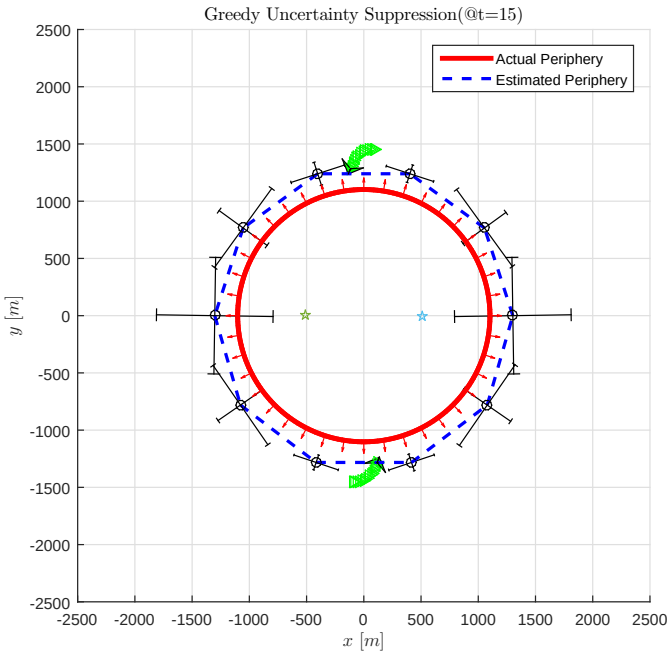
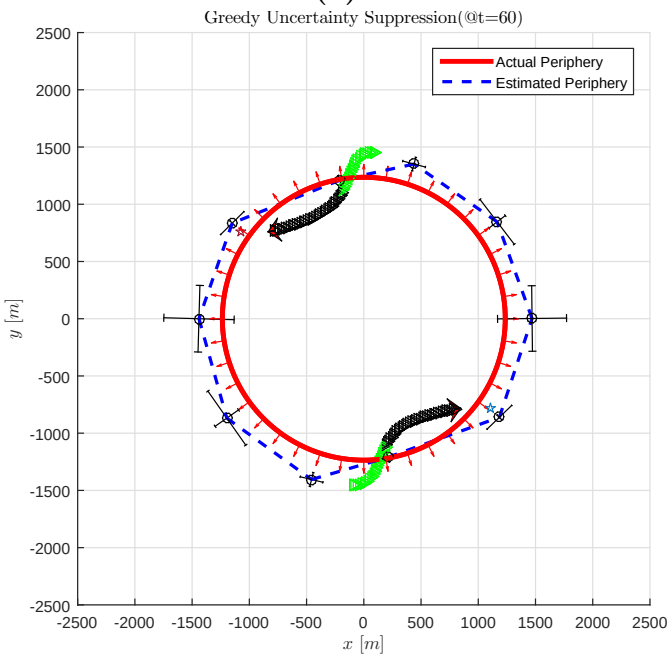


Figure 2.16: The major axis of uncertainty over a number of times. The major axis illustrated with an arrow. The crossbar represents the uncertainty of the first and last CP locations. The UAVs move on reroute trajectories. The direction and magnitude of the major axis changes with the deployment of the UAVs and with the incorporation of the observations.



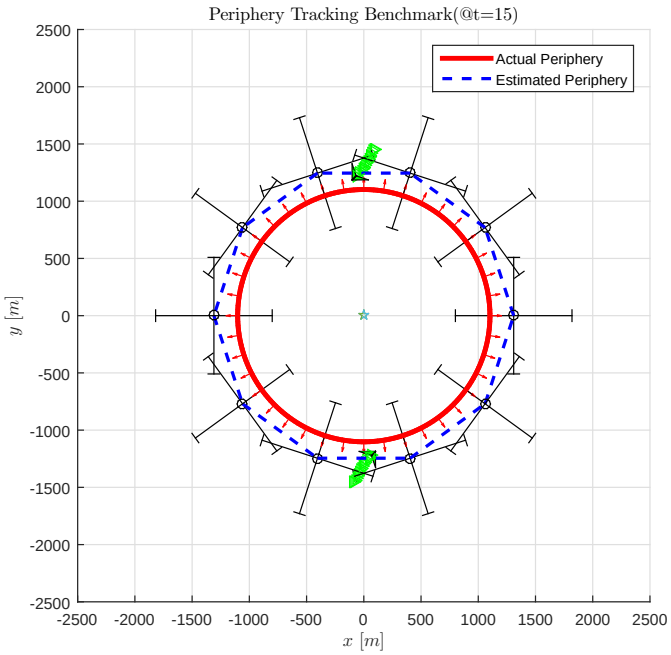
(a)



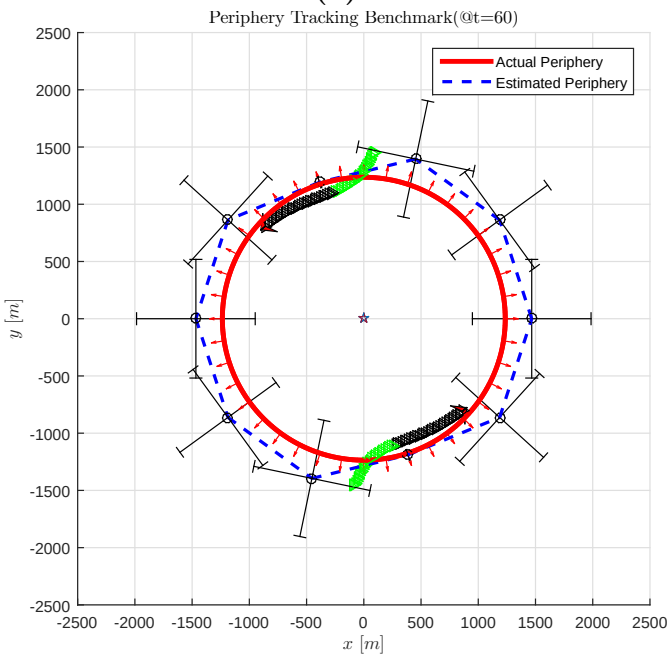
(b)

Figure 2.17: Two set periods of time for GUS strategy. Subfigure (a) presents the UAVs approach to the boundary from opposite sides. Subfigure (b) includes an update of the last crossing points and the UAVs heading to their assigned CPs. Note that the uncertainty of the close CP is already reduced.

The GUS strategy reduces the uncertainty as the observations get closer to the grid points. This is a result of the truncation step, which is incorporated in the one-dimensional probability distribution. Any temporary deployment of the UAVs influences the uncertainty of an arbitrary CP. Fig. 2.17 shows two periods of time of a propagated boundary, estimated boundary, and employed UAVs. When a UAV crosses the boundary, the estimator incorporates the measurement and updates the closest CP properties (for location and covariance).



(a)



(b)

Figure 2.18: Two set periods of time for Periphery Tracking strategy. Subfigure (a) presents the UAVs approach to the boundary from opposite sides. Subfigure (b) includes an update of the last crossing points and how the UAVs are directed to their assigned CPs. Note that the uncertainty of the nearby CP decreases.

In the traditional periphery tracking strategy, the UAVs are allocated optimally and follow the edge of the periphery continuously. Spacing UAVs evenly along the circumference minimizes the maximum associated age, which is the time elapsed since it was last observed. There has been thorough work done with a similar objective to this study [20]. That work, used here as a benchmark, has shown that by deploying multiple UAVs the propagated periphery can be followed autonomously. The simulated procedure of the benchmark predicts the covariance and sets it to a minimal value when a UAV crosses the CP.

Assessment of strategies outputs

The performance measure can be used to evaluate each method. The results are achieved by taking each estimated CP and comparing it to the actual periphery.

The performance measure accounts for two performance indicators: errors and uncertainty. The errors indicator comprises the mean-square-error, where the errors are between the predicted and the actual periphery. The uncertainty indicator is simply taking the mean of the CPs' variances perpendicular to the boundary.

$$Perf = \sqrt{\sum_{i}^{N}(err_{CP_i}^2 + \sigma_{\perp_{CP_i}}^2)} \tag{2.64}$$

Both indicators are weighted equally in the combined performance measure. Fig. 2.19 demonstrate the calculated parameters of the performances measure.

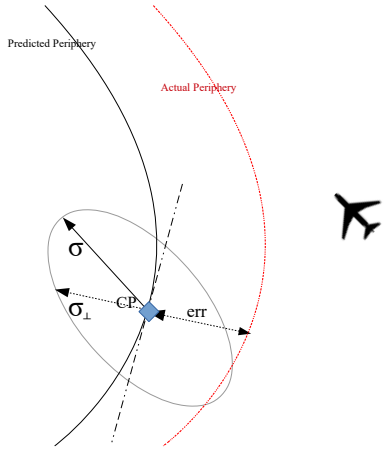


Figure 2.19: The performance indicators are presented. The size of the error between actual and predicted boundary and the perpendicular variance are presented.

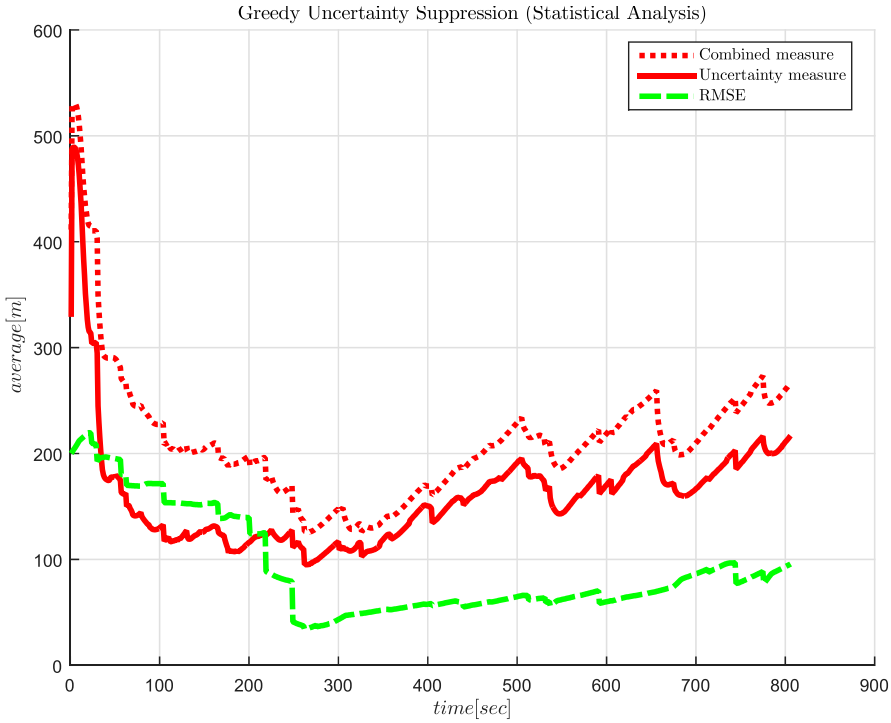


Figure 2.20: Performance analysis. The solid red line represents the perpendicular standard deviation average, the dashed green line shows the cumulative root mean squared error, and the dashed red line is the combined performance measure. Note that the mean value of the uncertainty is reduced during the mission, and the error increase as the periphery evolves since the number of crosses per AOI get smaller.

Figure 2.20 shows the combined performance measure with its two performance indicators. If there were no errors and no uncertainty, then the traditional periphery tracking was an optimal approach. In practice, the uncertainty grows with time, and although errors are reduced to a minimum when the UAV crosses by the CP, the spread rate is not observable, and the errors continue to grow shortly after updating the location with the nearby CP.

Fig. 2.21 demonstrates the scenario with running GUS. A local error-bar represents the uncertainty of each CP. The size of an error-bar is correlated with the size of the perpendicular and tangential variances. The resulting trajectories are different from the benchmark, presented in Fig. 2.22.

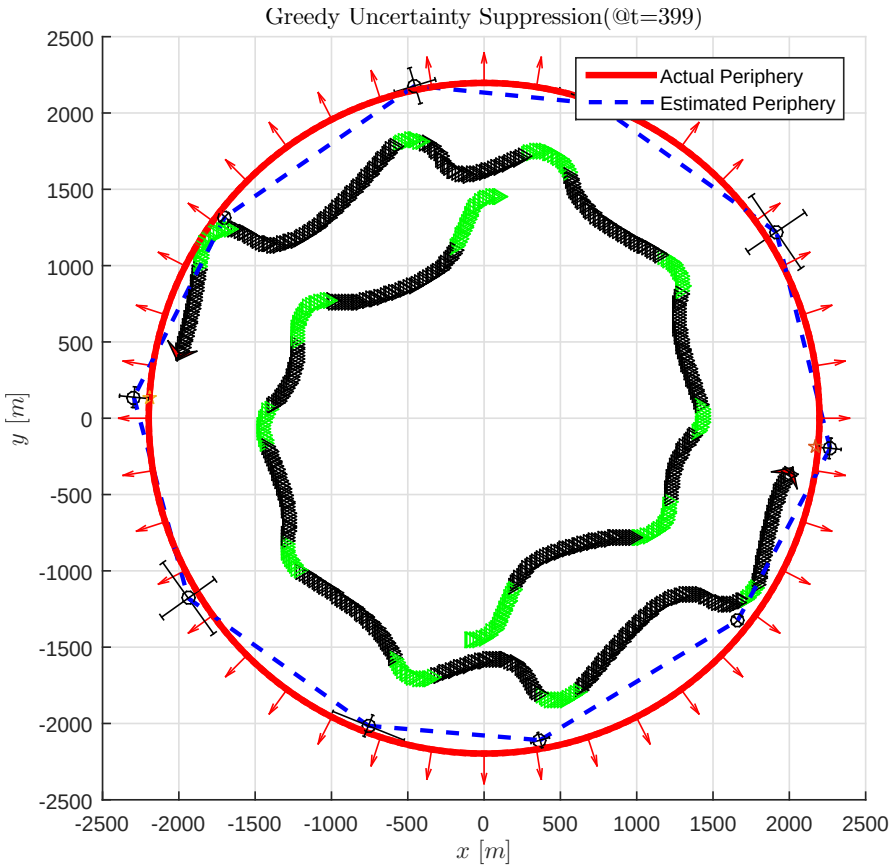


Figure 2.21: Estimation and Coordination with the GUS method. The UAVs switched from the deployment phase to track the highest uncertainties. The actual periphery is a red solid line, and the predicted periphery a blue dashed line. The UAV trail is in green where the UAV is OUT and in black where the UAV is IN. The error bar associated with each CP represents its current uncertainty. Note that the error bar decreases as the UAV approaches an arbitrary CP and that the observations cause the directional uncertainty of the other CPs to decrease.

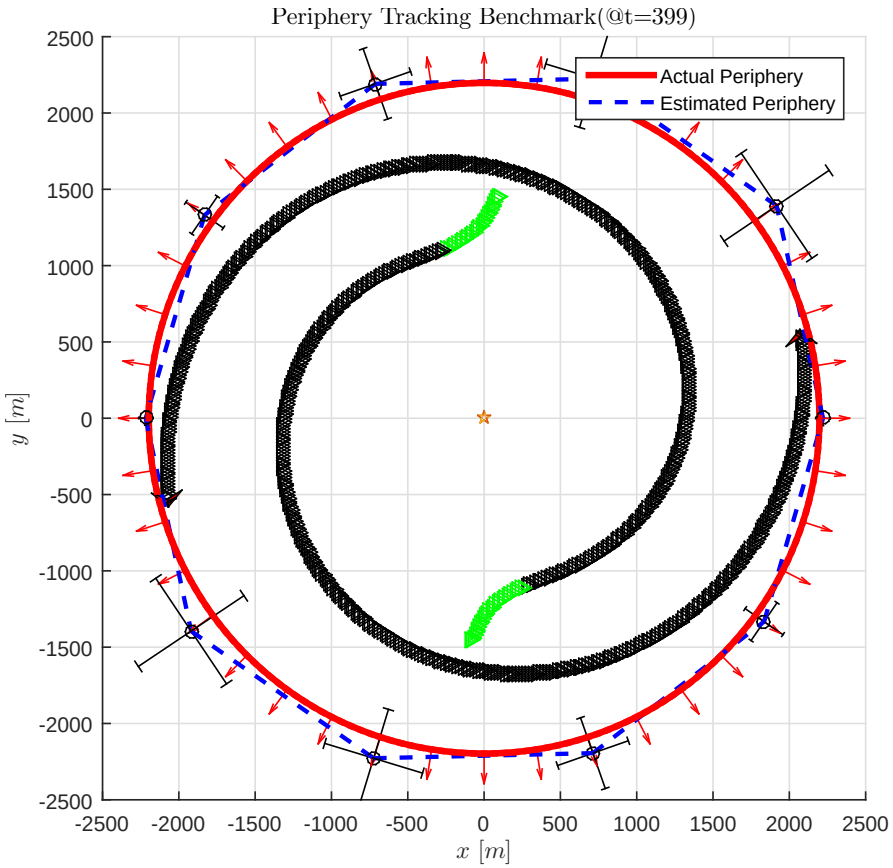
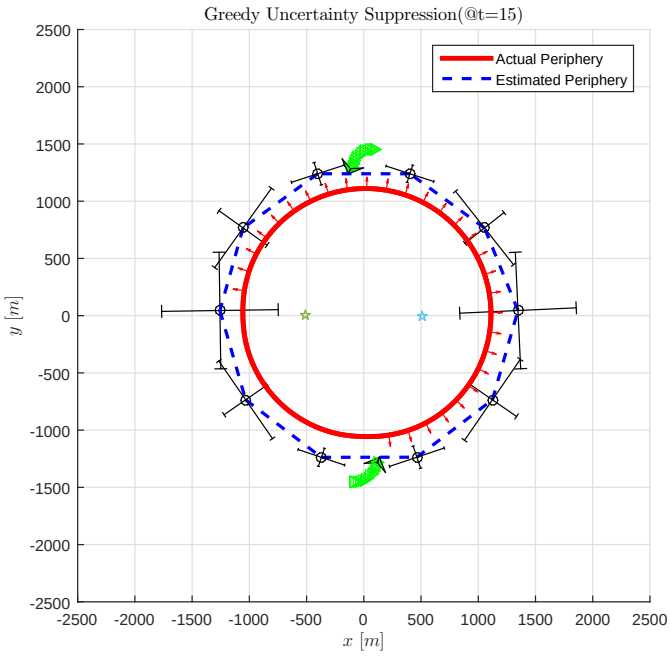
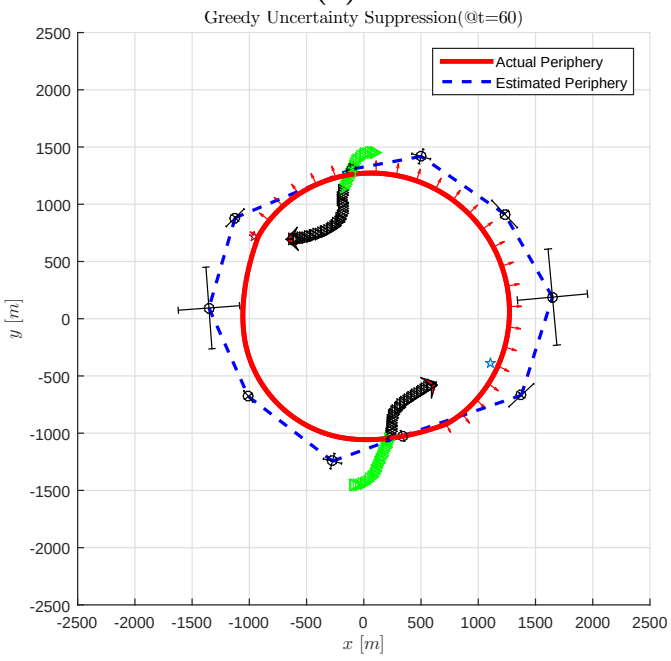


Figure 2.22: A benchmark for periphery tracking. The UAVs fly evenly spaced along the edges of the propagating perimeter. The actual periphery is a red line, and the predicted periphery is a blue line. The error bar associated with each CP represents its current uncertainty and not its associated age. Note that the error bars grow with time and are reduced to a minimum size as the UAV crosses a CP.

Fig. 2.23 shows two periods of time of a propagated boundary, estimated boundary, and employed UAVs for a scenario with wind. When a UAV crosses the boundary, the estimator incorporates the measurement and updates the closest CP properties (for location and covariance).



(a)



(b)

Figure 2.23: Two set periods of time for estimated periphery with southwest wind. Subfigure (a) presents the UAVs approaching from oposite sides of the boundary. Subfigure (b) includes an update of the last CPs, and the UAVs are headed to their assigned CPs.

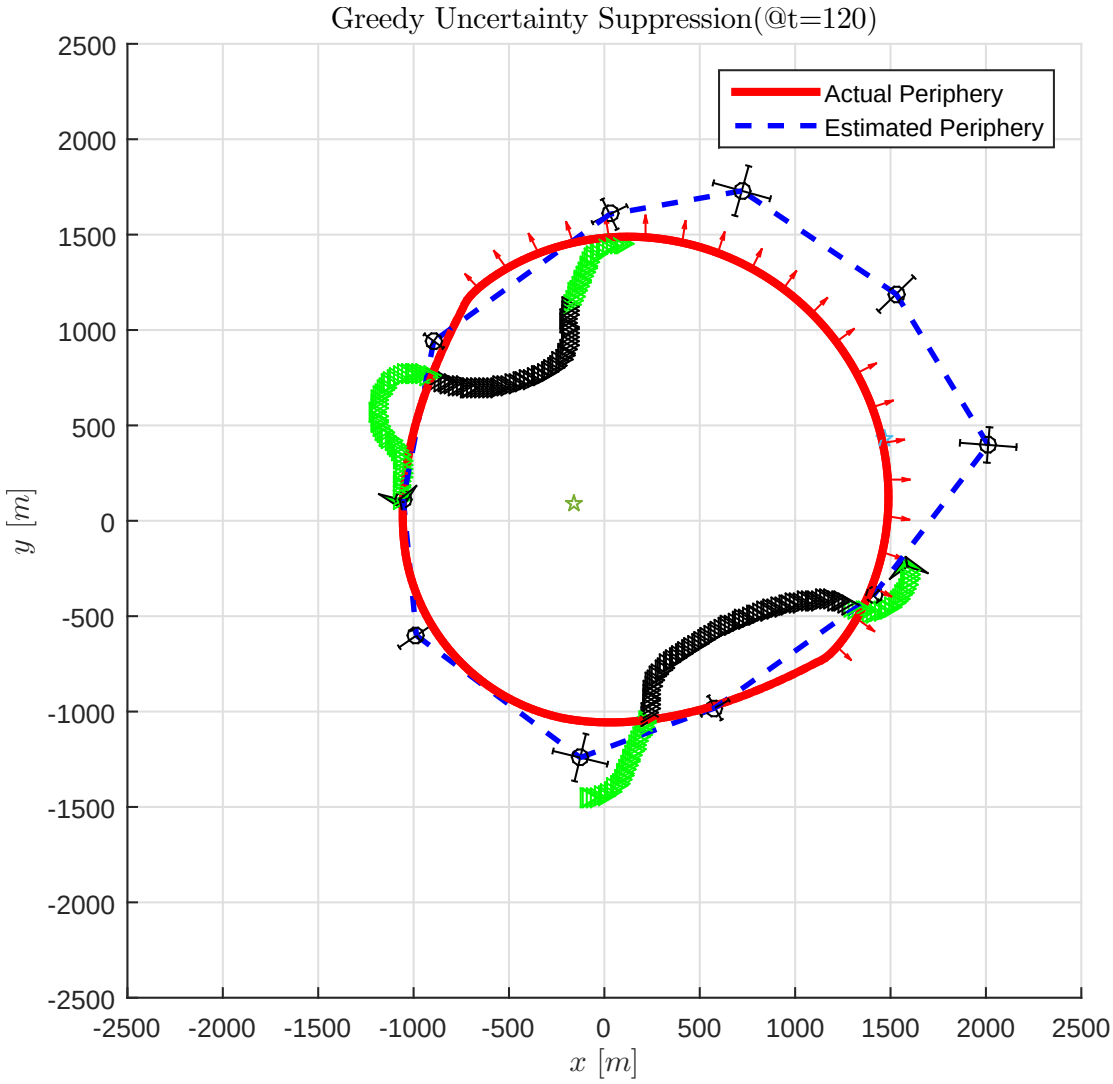


Figure 2.24: Estimated periphery with southwest wind (at a later time). The plot presents the UAVs approaching from oposite sides of the boundary, the updated CPs, and the lopsided periphery.

Fig. 2.24 shows a later period of time of the propagated boundary, estimated boundary, and employed UAVs for a scenario with wind. The periphery with a constant wind is expected to have a backing fire segment and a heading fire segment (see Fig. 4.6) and the entire periphery to be more lopsided.

The performances in Fig. 2.25 are evaluated relative to the perpendicular component of the local predicted periphery.

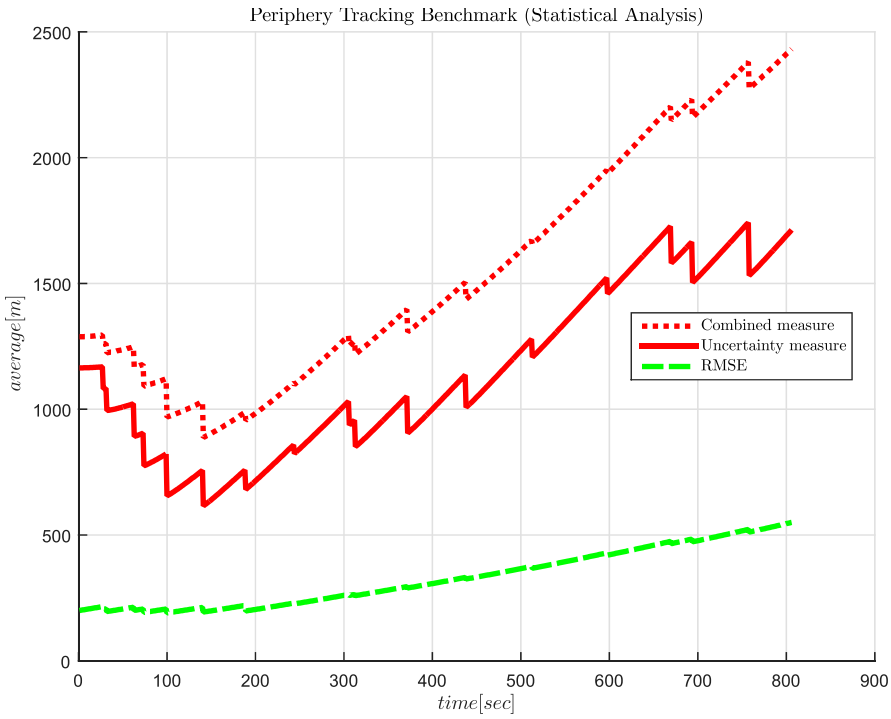


Figure 2.25: A performance analysis of the traditional Periphery Tracking with a southwest wind. The solid red line represents the perpendicular average standard deviation, the dashed green line shows the cumulative RMSE, and the dotted red line is the combined performance measure.

Figure 2.25 demonstrates a scenario which includes two UAVs. The estimator reduces the uncertainty as long as the UAVs are visiting or approaching different parts of the periphery often. When the periphery growth is above a certain limit, and the number of assigned UAV is too small, the performance measure indicates a necessity of more resources.

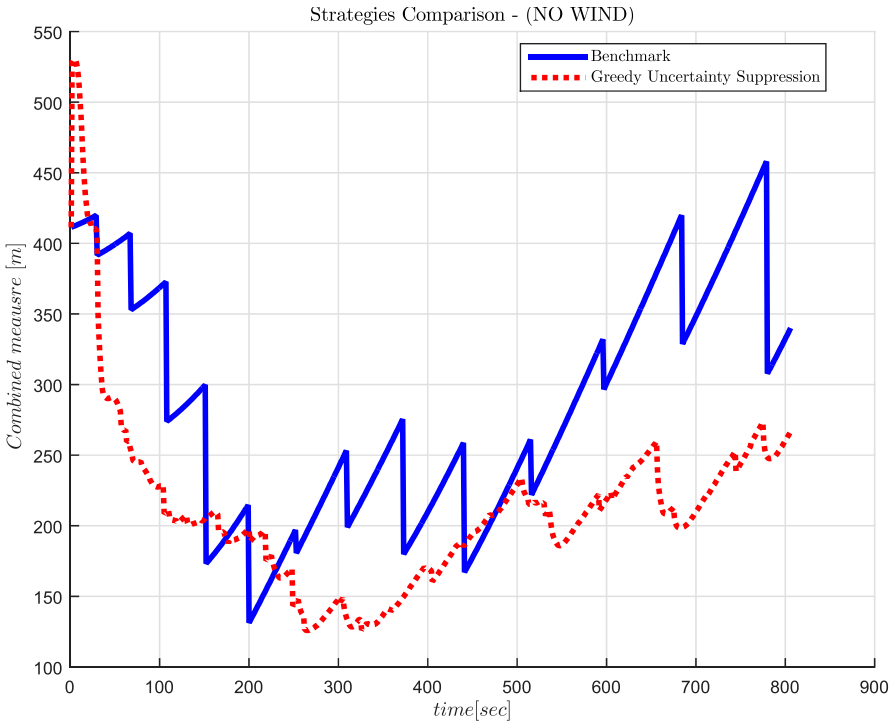


Figure 2.26: A comparison of the Greedy Uncertainty Suppression and the benchmark. The solid blue line and the dotted red line represent the combined RMSE performance measure over time for the benchmark and the GUS strategies accordingly.

Figure 2.26 demonstrates that the GUS and the benchmark have similar performance and the same trend over time of the mission. The GUS estimator reduces the uncertainty based on the algorithmically derived (non-constant) visiting period of the selected CPs. The benchmark has a fixed visiting period when the UAVs move along the periphery.

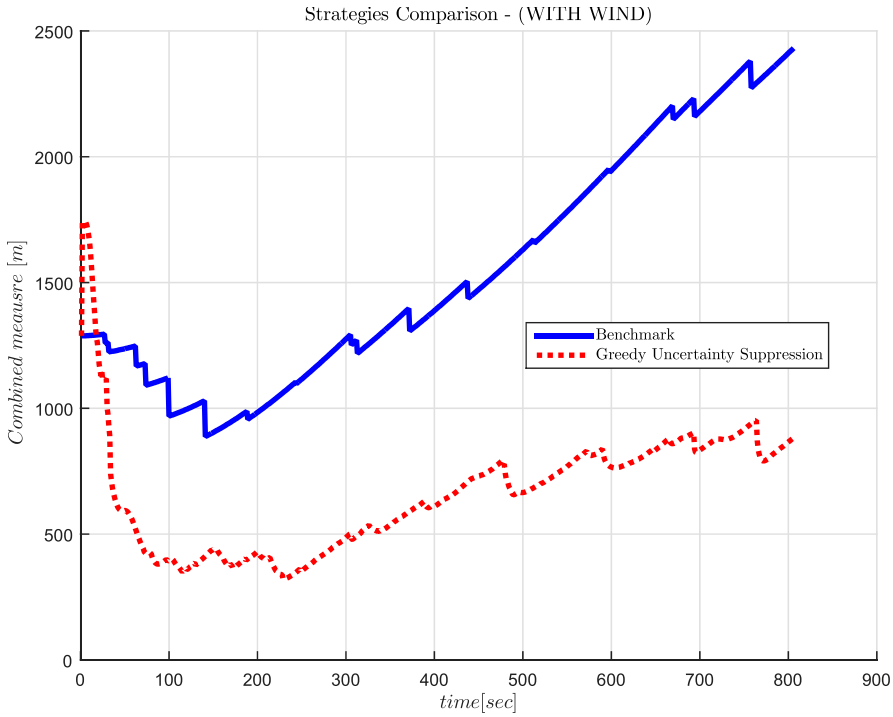


Figure 2.27: A comparison of strategies with a southwest wind. The solid blue line and the dotted red line represent the combined RMSE performance measure over time for the benchmark and the GUS strategies accordingly.

Figure 2.27 demonstrates that the GUS and the benchmark have very different performance for a scenario with a wind. The GUS reduces the uncertainty much more over the time of the mission.

2.4 Conclusions

Several characteristics are common to the implementation of the methods in this work. First, the methods are implemented such that they can work recursively. Hence, the output of one iteration becomes the input to the next. The alternative is to process all the past observations in each time-step, although this is not recommended and sometimes it is not even possible in real-time systems. Second, the initial covariance is based on previous knowledge of the system and good judgment. The results show that even if the initial guess is too high, the

variance will converge to some reasonable value over time. Third, the prediction step applies to all of the monitoring techniques (both GUS and the traditional Periphery Tracking). The prediction model for the spread of the periphery is (knowingly) inaccurate and includes an additional term which represents the differences relative to truth dynamics which causes the variance to inflate without new measurement updates.

The QKF is similar to Kalman filter architecture (predictor-corrector). Data fusion corrects the prediction step. One should notice that the first term in the innovation ($E(z|z \in A) - H\bar{x}$) is similar to the usual distribution method. The truncation, although with an optimization approach, is also utilized in MLE and MAP approximations. This explains the reason why the approximations have similar trends as QKF. The MAP technique, however, compared with the QKF estimator deflates the variance much more slowly as each update step is a product of the resulting ML distribution and the prior distribution.

Usually the update step in the KF deflates the covariance matrix (Eq. 2.17) but QKF inflates the covariance matrix (Eq. 2.24) relative to no quantization. Each incorporated observation contains new information, and it is combined with the state estimate in the update step. The measurement observed by the sensor is binary information (either inside or outside the periphery). Although the observation state does not comprise the complete state vector, the QKF approach and the alternative approximation techniques update the state vector completely.

The estimation process of the QKF estimator updates the covariance matrix. The diagonal entries represent the principal uncertainties where each one is associated with a state vector element. The off-diagonals represent the relationship between elements of the estimated vector or how correlated they are. QKF and MAP approach implicitly use these correlations to update the unobserved

state. For example, the relative distance to the boundary is effectively measured, since the location and the spread rate are correlated in the dynamic model any changes of one affect the other. Fusing spatial measurements in the estimators also changes the spread rate of the CPs in addition to updating their location. Moreover, even in the one-dimensional update step, data incorporation affects the correlated states that are codependent (for instance, x_a with v_a).

Although the data is quantized and the information is partial, methods that use multi-sensor and fit the observation to distribution properties help reconstruct the data, especially if the sensors are spatially distributed. A multi-sensor application can mitigate the quantization effects, and it is expected to achieve more accurate results by using multiple sensors (deployed on multi-UAV platforms).

The periphery is defined by a set of CPs where their state is continuously estimated. The QKF method estimates their position with binary observations. The results show that even though initial uncertainties can be significantly large, the QKF can significantly reduce the variance. Although it reduces the variance from one direction, it is shown that by adding more UAVs that move in different directions, the growing variance can be limited. By altering the trajectory toward the highest major axis of the ellipse (the uncertainty), the UAV can continuously reduce the uncertainty of the periphery.

The traditional monitoring problems follow the least visited point of interest and try to minimize the maximum visited points. It is argued that the best way to monitor the perimeter is by following the CPs. However, tracking the periphery to update the CPs misses one of the primary objectives of monitoring problems which is the uncertainty. A UAV fly-by CP can utilize the additional observation to improve the accuracy of CPs, which need frequent visits (for example, in a windy scenario where the heading fire moves fast).

When the allocation process assigns a task to a UAV, it should consider the contribution to the periphery along with the assigned UAV path. CPs that are on the way can be influenced and therefore the uncertainty can be reduced.

Previous research has de-emphasized sensing issues. The GUS strategy presents the use of degenerate sensor capabilities to improve the estimation method. The target space derives from unexpected changes that cannot always be predicted by a dynamic model. It is essential to deploy part of the UAVs to validate any deterministic models. The results show that estimation based on the observation adjusts the unobserved state (i.e., spread rate).

The more UAVs are involved in the mission, the more accurate are the estimation results. If the periphery expands and there are not enough UAVs to travel to the CPs and reduce the visiting time, the error increases continuously. Since the obvious approach to monitoring the fire is by following its perimeter, that implementation is examined and compared quantitatively. The deployment setup explored here starts at equidistant points around the perimeter. The initial deployment of UAVs is idealized for the benchmark, and it is expected to worsen in different deployment setups. Future work will elaborate more on different, unequal deployment around the perimeter, which will degrade the performance of the periphery tracking benchmark but does not affect the performance of the GUS strategy.

It is expected that QKF will be the best estimator if the measurements are Gaussian, and if it is not the best, this is likely because the measurements are not Gaussian.

Chapter 3

System Design

3.1 Introduction

For many experimental applications, UAVs can enable or enhance the efforts available to researchers. Much work has been done to make UAVs useful in myriad scenarios. In some scenarios, operating in the environment requires special skills or training that the researchers do not have; here an autonomous system can enable access that was previously difficult to obtain. In recent years, there has been rapidly increasing interest in UAVs where the scientific question or the research project requires an airborne platform.

Technological progress has made it possible to use inexpensive autopilots on small UAVs. The development of high-density batteries, long-range and low-power radios, cheap airframes, high-performance microprocessors, and powerful electrical motors all make experimental research with UAVs more practical than ever [22]. The availability of UAVs as a lab resource allows researchers to explore many new kinds of scientific questions such as control or estimation problems. The flexibility of the system design further allows for quick changes, reducing the project workload.

A modern UAV system consists of an on-board control system (i.e.: auto-pilot) and Ground Control Station (GCS). The autopilot utilizes various sensors, communication modules, a power supply unit, and embedded software to control the UAV. The autopilot software is the real-time implementation of the guidance, navigation and control algorithm; one of the demands on designing a rapid prototype testbed is to enable control algorithms, discussed briefly below.

Autopilots control and guide the UAVs in flight. They rely on data gathered by various sensors and on a central processing unit (CPU), which carries out the instructions of the program. The objective of an autopilot system is to consistently guide the UAVs to follow reference paths or navigate through several waypoints. A UAV autopilot system is a closed-loop control system comprising of two parts: the state observer and the controller. A typical observer is designed to estimate the state (e.g., attitude) from flaure sensor measurements (gyro, accel); advanced control techniques are used in the UAV autopilot systems to guarantee smooth, desirable trajectory navigation.

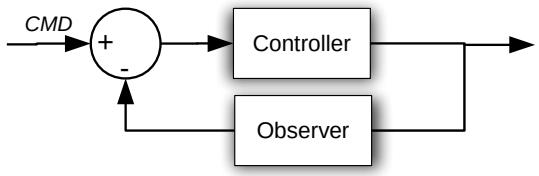


Figure 3.1: Basic closed-loop block diagram.

This chapter focuses on the design of a multi-UAV system that is used in this research and future projects. The emphasis is on the need for Guidance Navigation and Control (GNC) research; the primary objective of this research is to design an efficient process to develop and test new GNC algorithms. This chapter provides a review of the existing autopilot, and the migration process from the previous

successful rapid prototyping concept to a new design; it also includes an analysis of how to migrate to the new SLUGS II design.

Off-the-shelf Autopilots

Many popular autopilots are available on the general market today (Pixhawk, MatrixPilot, Kestrel, Picolo, and Vector to name a few). There are also ones that are government or military and thus are not available to the public for evaluation. A typical commercial off-the-shelf UAV autopilot system comprises a GPS receiver, an IMU (inertial measurement unit), and an onboard processor (state estimator and flight controller). The autopilots can be compared in terms of their sensor configurations, state estimations, and controller performances, where sensors, processors and peripheral circuits are all integrated into a single PCB (Printed circuit board). A complete comparison of the software implementation is not feasible. However, the information gathered from open-source autopilot development is critical to understanding the benefits of an in-house R&D autopilot.

Open-Source Autopilots

The advantage of open-source autopilots are their flexibility in both hardware and software. The open-source autopilot is available to all customers, and experienced users can adapt it to their own needs. Furthermore, due to a large number of contributors, the open source autopilot evolves quickly. Researchers can easily modify the autopilot based on their own special requirements. On the other hand, the quality control of the open source autopilot can be unreliable at best if not consistently maintained.

SLUGS AutoPilot Design

SLUGS (Santa Cruz Low-cost Unmanned Aerial Vehicle Guidance, Navigation & Control System) is a platform that includes autopilot software and hardware components that enable a flexible environment for research in GNC applications [37]. The SLUGS was designed primarily for GNC research, and it has already been used in many flight-tests. It is also part of the experimentation with fixed-wing UAV systems, presented in the following section.

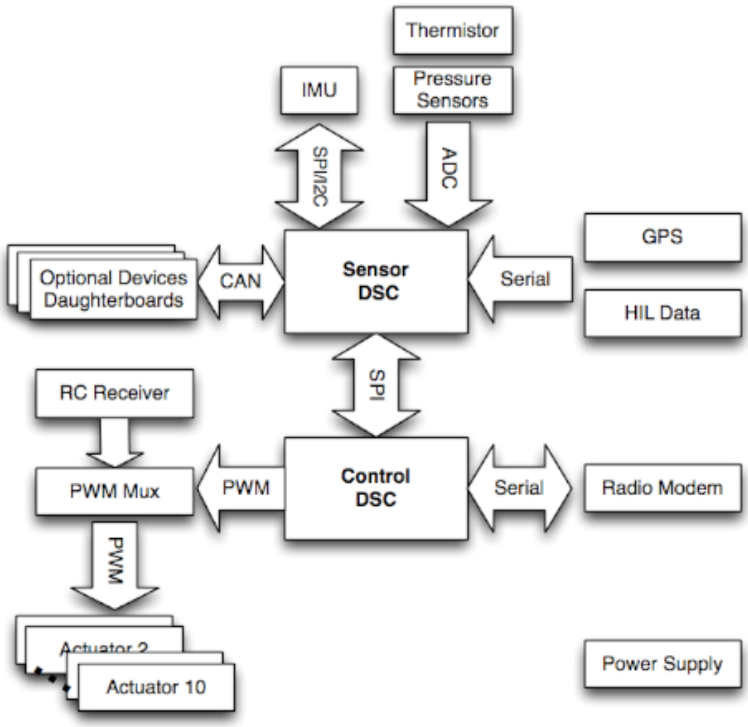


Figure 3.2: SLUGS block diagram is shown. It describes the design of the SLUGS Auto Pilot and all its components. The diagram is imported from the ASL website.

The SLUGS autopilot is an open-source system like several other commercially available autopilots. The key advantage of the SLUGS autopilot over the commercial autopilots is in the easy modifications of its essential components. First, the computations are done with two separate Digital Signal Controllers (DSC),

one for the GNC tasks and one for sensors computations. Second, the software is auto-generated from a Simulink/Matlab integration; the IDE (Integrated development environment) of the GNC algorithms. This create a smooth simulation to flight deployment.

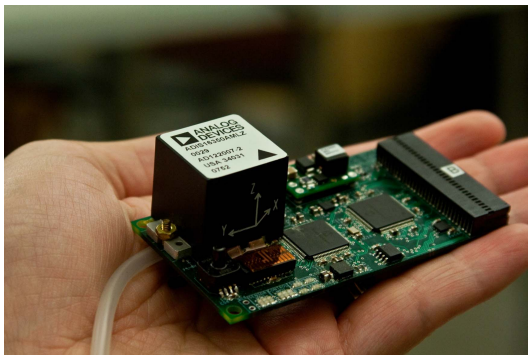


Figure 3.3: SLUGS board

Off-the-shelf Simulators

Although the hardware design has a tremendous benefit in commercial products, it is a burden for R&D autopilots. The first step following the development of new GNC algorithm is to test it on a high-fidelity simulation [40]. Thorough testing for most types of autopilots is handled by running numerous simulations. System simulation involves testing the algorithm with all the components of the system. Most of the components, however, are replaced with models at different levels of fidelity. By simplifying the models, the simulation design is guaranteed to decrease the computational load, while the use of highly accurate models ensures high fidelity.

Fixed-wing UAV

A fixed-wing UAV and rotorcraft both have advantages and disadvantages compare with each other . Fixed-wing UAV tend to be more benign in the air with respect to piloting and technical errors, as they have natural gliding capabilities. Fixed-wing UAV can also carry greater payloads for longer distances on less power.

There are many kinds of fixed-wing UAVs from electric battery powered small foam planes to large-scale wooden replicas with multi liquid fuel engines, and everything in between. An aircraft that suits the mission profile and the research needs for this project is a small electrical foam UAV.

When precision missions are required, fixed-wing aircraft are at a disadvantage, since they must have air moving over their wings to generate lift. They must constantly remain in forward motion, which means that they cannot hover in one spot the way a helicopter or quadrotor can, and as a result cannot provide the same level of precise camera positioning. A fixed-wing, however, is the best choice for longer missions and larger payloads.

3.2 SLUGS II AutoPilot Design

The SLUGS II design improves on previous SLUGS design because it provides rapid prototyping control for multi-UAV systems. The control design process is made up of many iterations that can be verified and validated through both simulation in the Simulink environment and with auto code generation.

3.2.1 Software Design

The complete autopilot algorithm is implemented in Simulink using block diagrams and Matlab toolboxes. Simulink blocks and Matlab routines are effective software that can be used to modify the algorithm and verify the design. Once the model is updated in the Simulink environment, it then generates the new code with the updated features. The R&D work in a model based environment makes

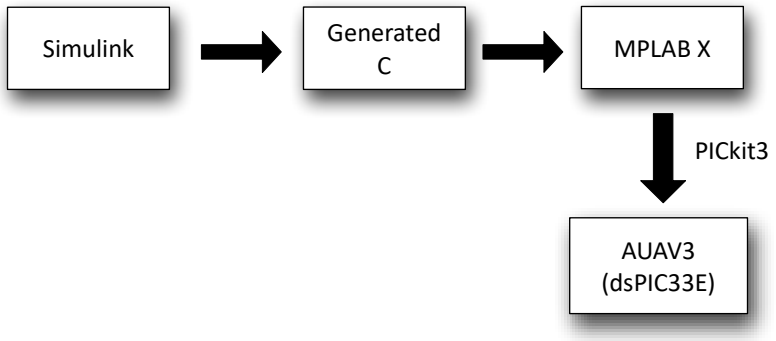


Figure 3.4: SLUGS II code generation workflow.

the programming phase easier. Simulink includes tools that automatically generate and compile the code. The code is then deployed directly to the autopilot hardware [41].

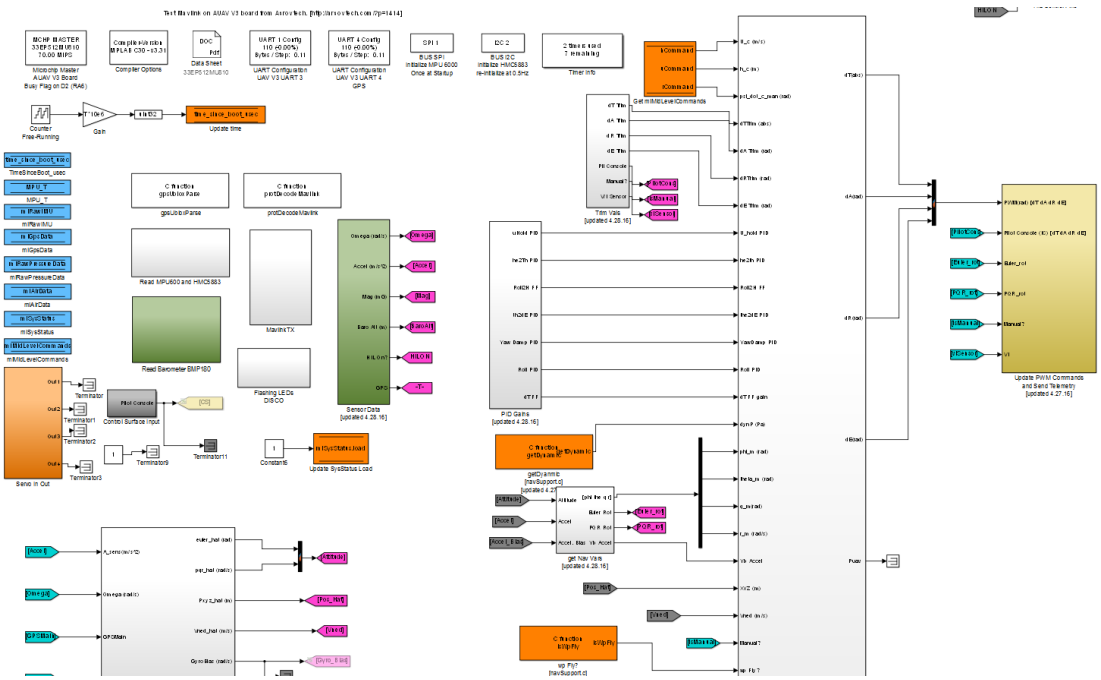


Figure 3.5: SLUGS II Simulink Model is shown. The diagram includes the configuration blocks, the main controller block on the right and the sensor blocks on the left. The block diagram imported from SLUGS II Simulink model development environment.

3.2.2 Hardware Design

The literature on COTS autopilots suggests that the minimum research autopilot requirements are: robustness and attitude accuracy, enough for a low altitude flight surveillance. Hardware must includes sensors on-board and software for an attitude solution [40]. This hardware design makes an important contribution to the research framework because it introduces a new design. The SLUGS embedded system features two Microchip dsPIC33F microcontrollers. That design allows SLUGS to implement more complex and effective GNC algorithms. It provides a high level of safety and fault tolerance features, and it is designed such that the autopilot system would have more than enough processing power. However, it means more maintenance for the research autopilot IDE, and increased cost.

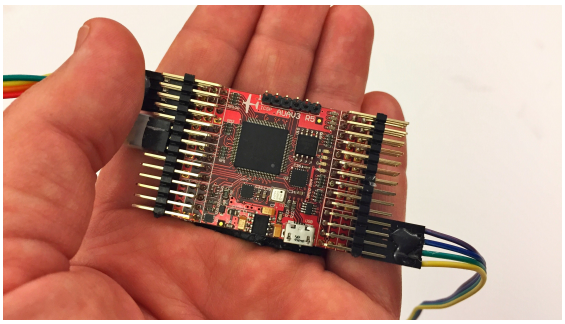


Figure 3.6: AUAV3 board

SLUGS II simplifies the existing design by using on a reliable commercial-off-the-shelf (COTS) hardware. The AUAV3 is a commercial open-hardware development board (all PCB layouts are provided) [6]. It features a single Microchip dsPIC33EP with twice the clock rate of the dsPIC33F. The AUAV3 board (see fig. 3.6) comprises peripheral circuits for IMU, Magnetometer, Barometer and the standard communication interfaces (SPI, CAN, UART and I^2C). Researchers have examined using the AUAV3 to replace the in-house SLUSG hardware. The following sections discuss the SLUGS II Simulink model migration process in more detail, and the steps for verifying and validating the performance with a series of flight-tests.

The design of SLUGS II is such that it can be adapted to various different scales. It provides a solution for multiple UAVs in the testing environment when they are needed for research. The major challenges of research in multi-UAV are handling duplicate systems with low maintenance cost, reliability, and researchers' insufficient skills. The UAV is linked with continually changing technology so that new infrastructure needs to be assessed and adopted in order to improve the existing system. The system migration process can help with this maintenance

by enhancing the new R&D autopilot environment. Furthermore, eliminating the need to maintain multiple platforms can reduce overhead costs, and development difficulties.

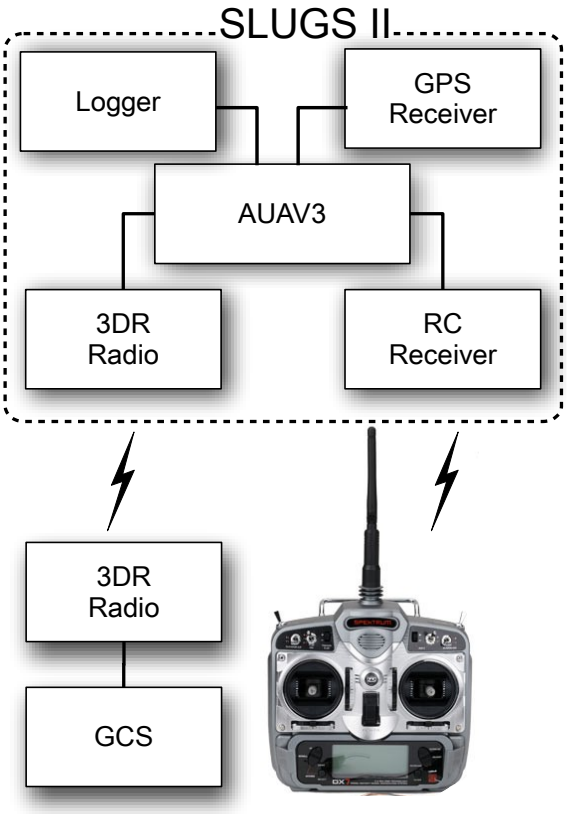


Figure 3.7: SLUGS II basic components.

3.2.3 Migration Process

The AUAV3 addresses the issue of the skills needed to develop or maintain in-house hardware. Commercial hardware is constantly being updated, and for the

R&D autopilot, this is an opportunity to put all the efforts into developing GNC algorithms and utilizing low-cost COTS hardware. The old hardware is difficult to integrate with newer sensors and sensing technology. Complex applications require a flexible and adaptive R&D autopilot to keep up with a dynamic environment.

The simulator integrated within the Simulink development model is another challenging component. It needs specialized skills to tune and adjust to different platforms. Porting the X-Plane simulator improves the development effort and further reduces costs. Different airplane models can be found on the local simulator database instead of tuning the aerodynamic coefficients of a six-degree-of-freedom (6DOF) model by hand.

Two components are migrated as part of SLUGS II design. The benchmark configuration takes the MatrixPilot open-source autopilot and deploys the code on the AUAV3 board. Performance benchmarking ensures that the migrated components perform as well as or better than the old components. The new configuration is then evaluated in multi-UAV software in the loop (MSIL) simulation and in real flight-tests.

Once the assessment of the AUAV3 board is completed, the Simulink model is then modified. The model adjusts to the new dsPic configuration. This integration phase includes eliminating the blocks that handle communication between the separate processors, improving the modeling style, optimization, removing dead code, and identifying incompatible porting issues. Configuring the Simulink model to the new AUAV3 board is based on the Microchip dsPIC toolbox (a new revision of the Lubins Blockset [52]). Although the complete process requires significant manual work, the main intellectual property (IP) of the R&D autopilot remains almost untouched.

In the final phase, the newly migrated autopilot is subjected to rigorous testing

using test cases applied on the original design (SLUGS) and MatrixPilot [62]. Apart from the functional load testing, testing is carried out to ensure that the necessary performance level is achieved. The migrated auto-generated code is deployed, and parameters are fine tuned for the new airframe (BixlerII).

3.3 Ground Control Station

The Ground Control Station (GCS) is one of the most important components in a UAV system. It provides an operational interface to monitor and control the assigned task to the multiple UAVs. It presents any additional information that does not require the autopilot to complete its task, however it supports the user who monitors the mission to coordinate with other systems for better decision making. The GCS includes indications for the mission showing the relevant spatial data (i.e., geodetic coordinates) associated with the map of the area of interest.

The GCS communicate with the UAVs using a bi-directional data link (X-Bees transceivers). It runs on a mobile laptop computer that can easily be transported to the test site.

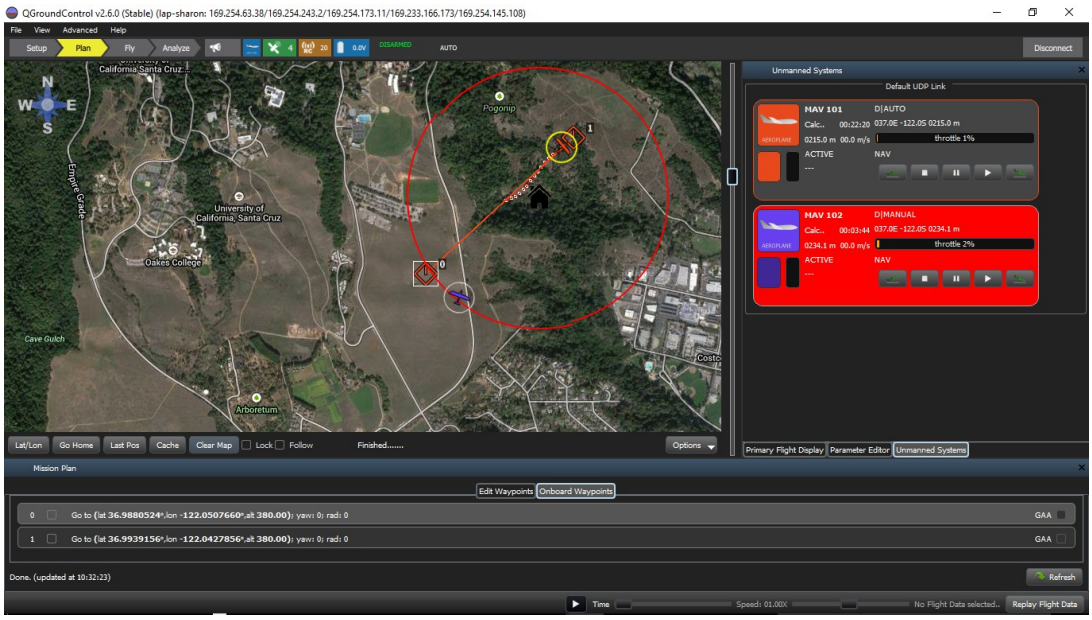


Figure 3.8: The graphical user interface of the Ground Control Station (GCS) is presented. The open-source software (Qt-Ground-Control - QGC) is adopted and extended to support the design of a multi-UAV monitoring system. The software supports the planning and visualization of the UAVs‘ trajectories in real-time.

3.4 Conclusions

Autonomous control is being increasingly employed on UAVs to support research application. The SLUGS autopilot IDE provides a flexible and rapid prototyping environment for UAV GNC algorithms [37]. With series of iterative modifications, driven both by control theory and the results from field tests. The SLUGS II demonstrates the effectiveness of the development process [28]. The hardware implemented for CPU redundancy provides a high level of safety and fault tolerance features, and is designed so the autopilot system can have sufficient

processing power. The downside, however, is that it adds a large maintenance burden for the research autopilot IDE. Supporting multi-UAV configuration makes it necessary that SLUGS II simplify the old design based on different COTS and expanded IDE functionalities. SLUGS II modifies the design process to add a verification step for the generated code in a flexible and friendly environment that is committed to the sequence of events in the software rather than to guarantee strong real-time performance execution of the code. The design validation is discussed in details in 5.

Chapter 4

Simulation

4.1 Introduction

Simulation allows for the safe testing of experimental hardware, software, and trying parameters. Crashing a virtual UAVs requires only a reset of the simulator, in contrast to the expense of crashing (and rebuilding) a real one. Applications that are capable of simulating multi-UAV are typically developed and utilized in-house; very few commercial products are availabble that can simulate multi-UAV, in real-time [53]. Developing UAV models (e.g., aerodynamic and dynamics models) and environment models (e.g., wind, atmosphere, etc.) is a time-consuming task that requires domain specific expertise. Off-the-shelf simulators are designed to use various types of models and provide a database of models.

This chapter addresses the challenges in verifying and validating a new simulation approach, specifically when developing a new control algorithm for UAVs' guidance.

4.1.1 Propagation Model

Developing a propagation model for known dynamics allows for prediction and to upcoming event locations. Some studies show the benefit of considering the uncertain environment during mission planning and how to formulate a stochastic process as part of the overall UAV coordination plan. [11] presents the decomposition method for solving the UAV coordination and control problem. The algorithm is implemented in an approximate decomposition approach that uses straight-line paths to estimate the time-of-flight and risk for each mission. They then show how to extend that formulation to capture the stochastic effects of an uncertain environment.

Significant research activity exists and modeling undertaking wildfire scenarios. For example, [84] develops model of fire-front propagation and [61] develops a wildland fire model.

4.1.2 Multi-UAV Simulation

The unmanned autonomous vehicle (UAV) is a compelling research area because it can dramatically lower the cost of the experiment phase of a research project; this is especially true where the tests are dull or dangerous to humans. Sometimes the research requires having special skills for working in that environment that the researcher might lack, and so replace this experimental phase with simulation. In recent years, there has been a rapidly increasing interest in the UAV in work where the scientific question or research project requires an airborne platform.

The first step following the development of new GNC algorithm for a UAV is to test it in a system simulation [40]. Thorough testing for most types of autopilots is handled initially by running simulations. System simulation involves testing

the algorithm with all the components of the system. Most of the components, however, are replaced with models it differing levels of fidelity. By simplifying the models, the simulation design is guaranteed to decrease the computational load, while the use of highly accurate models ensures high fidelity.

Applications that are capable of simulating multi-UAV are typically developed and deployed in-house. Very few of them are commercial products that can potentially simulate multi-UAV in real-time [53]. Developing UAV models (e.g., aerodynamic and dynamics models) and environment models (e.g., wind, atmosphere, etc.) is a time-consuming task that requires domain specific expertise. Off-the-shelf simulators are designed to use various types of models and provide a database of developed and validated models.

This chapter addresses the challenges in developing a new control concept for UAV, specifically verifying and validating the new approach. In a real-time environment with real-time dynamical conditions, the difficulty becomes even greater. The process is very tedious if it is done by reprogramming the autopilot and following the conventional software development process. It can be faster and more efficient if the integrated development environment (IDE) uses the latest methods and control theory, especially if the IDE supports automatic code generation and rigorous code verification.

4.1.3 Verification and Validation

Verification and Validation are two independent procedures that are used in system development to check that the system meets the requirements. Verification focuses on the software specifications, and validation ensures that the system meets the operational needs (e.g., real-time, frequency, etc.). Most generated code fails at the interface between the generated modules. In fact, system integration (handling the interface between subsystems) is often a very difficult task in human-generated code. Unfortunately, that cannot be detected in the Simulink environment, since Simulink does not execute the generated code.

A complete process that supports a multi-UAV configuration is needed to be considered by the autopilot system for real-time identification and task allocation. To support a multi-UAV configuration, the SLUGS II design extended the tools for software verification. The multi-UAV IDE offers code verification with a complete software in the loop (SIL) simulation.

4.2 Simplified Simulation

The simplified simulation main goal is to support the development of coordination algorithm. Designing the coordination scheme and the UAV guidance laws fully depends on the simulation fidelity. As long as the simulation represents all the physical processes details of the system it will expose the pitfalls of the proposed solution beforehand. If the design is focused on the control part of the airplane model it should be adequate for developing a dynamic model [10]. The dynamic model is derived from Newton's laws and defines the relation between forces and moments. In this simulation a steady state flight and leveled flight is assumed, therefore the focus is on the relationship between position and velocity

(Kinematics, see Fig. 4.1).

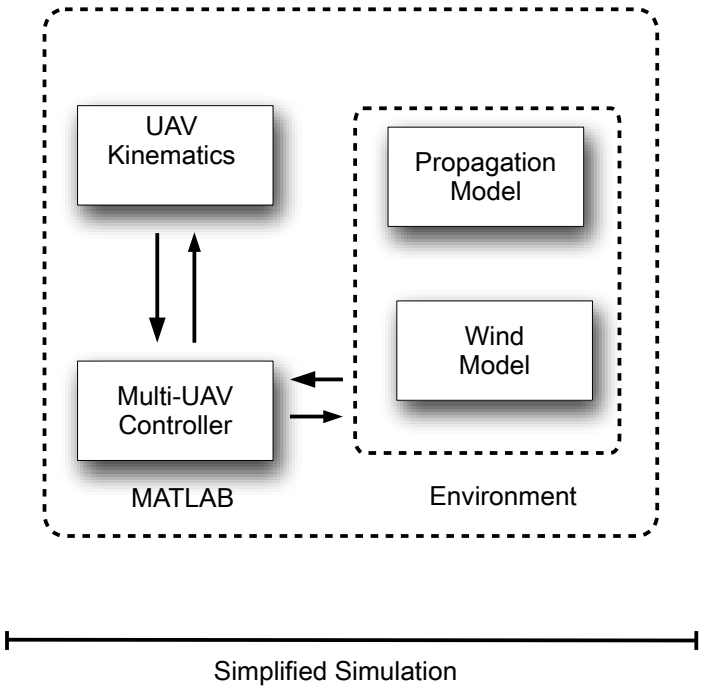


Figure 4.1: Simplified Simulation block diagram.

The simplified simulation introduces the first order constraints of a dynamical system where the vehicles are mobile and the environment domain changes. Two types of models are included: the physical models and the discrete models. Both type are software oriented implementations in which the execution processes guarantees to reconstruct the outcome, hence, the process is deterministic.

The physical models simulate continuous processes. A continuous process can model the uncertainties of sensors, actuators, and the working environment. Sensors are involved in the process by measuring the parts of state with associated noise. The implemented sensors modules are responsible for simulating the detected state variables and to introduce the additional (modelled) uncertainty in

to the observations.

UAV Model

The UAV model in the simulation is a mathematical representation of the actual motion of non-holonomic systems. The following equations show the translational kinematics. Six states (p_n, p_e, p_d, u, v, w) out of a total of 12 states in a complex flight model. The UAV is assumed to hold constant altitude and the wind is planar:

$$\begin{pmatrix} \dot{p}_n \\ \dot{p}_e \\ \dot{h} \end{pmatrix} = V_a \begin{pmatrix} \cos\psi \\ \sin\psi \\ 0 \end{pmatrix} + \begin{pmatrix} w_n \\ w_e \\ 0 \end{pmatrix} \tag{4.1}$$

where $\dot{p}_n, \dot{p}_e, \dot{h}$ are the inertial velocity vector, V_a is the airspeed, w_n, w_e are components of the planar wind velocity and ψ is the azimuth angle.

Figures 4.2, 4.3 show the feasible trajectories in the development environment. The UAV switched between two configurations whenever it planned to move from one waypoint to other waypoint.

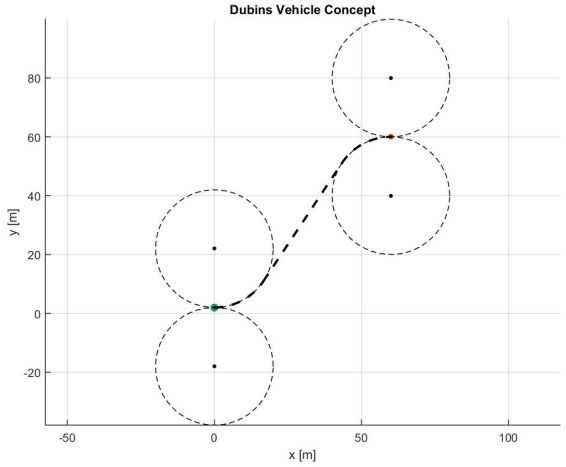


Figure 4.2: An example of Dubins Path is shown. The start point (in green) is attached to two circles tangent to the desired direction. The final point (in red) is attached to two circles tangent to the desired direction. The resulting time-optimal path (in bold) starts with a Left turn followed by a Straight line and finishing with a Right turn to the final configuration.

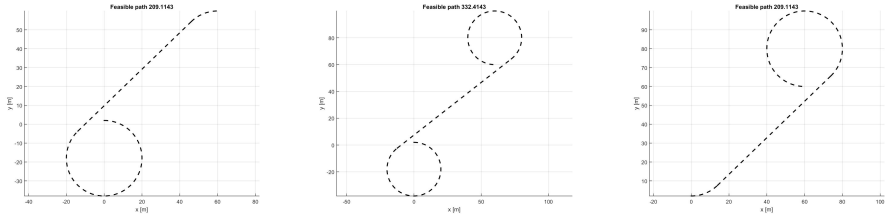


Figure 4.3: Three admissible paths of a selected example of Dubins Path are shown. Each path starts at the selected start configuration and finishes at the desired configuration. The length of the path is calculated and presented at the top of each figure. The shortest path is shown on the previous figure.

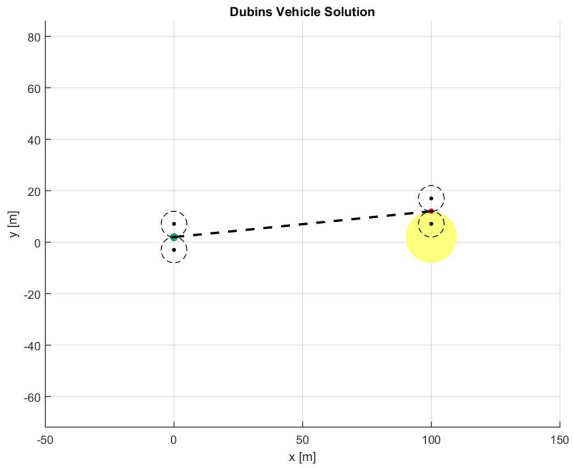


Figure 4.4: An example scenario resulting from Dubins Path is shown. The start point (in green) is attached to two circles tangent to the desired direction. The final point (in red) is attached to two circles tangent to the desired direction and the AOI (in yellow). The resulting time-optimal path (in bold) starts with a short left turn, continue with a straight line, and finishes with a short right turn to the final configuration.

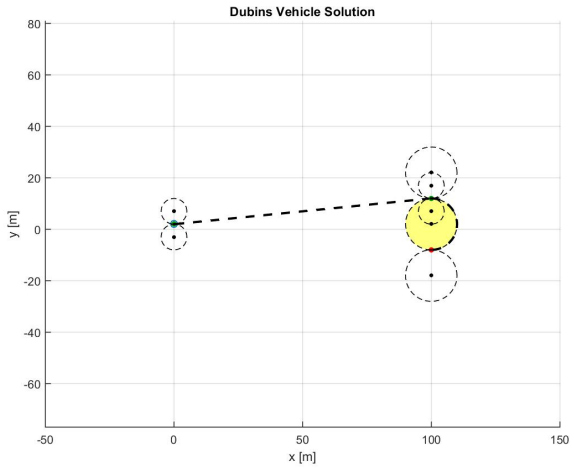


Figure 4.5: A follow up step of an example scenario is shown. The start point (in green) is attached to two circles tangent to the desired direction. The final point (in red) is attached to two circles tangent to the desired direction and to the AOI (in yellow). The resulting time-optimal path (in bold) includes only a right turn leading to the final pointing and orientation.

4.2.1 Propagation Model

The propagation model helps anticipate and simulate the boundary location in time. The developed model represents the fire-front propagation of a wildland fire. Although the adopted modeling approach is a simple representation it considers two factors that been known to be most influenced on the spread rate. It assumes that the local spread at any given point on the perimeter is perpendicular to the fire perimeter into the unburned environment and that the fire has a local rate of spread (ROS) normal to the fire-line.

The implementation of this model is a greedy evaluation; each point out of a set of grid points along the periphery is evaluated. Wind velocity and slope are incorporated into the propagation model. The model combines the wind velocity directly to each grid point of the boundary. The slope is relative factor which scales from 0 to 1 where 0 is the case where the terrain has no-slope and 0.3 is for 30% slope of the terrain. Fig. 4.6 demonstrates the effect of wind, terrain slope and the combination of both in propagation. Wind sets the dominant direction of spread. If the wind and the terrain start in a gully (i.e.: upslope) then combined factors causes the fire to spread even faster.

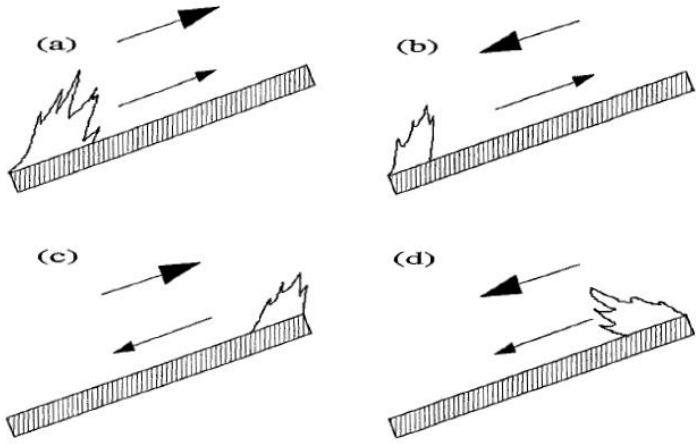


Figure 4.6: Various combinations of wind velocity and slope: (a) upslope heading fire, (b) upslope backing fire, (c) downslope backing fire, (d) downslope heading fire. Small arrows indicate direction of fire spread, large arrows indicate wind direction. Figure adopted from [96]

The propagation model employs the wind effect on each grid point (i) of the discritized boundary:

$$\begin{pmatrix} \dot{p}_N^i \\ \dot{p}_E^i \end{pmatrix} = V_{SR} \begin{pmatrix} \cos\psi^i \\ \sin\psi^i \end{pmatrix} + \begin{pmatrix} V_{W_N} \\ V_{W_E} \end{pmatrix} \tag{4.2}$$

where, V_{SR} is the nominal spread rate, V_W is the wind velocity vector and ψ is the heading angle of the grid point compare to the origin of the fire.

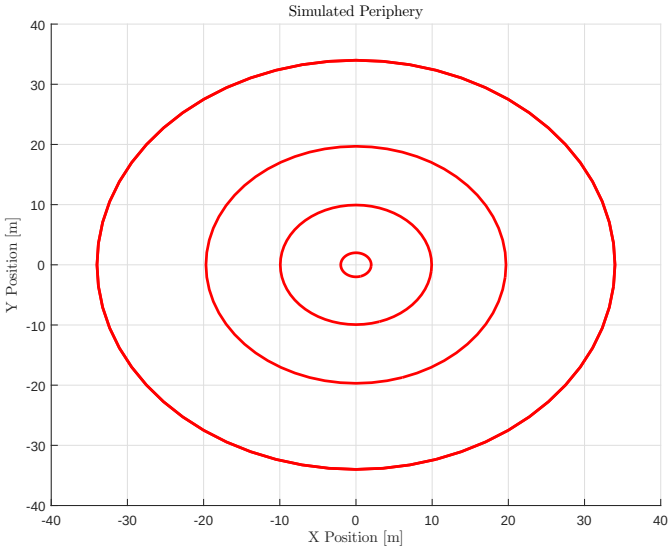


Figure 4.7: Simple scenario of the propagated model outcome is presented. The red circle lines are set periods of time for the simulated boundary. The boundary expend in time with a spread rate of 2 [m/sec]

Fig. 4.7 and 4.8 show examples for 60 grid points to represent the propagated boundary in a 35 second scenario. The first case spreads out equally in time with constant spread rate. The second example includes small wind of 0.1 [m/sec]; the effect is noticeably dominant.

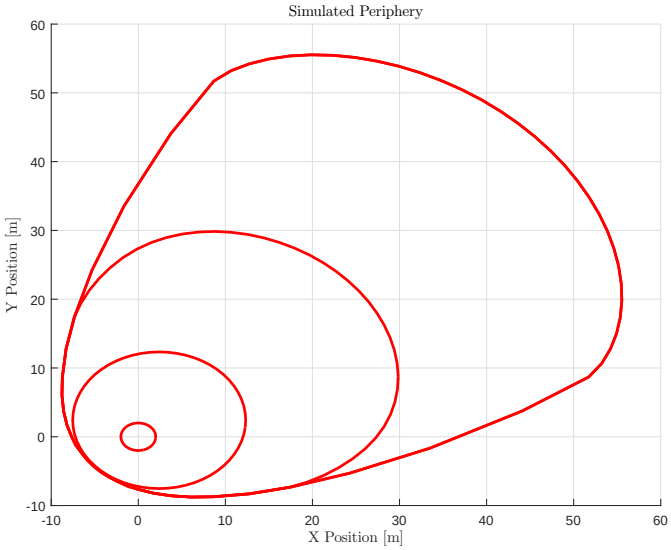


Figure 4.8: Scenario of the propagated model with wind is presented. The red circle lines are set periods of time for the simulated boundary. The boundary expend in time with a spread rate of 2 [m/sec] and a nominal wind velocity of 0.1 [m/sec]

4.3 Multi-UAV Software in the loop

Further simulation for a higher level of fidelity testing during the final steps of developing the coordination scheme. MSIL simulation allows running the SLUGS II research autopilot on a computer before running it on the target processor. It communicates with a simulator for simulating high fidelity flight dynamics (X-Plane). The MSIL simulation is meant to run a single or multi-UAV configuration and support the external interfaces and built-in internal calls (for example, memory, timing and peripheral libraries) of every instance of the SLUGS II autopilot code.

The MSIL software includes the generated code, which is compiled together with a handling layer (a real-time wrapper software). The RT Wrapper interfaces with the external software through a User Datagram Protocol (UDP) socket or a serial port. The MSIL simulation controls the simulated GPS, telemetry and

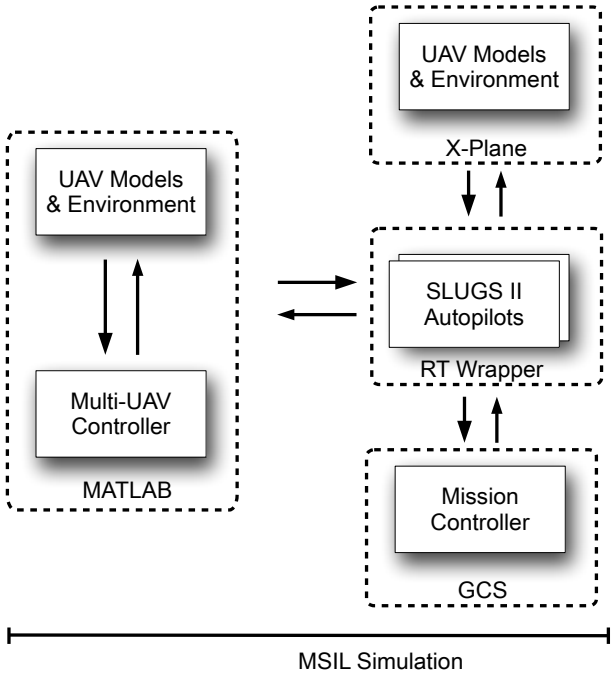


Figure 4.9: MSIL block diagram.

remote-control (RC) inputs for a real RC controller (training mode). The autopilot researcher benefits from the ease of integrating the original generated code and having an easy, friendly environment for debugging.

The GCS unit controls the UAVs through a communication bridge to ensure a two way communication between the GCS and the SLUGS II autopilot. The autopilot can directly managed information from the serial port (or in case of MSIL from the buffer of the serial port). The RT Wrapper (in fig. 4.9) is responsible for managing the buffers and for distributing the MAVlink messages between real UAVs or simulated modules.

The coordination algorithm is executed in Matlab and works as an extension of the GCS. The RT Wrapper creates a tunnel between Matlab and the SLUGS II software through a physical communication link (UDP) using the MAVlink protocol.

4.4 Multi-UAV Hardware in the loop

The Multi-UAV hardware In the Loop (MHIL) simulation runs the SLUGS II software stack on the AUAV3 flight controller using raw sensor data fed in from the simulated environment running on the desktop PC. HIL simulation replaces the UAV and the environment with a simulator (the simulator has a high-fidelity aircraft dynamics model and environment model for wind, turbulence, etc.) The physical autopilot hardware (AUAV3) is configured exactly as for flight, and connects to a computer running the simulator rather than the aircraft. In this sense, the AUAV3 does not know it is flying a simulation.

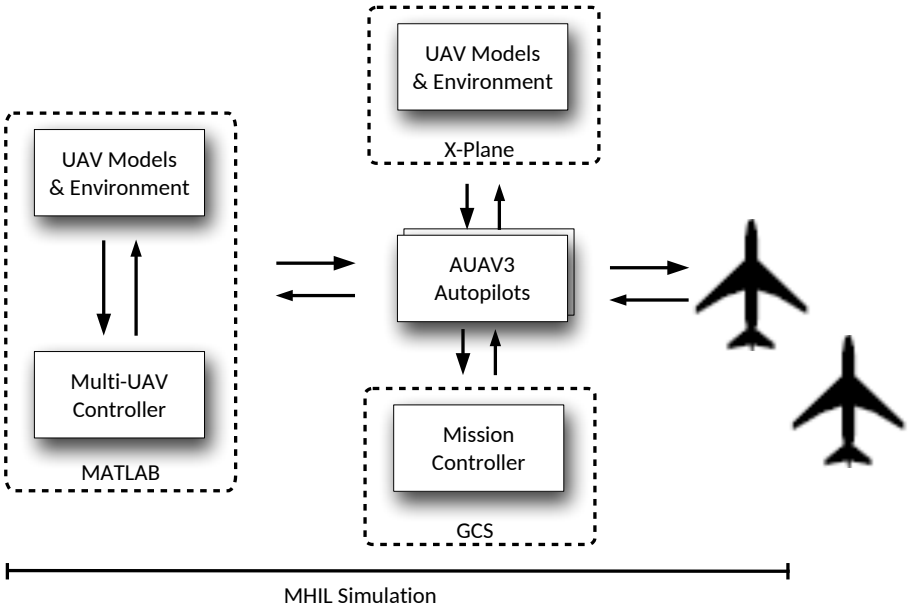


Figure 4.10: MHIL block diagram.

Figure 4.10 shows the MHIL setup. The involved units in the MHIL configuration are depicted along with their associated interfaces. The AUAV3 and the GCS are connected physically by a telemetry link. The autopilot is connected to

a computer running the simulator. The simulator is fed by the servo commands and responds with sensory values from the simulated airplane model. The generated sensor values are similar to the IMU output and injected to the navigation algorithm as the UAV autopilot flies the high fidelity flight situation.

4.4.1 Flight Simulator

X-Plane is a COTS flight simulator. Its framework allows researchers to study UAV control algorithms using realistic UAV models in real-world modeled environments. The simulator database provides a large number of UAV models that are widely available for the R&D autopilot and a rich networking interface that allows it to be interfaced to other software (e.g., GCS). It interfaces with the SLUGS II MSIL software, allowing SLUGS II to simulate flying a wide variety of aircraft.

Using X-Plane with SLUGS II MSIL or MHIL is a good way to validate a flying SLUGS II and evaluate its performance with the use of the GCS. It can also be used to test SLUGS II in unusual situations and to develop support for UAV features not available in other simulators.

4.5 Conclusions

SLUGS II modifies the design process to add a verification step for the generated code in a flexible and friendly environment that is committed to the sequence of events in software rather than to guarantee valid real-time performance execution of the code.

Chapter 5

Results

5.1 Introduction

In the end, all of the various functionalities must work both as individual subsystems, but also integrated as part of the entire system: experiment with the UAV design, the basic multi-UAV flight formation, and the monitoring system. Each one is a step in validating the complete system design which addresses the full multi-UAV monitoring problem.

The system architecture can be utilized in a centralized or a decentralized scheme of operation to enable coordination and information sharing. In a centralized system configuration, the UAVs relay real-time information between each other through the GCS. Alternatively, the UAVs could transmit real-time information between group members (a decentralized scheme configuration).

The platform used for the first flight tests was a Phoenix R/C aircraft; later the platform was changed to a Hobby King Bixler 2. Both of the planes are low-cost foam kits and have a flying weight of approximately 2 lbs. The Phoenix and the Bixler 2 both feature a pusher propeller configuration that reduces vibration and increases overall robustness for a belly landing (nigther aircraft has landing

gear). The wings and fuselage are reinforced with carbon fiber tubes that provide ample rigidity to the airframe [57]. The aircraft is hand launched for take-off.

Figure 5.1: RC model plane - EPO glider phoenix 1370mm.

The Bixler 2 wings are almost an elliptical platform with curved winglets for increased flight efficiency. The power plant for the Bixler 2 aircraft is a 1200 kV brushless DC electric motor. The power source used is a 2200mAh Lithium Polymer battery. This battery provides sufficient current for the electric motor, servo, and the AUAV3 autopilot board, through the Electronic Speed Controller (ESC). The ESC provides a 5.0 volt supply to the servos and the AUAV3 autopilot through the Battery Eliminator Circuit (BEC), and also provides a control signal and power to the brushless motor. The BEC is designed to keep servos R/C receiver running while the baking has dropped too far in voltage to power the motor.

Figure 5.2: RC model plane - Hobby King Bixler 2.

5.2 Simulation Results

The SLUGS II autopilot, like most other autopilots, uses a Proportional-Integral-Derivative (PID) control method for the low-level control loops [37]. The flight controller is developed as a Simulink model, and although it is relatively easy to alter its structure, it requires extensive knowledge about the inner and outer loop structure to redesign the controls. The simulation tests were devoted to validating the viability of the flight controller as flyable. This part of the testing covers the tuning process of the PID gains for the various autopilot control loops.

5.2.1 SLUGS II Validation

The goal of the SLUGS II validation is to support the R&D monitoring system development. The validation relies on several factors, including flight controller and path following performances. The flight controller has been extensively tested within the simulation. The environment supports parameter tuning which can accommodate hardware changes and flight mode extensions.

The most important feature of the SLUGS II autopilot for the R&D monitoring system is its autonomous waypoint navigation capabilities. The ground operator, through the GCS interface, can specify a sequence of waypoints to define the path the vehicle should follow. Fig. 5.3 describes an example of a running scenario with four waypoints and shows how the vehicle follows the desired path while tuning PID gain parameters.

Fig. 5.3 shows that initially, the gains were too low, and the system had a slow response.

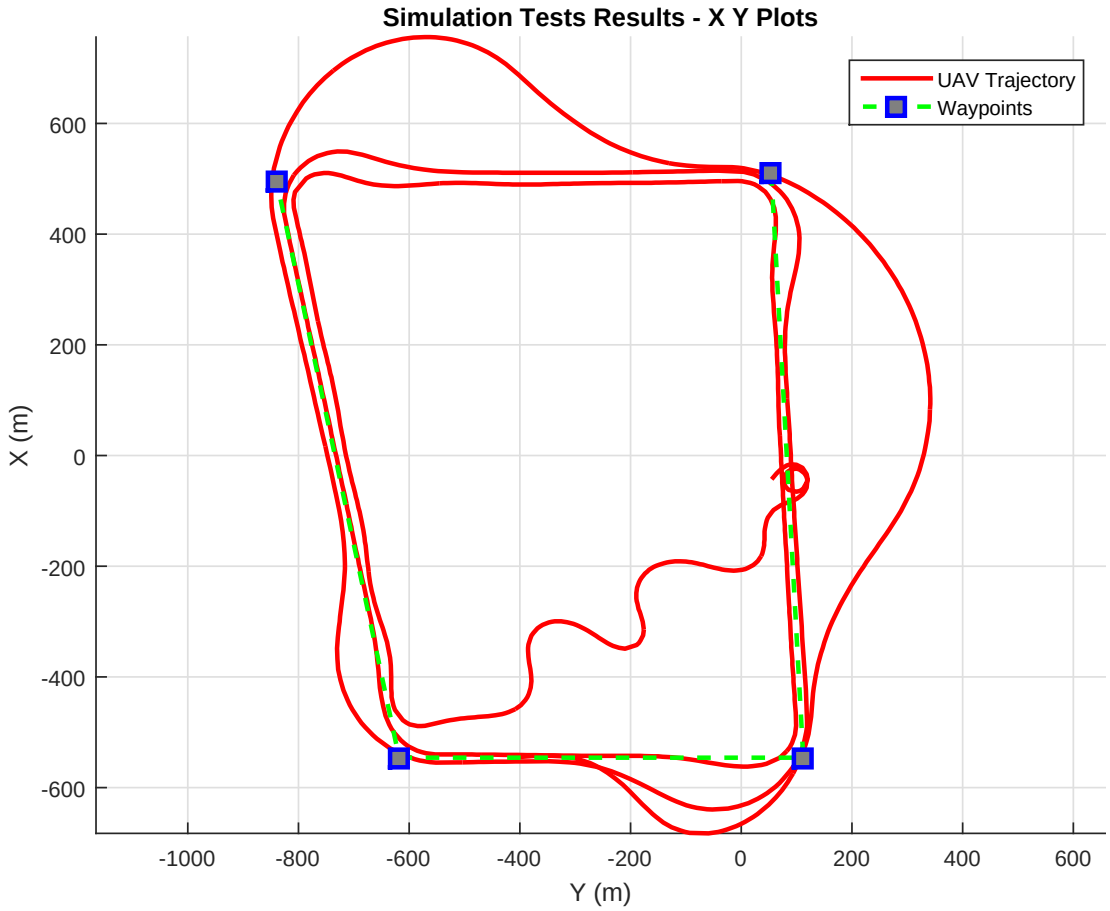


Figure 5.3: Simulated scenario with a single UAV is presented. The UAV trajectory is in X Y Cartesian coordinate frame and is relative to the Home position. The first segment of the trajectory started from take-off controlled manually by the safety-pilot (RC) and switched to autonomous mode after 23 seconds. Three laps were tested with different PID gain for tuning the roll command.

5.2.2 Monitoring System Validation

The nominal scenario to operate the monitoring system starts with deploying a fleet of UAVs. The GCS has been utilized for planning the paths for the deployment phase based on the deployment scheme. The simple deployment scheme includes a fixed heading path mission plan. That path allows the sensor to detect the boundary crossing and switch to the next phase of the coordination policy:

the autonomous allocation phase.

Two UAVs deployed on a preplanned mission (a transect line in fig. 5.4, 5.5):

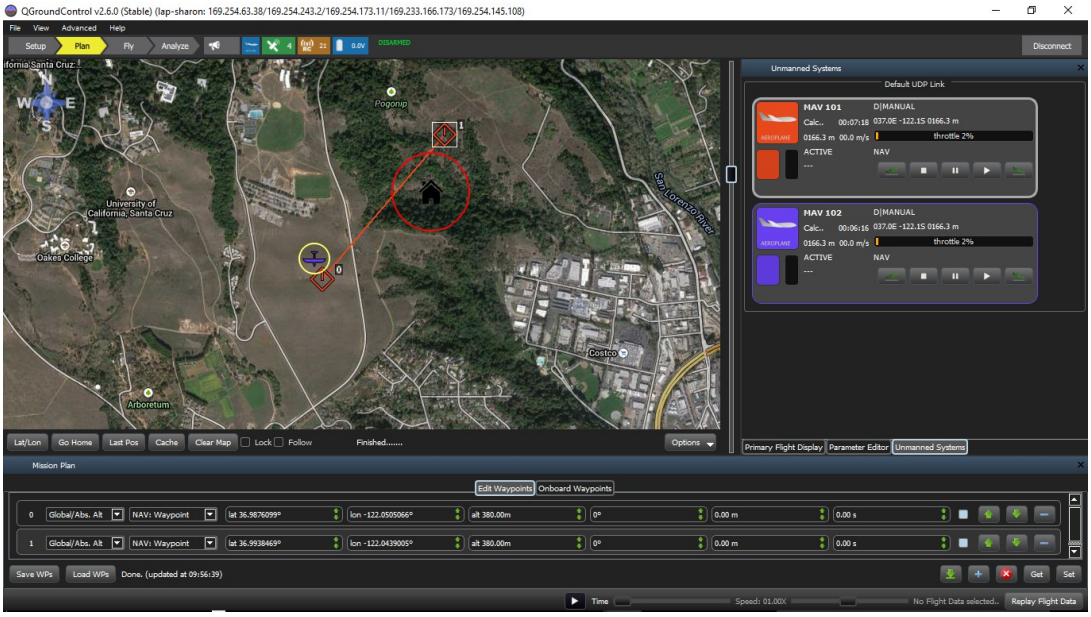


Figure 5.4: Graphic user interface (GUI) of the GCS is presented. The first mission plan for the first UAV is being uploaded.

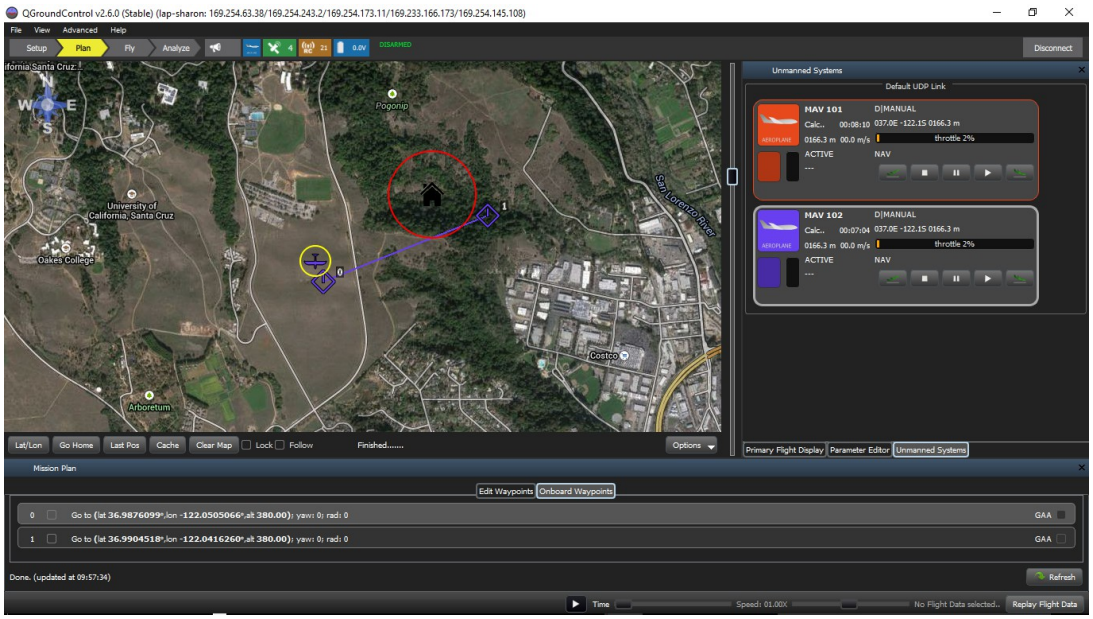


Figure 5.5: The GUI of the GCS is presented. The first mission plan for the first UAV is being uploaded.

The dynamics of the UAVs is simulated in the MSIL configuration by the X-Plane simulator and thus closer to real dynamics of the vehicles. Fig. 5.6 shows the GCS during a simulated flight. At the same time, the animation of the simulation can be visualized in X-Plane while the UAVs follow preplanned trajectories.

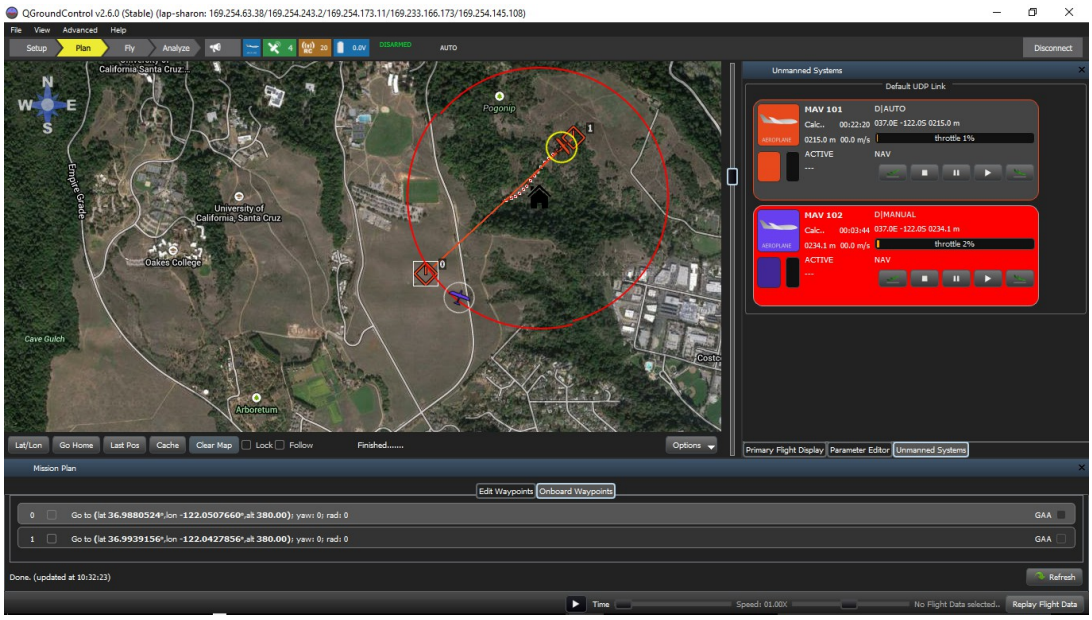


Figure 5.6: The GUI of the GCS shows the execution of a simple deployment mission plan. The deployment phase has started and UAVs are in flight.

The dynamics of the UAVs is simulated in the MSIL configuration by the X-Plane simulator, hence closer to the real dynamics of the flight vehicles. Fig. 5.7 demonstrates the performance during deployment of the UAVs. The estimator works in the background and improves the predicted periphery while the UAVs follow fixed heading paths.

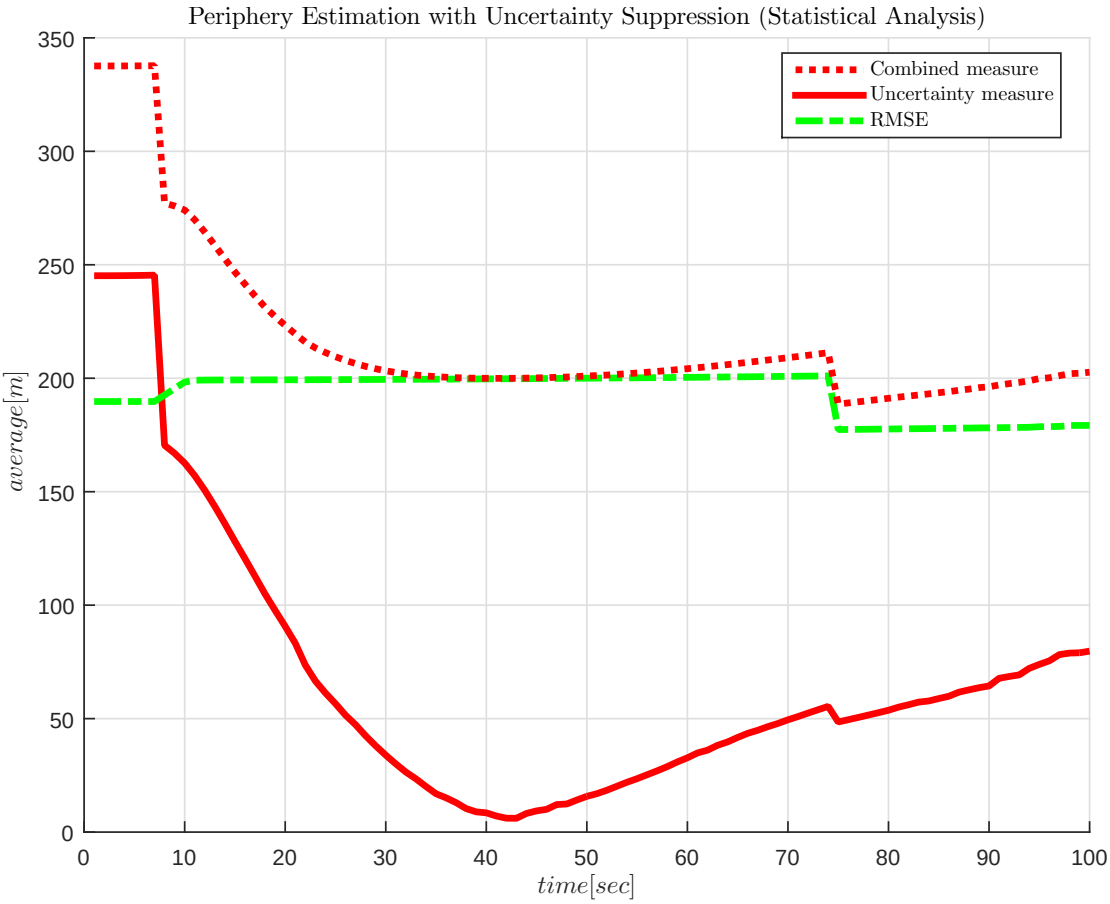


Figure 5.7: Performance measure of the deployment is presented. The combined measure shows that the uncertainty does get reduced during deployment before switching to autonomous allocation mode.

5.3 Flight Tests Results

This section is a brief description of a very long process of redesigning the R&D autopilot framework. The SLUGS II autopilot (a new design) is being employed in both a simulated environment and a real-time environment. The development process includes many experiments to verify that the new hardware and software design keeps the waypoint-following performance and can achieve the requirements of the multi-UAV monitoring system.

The main objective of the flight tests was to validate the development environment workflow: MSIL, MHIL and flight test. The AUAV3 board was the required first step for hardware validation. As the AUAV3 is a new hardware design, requiring it to work is a very significant step for validating the full configuration. The AUAV3 has been tested with an open-source autopilot (MatrixPilot). Next, the SLUGS II autopilot code was deployed to the AUAV3 with adjustments to the new hardware configuration. With that setup, the flight test was simulated in MSIL and MHIL configuration where the actual flight code is the generated code from the SLUGS II autopilot Simulink model.

Presented here are selected field tests result from more than twelve field tests.

5.3.1 AUAV3 Flight Test I

The first field-test was conducted to examine the real-time performance of different levels of autonomous control with the AUAV3 board. To have a consistent benchmark, the same mission plan for a single UAV was uploaded. The mission plan included four way-points and was executed by the MatrixPilot autopilot software whenever the autopilot is switched to stabilized or autonomous mode.

Fig. 5.8 shows snapshots from the GCS while performing the flight. The UAV (Bixler2) executed the mission plan and followed the given way-points on the

resulting trajectory from the guidance algorithm in the MatrixPilot software.

Figure 5.8: Single UAV flight performing mission plan (as presented on the GCS)

Because multi-UAV systems are complicated to debug during a development process and a cooperative mission is hard to analyze after executing a test, the testing procedure includes a critical step through the developed MHIL simulation.

Figure 5.9: MHIL test setup is shown. On the left, QGC software runs on a separate PC. In the middle, X-Plan simulator software runs on a separate PC and communicates with the QGC software and AUAV3 through a serial link. On the right AUAV3 runs the test autopilot software version and is connected through a serial port to the PC.

Fig. 5.9 shows the integrated development environment used for MHIL. The MHIL executes the full configuration of the UAVs' hardware without actually flying the platforms. All the hardware components (AUAV3 board[6], servos, power supply, telemetry and radio control) in this specific configuration have been tested. In MHIL mode, the X-Plane simulator (which provides the dynamic environment of the airplane model) is utilized. The physical parameters (e.g.: position and attitude) output are sent from X-Plane to another software module,

which translates and injects them into the real-time autopilot software under test (MatrixPilot[62]).

5.3.2 AUAV3 Flight Test II

Two flight tests were flown to examine the features of the path planning tool developed as part of the mission coverage described in [31]. The test was to verify the use of the algorithm for both a point-to-point tour scenario and an area coverage scenario. The test was also utilized for validating the new setup. The vehicle used for these experiments was the Phoenix fixed-wing UAV equipped with an AUAV3 flight controller running the MatrixPilot autopilot. Both flights consisted of the aircraft flying multiple passes through a series of waypoints [31].

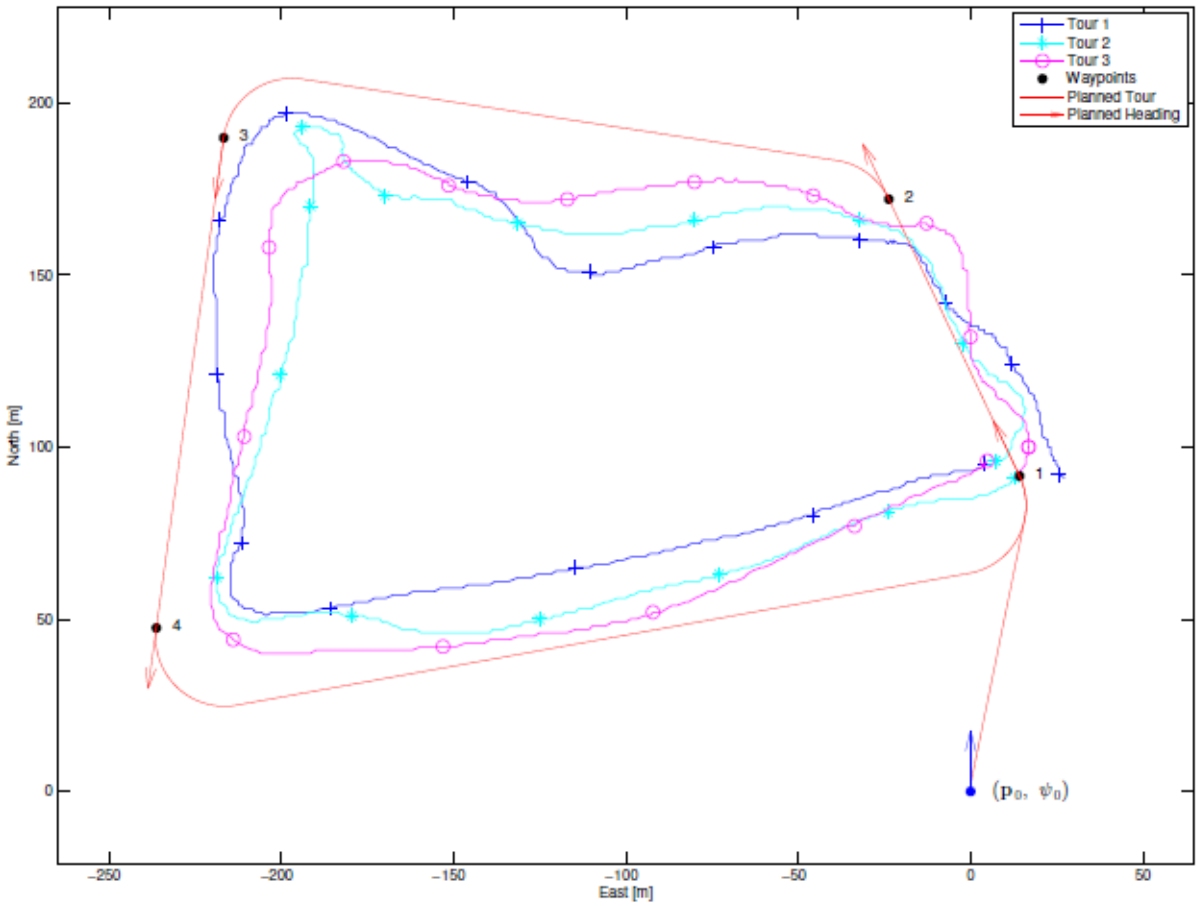


Figure 5.10: A test flight conducted with Phoenix fixed-wing UAV equipped with an AUAV3 flight controller running the MatrixPilot autopilot [31]

The AUAV3 based MatrixPilot directly processes the pilot‘s control commands from the radio receiver and sends out servo signal commands, using the built-in pulse-width modulation (PWM) features of the dsPIC33F microcontroller. The

AUAV3 module also hosts peripheral components that combine the MPU6050 gyroscope and accelerometer, a HMC5883L magnetometer and a BMP-180 barometer. The AUAV3 module hosts a custom add-on SD card module (SLOGGER) used as a data-logging device.

Figure 5.10 shows the UAV's flight path. The autopilot struggled to track the trajectory. The path became oscillatory in-between waypoints. There was a difference between traversing the downwind leg or when traveling upwind. These effects are mainly caused by the wind and are accounted for in the SLUGS II autopilot (based on the previous implementation). It is clear from the performance that MatrixPilot does not compensate for ground-speed changes.

The resulting path is a measure of the autopilot's guidance performance and of how well the autopilot can follow waypoints. These results are important because they show how a real aircraft with an AUAV3 autopilot performs in real-time and that its onboard sensors provide enough information to stabilize and control the aircraft.

5.3.3 SLUGS II Flight Tests

The goal of the SLUGS II flight tests was to achieve acceptable stability and quick response time. The SLUGS II R&D autopilot has been designed so that it can steer the UAV to a destination and follow a trajectory defined by multiple waypoints.

The SLUGS II parameters were known for the simulation configuration and were tuned to give good simulated flight properties. However, in a real flight test configuration, these parameters need to be modified for stable flight. It is essential to re-tune the PID gain parameters for any change in the system hardware. The tuning procedure involves setting the UAV into autonomous mode.

The UAV executes the mission consisting of multiple waypoints. The path is repeated continuously, and the UAV returns to the first waypoint after visiting the last waypoint. During the flight tests, the PID gains have been changed to examine the autopilot control response.

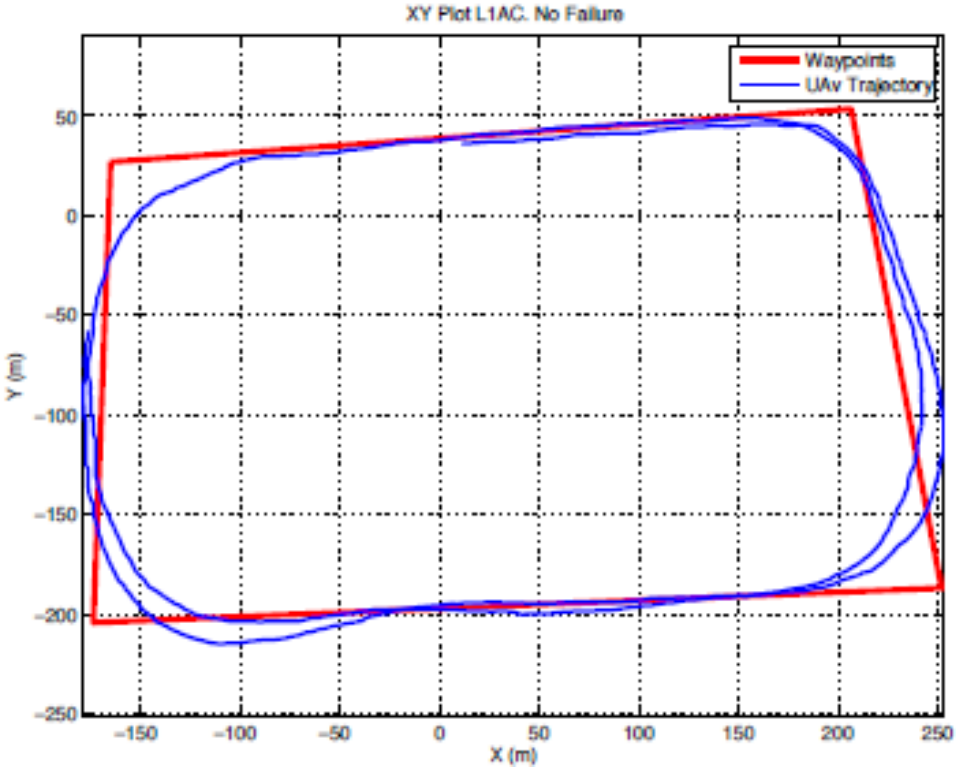


Figure 5.11: Flight test with SLUGS autopilot configuration is presented. Example of performance for flight test 1 from[37]

Figure 5.11 shows a mission plan with four waypoints executed by the SLUGS autopilot with the Mentor airplane as a benchmark for the new revision of the autopilot.

Figure 5.12 shows two laps executed by the UAV after completing the tuning procedure for the SLUGS II R&D autopilot.

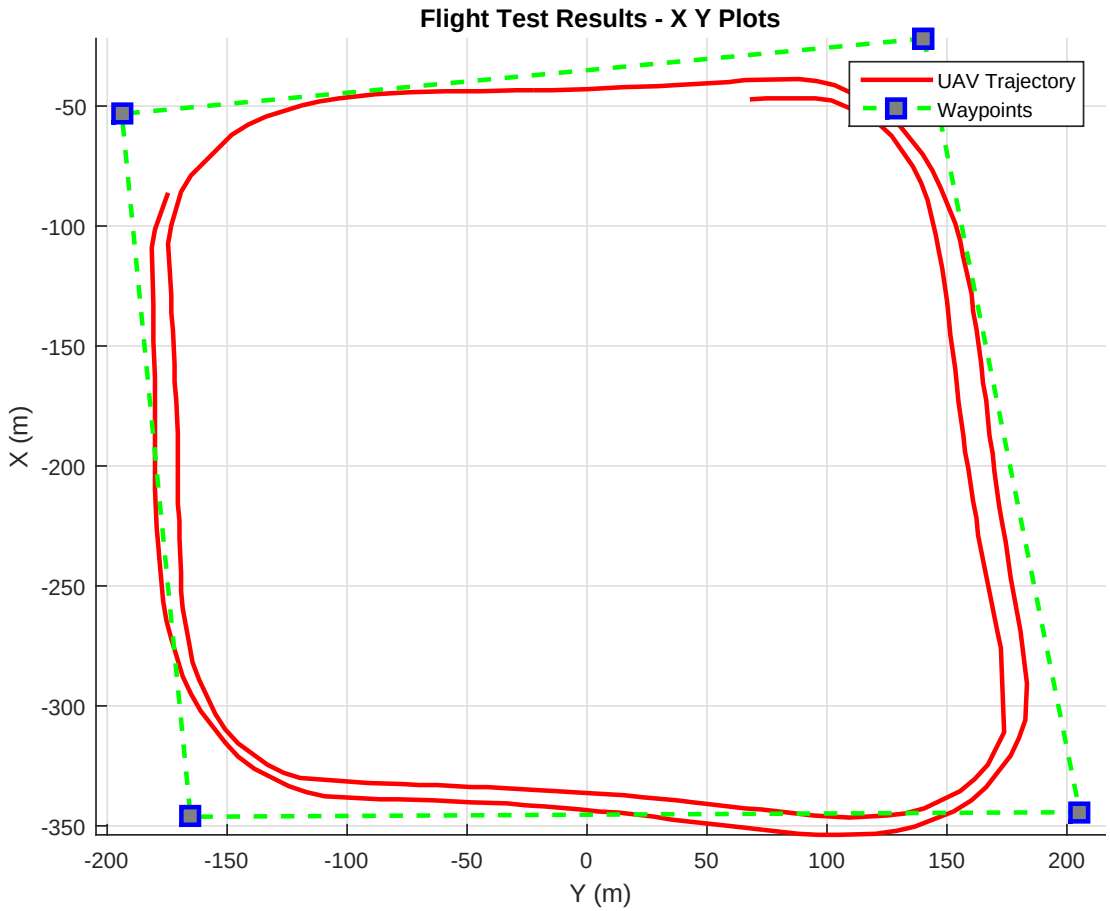


Figure 5.12: Flight test with a single UAV is presented. The UAV trajectory is in X Y Cartesian coordinate frame and is relative to the home position $(36.989^o, -122.0514^o)$.

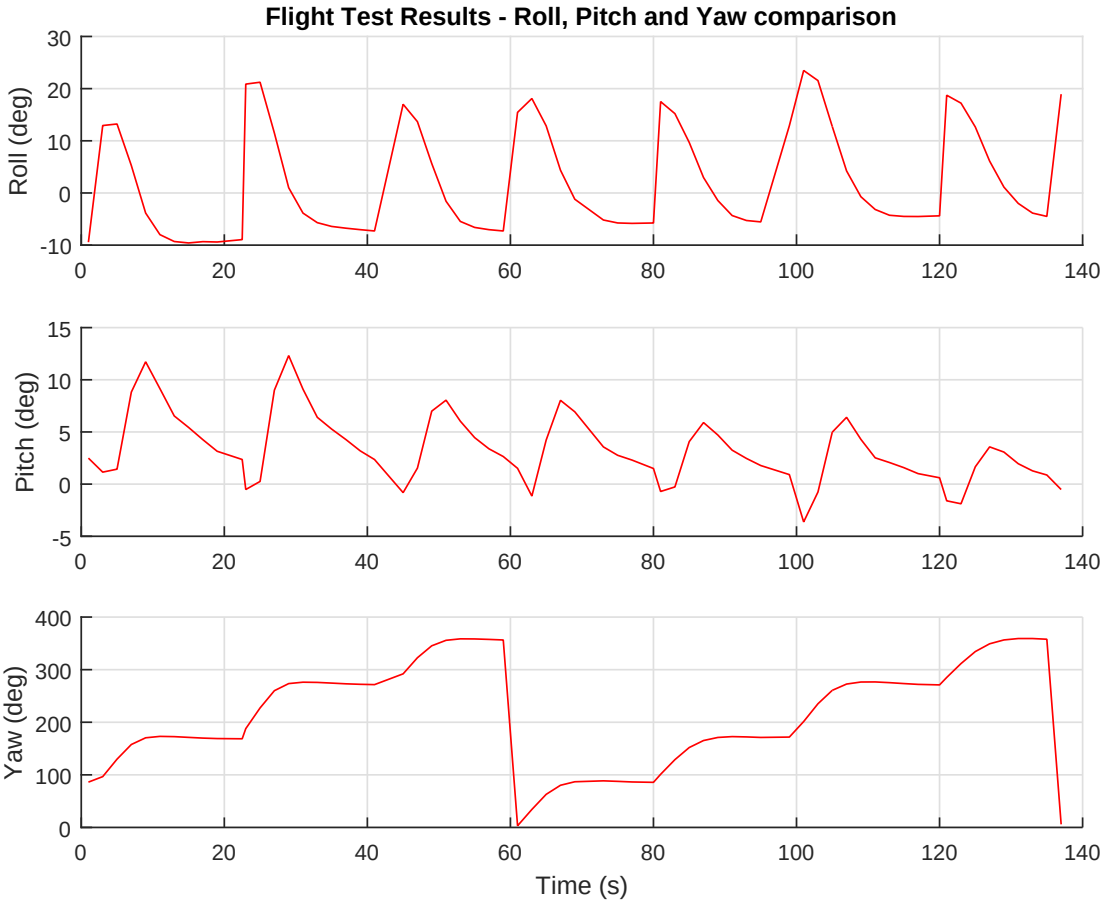


Figure 5.13: Flight test with a single UAV is presented. The body angles are the relative attitude of the body. It verifies the results of the simple scenario used to fly the previous version of SLUGS autopilot.

5.3.4 Multi-UAV System Validation

The Multi-UAV system was built with three components: two UAVs and a GCS. The basic multi-UAV flight formation takes into account the communication aspects of the deployed system. The goal of this set of flight tests was to examine different strategies and major challenges for formation flight testing.

The system architecture utilized a centralized scheme of operation to enable coordination and information sharing. The implementation included a manual

mechanism that allowed one UAV to be in "Leader" mode and assigned the other as the "Follower" UAV. The mode of operation of the UAVs could be set dynamically. That is, their role could be changed during the field test. More specifically, in the field test data shown, the GCS handles the mission and the two UAVs were using a predetermined trajectory for the flight test.

Fig. 5.14 shows the top level integration design. Two UAVs utilized communication through the GCS or directly between them (using XBees transceivers). Since the flight includes manual remote control, RC controllers and receivers were also integrated within the UAVs.

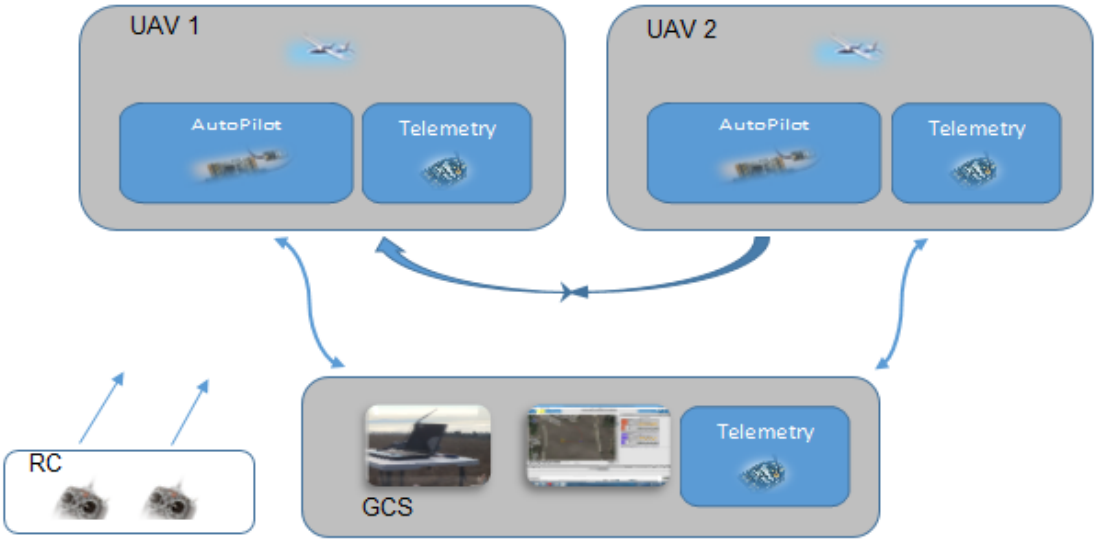


Figure 5.14: A top level diagram of the Formation Testing System is shown. Two UAVs which include an AutoPilot unit and a Telemetry unit can communicate with the ground control station (GCS) and can also be controlled by a remote-controller for a fail-safe procedure.

Fig. 5.15 shows the outcome of the integration with the basic real-time components; autopilot, the UAV platforms and the GCS a moment before performing a field-test.

A field-test demonstrating basic flight formation was used to examine the

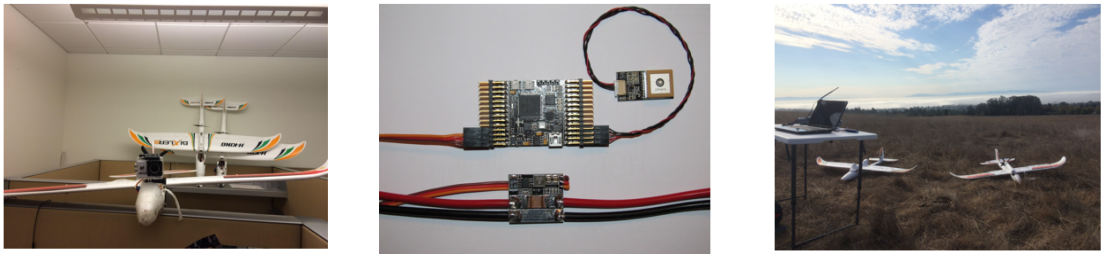


Figure 5.15: Experiment hardware is shown. On the left, three airplanes model that have been used during the field test. In the middle, the AUAV3 board used for running the autopilot software. On the right, the setup of the field

integrated system. The system was switched to Autonomous mode immediately after launch and after the UAVs were far away from each other, the formation configuration was initiated. Fig. 5.16 shows a sequence of positions for each UAV. The "Leader" (in red color), tracks the default trajectory and the "Follower" (in blue color) updates the mission plan with a new waypoint sent from the current "Leader" position at 1Hz. The updates are transmitted by the GCS to all the "Follower" members.

Figure 5.16: Two UAVs in Leader-Follower configuration

5.4 Conclusions

The SLUGS II autopilot obtains the same functionality in the migrated Simulink model as found in the original model. It uses average CPU loading that does not exceed 60%. The reserve computation time leaves enough resources for further

enhancement and evolution.

The design architecture for local data-sharing and for a centralized control scheme have been explored. The Leader-Follower preliminary tests demonstrate communication difficulties. Only a few low-cost communication components that support Mesh-protocol were found, and most of the transceivers tended to fail. These transceivers support a decentralized configuration through their mesh protocol feature. They are, however, not robust. The final field test in that configuration was set to work in a centralized configuration although single point failures could cause a centralized system to be non-functional [78].

The system experienced communication dropouts, and that is one of the reasons why the estimator has been designed to work sequentially. Traditionally there are two stages for a recursive estimator; predict and update. Thus, the estimator considers cases where observations are not available. At any arbitrary time step, it executes the predicting step only, with no update step until a measurement is available.

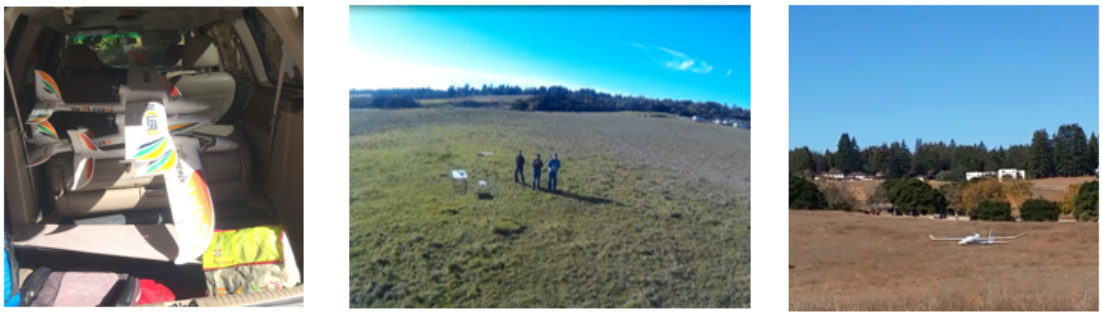


Figure 5.17: Field Test in UCSC - Before and After

From the field-tests one can conclude the following results:

- The developed test-bed permits examination of different types of communications topologies and guidance systems to obtain UAVs formation flight.

- The developed setup offers fast and satisfactory integration of COTS: Airplane models (Bixler2), Auto-Pilot board (AUAV3), Auto-Pilot open source software (MatrixPilot & SLUGS II) and GCS open source software (QGC).

The AUAV3 board comprises an MPU6050 gyroscope and accelerometer, a HMC5883L magnetometer and a BMP-180 barometer. The barometer gave noisy measurements which required designing a new robust altitude sensor filter in the Simulink model. Moreover, the differential pressure sensor and airspeed sensor used in SLUGS are not available in SLUGS II setup. It is strongly recommended to integrate a differential pressure sensor to get a better altitude keeping and speed control.

Chapter 6

Conclusions & Future Work

6.1 Conclusions

In summary, this thesis has presented the core methodology for coordinating a fleet of UAVs to suppress the uncertainty of a generic ground phenomenon. The coordination technique integrated with an R&D monitoring system which was designed carefully to improve the estimation of a propagated periphery supports decision making in an operational scenario.

The periphery estimation method shows that it is essential to deploy a number of the UAVs to validate the accuracy of the deterministic models of the phenomena. The results show that estimation based on the spatial observations adjusts the unobserved state (i.e., spread rate). It also facilitates a preceding step based on approximation estimators to show whether the developed solution (QKF & GUS) is robust.

The coordination technique reduces the variance from more than one direction; it is shown that by adding more UAVs that move in a different direction, the growing variance can be bounded. The prediction can point to the necessity for more resources (UAVs) in real-time.

The system design comprises the major components of a R&D monitoring system: high-level-controller, flight control, and ground control. The development process of the UAV flight controller (autopilot) has been improved with COTS board and a new development environment for software validation. The SLUGS II autopilot obtains the same functionality in the migrated Simulink model as found in the original model. The generated code uses on average a 60% CPU loading. The reserve computation time leaves enough computational resources for further enhancement and evolution.

MSIL simulation tests the generated code in a flexible and friendly environment that is committed to the sequence of events in the software rather than to guarantee valid real-time performance execution of the code. The system is designed to be agnostic as to the type of phenomenon that is being tracked, and can be made to work well for a number of different scenarios.

Wildfire incidents are an example of a stochastic phenomenon, and knowing the fire boundary with high certainty would improve decision-making by the ground team. The variety of sensors available today is enormous, but sometimes the environmental conditions are so severe that it is impossible to benefit from the best sensors. This research shows that even if the sensor is binary, the QKF estimator can achieve high performance estimation of the periphery. It shows that the quantized information can improve the prediction with the right deployment and with a coordination scheme for multiple sensors. This approach relies on a fast deployable fleet of UAVs designed to detect limited information, generate an uncertainty map, and incorporate the information into a new mission plan.

The developed estimation technique examines how the initial covariance (or uncertainty) of the CP evolves while the CP state vector is predicted based on the dynamic model of the propagation. In any operational application, the dynamic

model is by definition an approximation and therefore will drift away from the actual boundary. To represent inaccuracy in the model the derived model assumes it has a process noise. One would expect a gross accumulation of error as the time passes in "open loop" to increase.

6.2 Future Work

The deployment stage helps achieve less uncertainty while the UAV search for their first periphery crossing points. Extending this stage with different deployment schemes for a fleet of UAVs can both improve uncertainty convergence and error convergence.

The propagated phenomenon can have different process probability distributions. Extending the deterministic model to account for vegetation type and ridge orientation can improve the prediction when observations are not available or do not have enough influence on the boundary.

The sensing capabilities are assumed to be poor. The QKF method is not limited to only one type of sensor, and it should then be explored and adjusted to other sensing capabilities with different noise distributions.

The high-level-controller was developed and tested entirely within Matlab environment. In a final design, the controller should be deployed on the GCS for centralized architecture or on the AUAV3 for decentralized architecture.

Appendix A

Appendix

A.1 Methods

A.1.1 Coordination

The definition of coordination has much diversity [60]. All the definitions include a resource that needs to be shared by a group of entities. In the study on coordination([60]) the author offers the following definition:
'Coordination is managing dependencies between activities'.
The multi-UAVs in the proposed research are intended to track a reference line by coordinating assigned segments between them. The UAVs shared resource is the target space at a specific time, meaning that there are dependencies requiring a solution. This is due to the fact that the paths will likely cross.

The process of coordination connects tasks and resources. The basic task is to employ an allocated segment for a limited time while another is to assign the same type of task but with a different segment. The completed step produces additional information to the next step of the coordination scheme. The timing of broadcasting this information is very crucial.

Solving the task assignment problem is possible using two architectures that are connected to the fact that information is sometimes unknown. A Centralized System assumes that clients and servers share their information with a central entity. In this architecture, the decision will be made by one authority, and allocation to the server will be sent from it. The other side of the spectrum is a client that can send new information to potential servers. The one who holds new information can possibly decide on the right task allocation for the potential servers. That type of architecture is closer to the definition of a Distributed Controller.

Another interesting research subject in the coordination topic is how to approach the analysis of coordination([60]). Bases on the definition of coordination it is clear that a good analysis depends on the critical parameters that might influence the performance of activities. Identifying parameters for evaluation will be the most important part of the analysis itself. In the proposed research, for instance, an uncovered segment of the AOI will have a high cost, a small cross-track error (relative to the actual perimeter) will have a small cost.

A.1.2 Mission Planning

Mission planning is widely used for UAV applications. Mission planning generates the overall mission properties for a given scenario. Classical mission planner constructs the plans based on deterministic models or predictors and re-plan in real-time to capture environmental adjustments. Another type of mission planner prepares the mission plans based on stochastic models and re-plan in real-time after adjusting the model's statistical properties.

Search, exploration, and coverage problems have been investigated extensively

in robotics and aerial literature [72]. Those tasks depend upon target properties (e.g., area, size, dynamics, etc.). Exploration problems detect the target space and explore it with no particular demand related to the past visited regions. Coverage problems find the optimal deployment of the sensors such that target space is covered continuously and completely.

Monitoring problems, however, differ from all three. The sensing capability is utilized for additional regions and therefore work more efficiently. The vehicle carries the sensor, moves continuously, and surveys the target space repeatedly. Monitoring missions maintain the updated "Map" of the area while trying to minimize the visiting time between current and oldest regions.

The main difference between the mission planning and mission controlling processes is that one is an offline process and the second is a real-time process. This work focuses on the challenging stage of mission control. The system updates the mission plan according to data collected in real-time. Any new measurement is additional information that is added to the offline planning. If the system has benefited from planning based on starting information, it surely has an advantage from information updates.

A.1.3 Path Following

In path following the objective is to be on the path rather than at a point at a particular time([10]). There are different techniques to follow a path by a small unmanned aircraft, and they are all based on a reference trajectory. The desired trajectory output from the mission planning stage is the reference trajectory, and the path-following scheme uses it. Path following points at a candidate waypoint to which the guidance system should point. The traditional guidance laws work with two types of data structure primitives which represent the planned trajectory:

straight-line and circular orbit.

The main reason why a UAV turn should have a circular orbit results from physical constraints. Fixed wing airplanes must keep their airspeed, hence, must keep moving while changing direction (i.e., turning). The formal definition for that type of dynamical systems is a nonholonomic system. Such a system is described by a set of parameters that vary continuously when the system is moving along a path.

Several studies present guidance logic for a terminal phase or trajectory-following [32]. The method adopted by most implementations is trajectory-following by using a point, (sometimes called 'virtual target'), along with the desired flight path. By moving that point and aiming the line of sight to it, proportional navigation provides decent and robust performance [86].

The authors [86] show that the guidance system uses the guidance strategy to follow a set of waypoints and sometimes switching between the mode of operations as the UAV reaches its capability to follow the track is required. Whenever the path includes a sharp turn, the guidance system should work to close the gap between a pair of configurations. The authors suggest adding an element of anticipatory control that enables tight path tracking when following curved paths.

A Dubins path is a known representation of the transition from one configuration (direction and position) to another [58]. Dubins paths based on specific kinematics resolve a time-optimal path between two configurations. The path connecting two configurations consists of a turn (i.e., circular arc), a straight line (straight and leveled) and another turn into the final configuration.

A.1.4 Gaussian Transformation

The observations are distributed in space ($\mathcal{R}^2$), and each one of them is part of the predicted distribution of the CPs state. To incorporate that information we should relate the observation to the CPs distribution with a spatial measure. That should be based on their relative position in space. One way to do it is by using the distance as a measure of impact on the examined CP.

In an attempt to address the transformation of probability densities (mapping problem) we rely on a related theorem from the literature. The theorem states that any random variable X with a multivariate Gaussian distribution can be interpreted as the result of applying a linear transformation ($X = BZ + \mu$) to some collection of n independent standard normal random variables (Z).

It is reasonable then to connect a line between the observation and the prediction CP location to provide, although partial, but additional information to the estimation process. We obtain a one-dimensional variance by transformation from original coordinates of the prior covariance. The projected variance is an accurate partial measure of the related estimated CP and its associated covariance.
The expected value is a second measure that should be evaluated in the new coordinate system, and it would be relative to the distance between the observation and between the predicted point, (the CP). Because of the relationship within the original coordinates system (x, y) implied by the covariance, knowing the distant class label (y_i) along the connected line reduces the uncertainty on both of the original coordinates.
Moreover, the observations are distributed in space, and therefore the general proposed method would incorporate the class labels according to their associated distance to the predicted CP. We propose a local transformation to solve for the

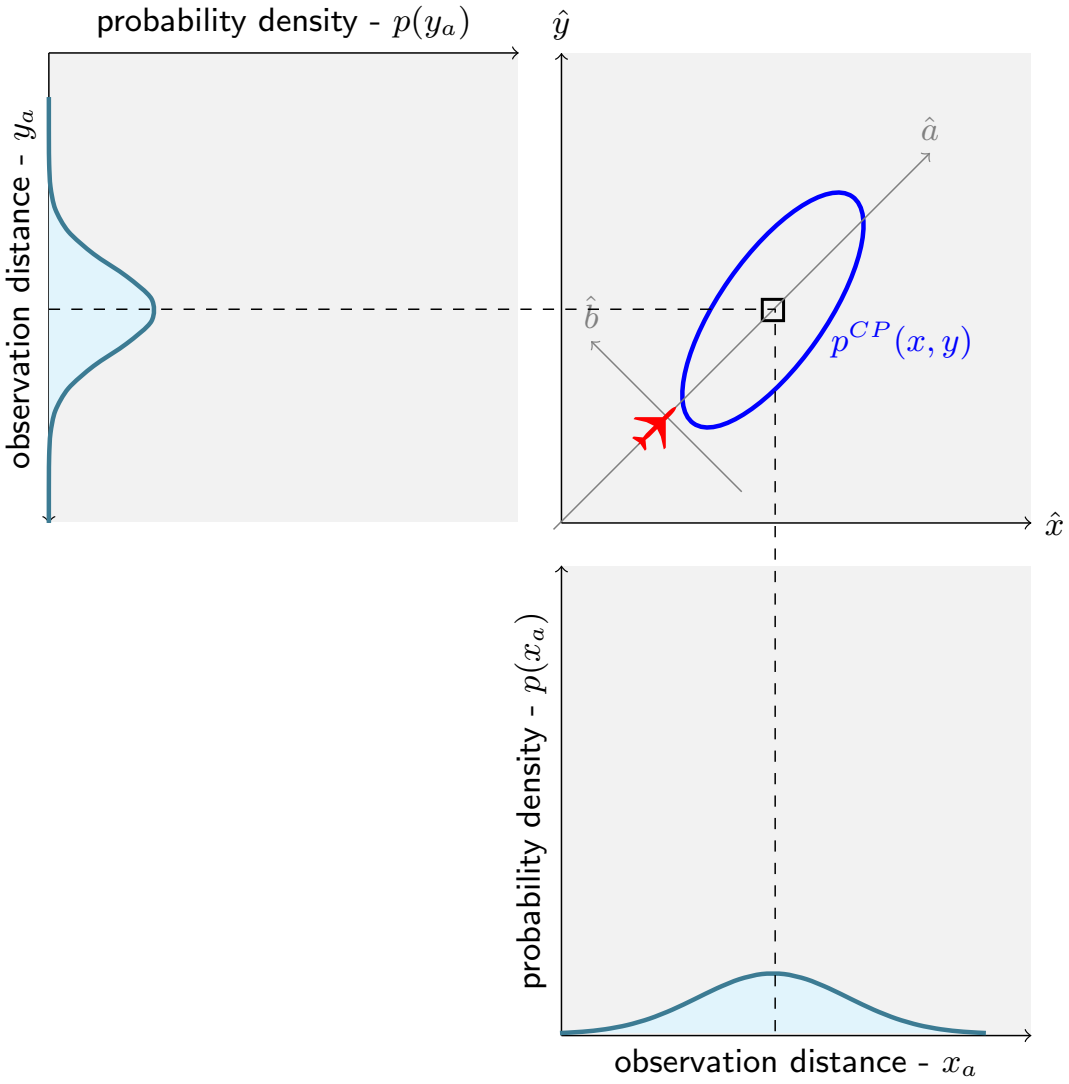


Figure A.1: The transformation method for a non-significant case is presented. The CP, (square marker in the figure), lies in the $\hat{x}, \hat{y}$ coordinate system, the covariance plots as an ellipse and the drone in red fixed-wing shape. The transformation probability densities of the original covariance are projected along the local coordinate system $\hat{x}_a, \hat{y}_b$. Notice that the location of the drone, in this case, is outbound and therefore the observation is not significant.

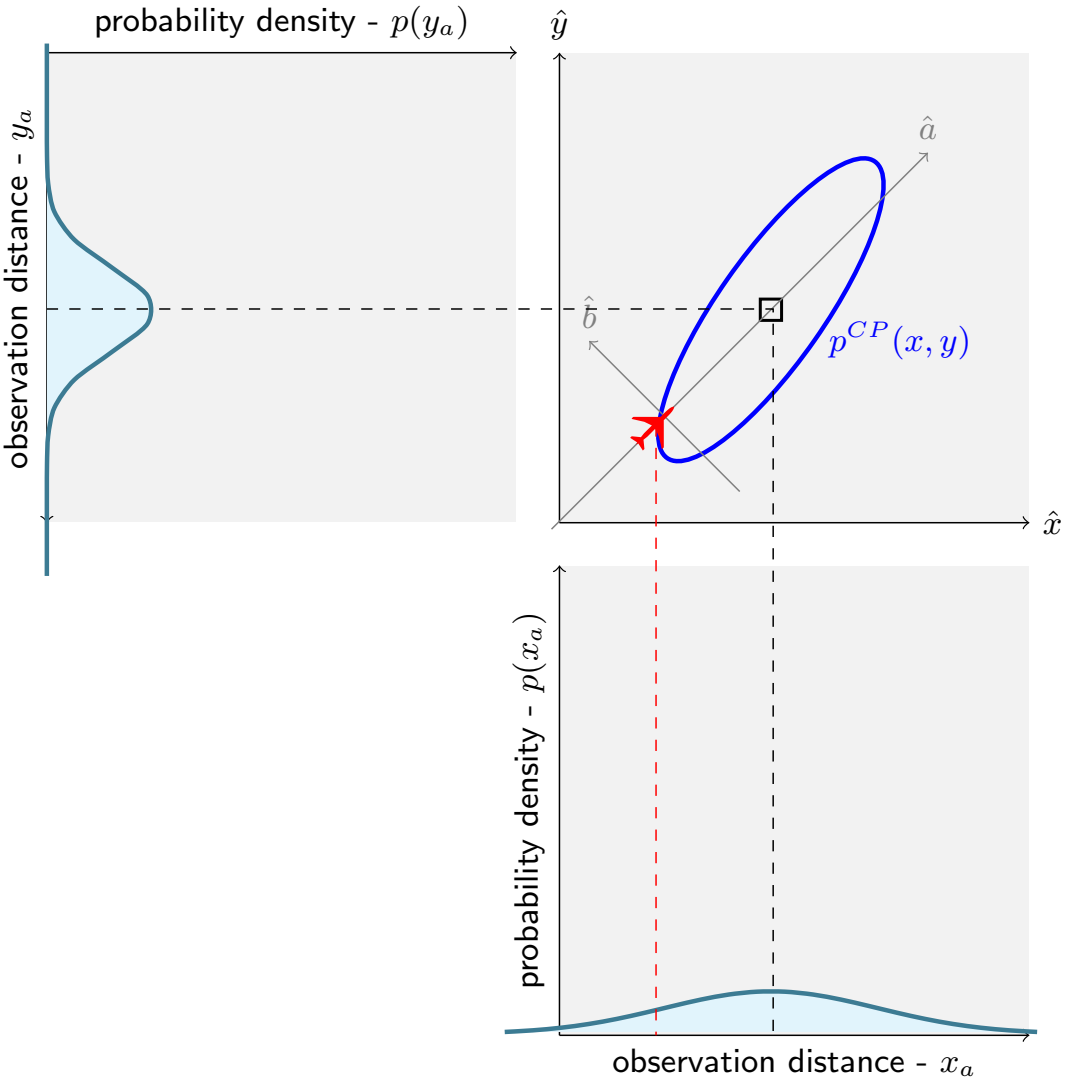


Figure A.2: The transformation method for a significant case is presented. The CP, (the square point), lies in the $\hat{x}, \hat{y}$ coordinate system, the covariance draws as an ellipse and the drone in red fixed-wing shape. The transformation probability densities of the original covariance are projected along the local coordinate system $\hat{x}_a, \hat{y}_b$. Notice that the location of the drone, in this case, is in boundaries (dashed red line) and therefore the observation is significant and should influence the estimation process.

probability of a random variable, assuming the CP location is Gaussian distribution, and also the projection of it is, $x_a \sim N(\mu, \sigma^2)$ takes values less or equal than a given real number $x_a^{CP} \in \Re$.

A.1.5 Approximate Particle Filter Method

Particle Filters are heuristic algorithms which rely on a statistical model of the examined process. The algorithms perform a two-stage transition sequentially and form the basis for Sequential Monte Carlo (SMC) methods. The typical problem that demonstrates the Particle Filter method is the localization problem (e.g., Terrain-Based Positioning). In this type of operational problem, the robot travels with an uncertainty of its actual location, and a reference map holds the surrounding environment. For example, when the reference is a terrain map, by measuring only the altitude above ground (in addition to noise), the reference map becomes useful information for localization. The main concept is to find the similarity between current measurement and any possible virtual location represented by the particles. Each particle is weighted relative to the error between the actual measurement and a randomly sampled measurement. A likelihood function evaluates the weight (e.g., pdf for a normal distribution). The smaller the difference, the higher the importance of that particle. The second stage in the algorithm is the resampling. The resample step adds random particles in between the existing list of the dominant particles.

One of the questions in this research is how to use that method when we do not have a reference map?

The practice shows that a moving agent can predict the boundary simply by choosing a set of distant particles (i.e., samples), which are uniformly distributed, and by evaluating the similarity between the observation and the particles it can map

the environment ([77]). The similarity suggests that particle's relative-location can either support or oppose the predicted boundary. If the particle is outbound and the measurement is also outbound, then the particle is less likely to represent the boundary. If the evaluated particle is inbound relative to the predicted boundary, it is more likely to represent the boundary distribution. The only information that we have in our problem is the Reported Location, which is the initial point of the propagated phenomenon. For simplicity, we assume that the measurements are only from outbound (represent the deployment phase of the mission). We assume that prior probability of the bounded area is proportional to the explored region. The second interesting question that arises is when is the best time to update the predicted boundary (the discrete control points) ?

The following equation averages the dominant particles according to their evaluated weights to find out the suggested update of a point on the boundary:

$$\hat{\mathbf{x}} = \frac{\Sigma w^{[i]} \mathbf{x}^{[i]}}{\Sigma w^{[i]}} \tag{A.1}$$

The likelihood function is used for statistical inference. It describes the function of a parameter for a given outcome and is only used after data are available. In contrast to likelihood function, the probability function is used before data are available and describes possible future outcomes for a given value of the parameter. In estimation methods, we apply the likelihood function to evaluate the measure of some attribute of a sample (statistic) from a set of measurements. In this work, the inference of the current distribution is based on indirect measurement of the propagated phenomenon. The observation is either inside the boundaries or out of boundaries and based on the location of the measurement the estimation process evaluate how to affect a currently predicted boundary (grid points).

A.1.6 QKF - MMSE Approximation

The overall objective of an estimation process is to estimate $\hat{x}$. To evaluate the parameters or the state of interest in terms of error we would usually compare it to the actual state x. The difference between the estimate and actual state, (which is unknown), is the error: $e = x - \hat{x}$. A function of the error that considers the ability of the filter to estimate data over a period of time is the expected value of the squared errors: $E\{(x - \hat{x})^2\}$. From all possible set of filters, the optimal filter is defined as the one that minimizes the mean squared estimate (MSE), or the Minimum MSE (MMSE).

Linear Observations

For the case of non-quantized measurement, assuming that the actual measurements are linear. The measurements are equal to sum of the state transformed into observation space and an associated measurement noise. Hence, observations can be modeled in the form:

$$z = Hx + v \tag{A.2}$$

Also, the new estimate of the state is a linear combination of the old estimate, (the prior), with the measurement residual:

$$\hat{x} = x^{-} + K(z - Hx^{-}) \tag{A.3}$$

Substituting A.2 into A.3:

$$\hat{x} = x^{-} + K(Hx + v - Hx^{-}) \tag{A.4}$$

By subtracting x from both side of the equation:

$$x - \hat{x} = x - x^{-} - K(Hx + v - Hx^{-}) = [I - KH][x - x^{-}] \tag{A.5}$$

and by deriving the error covariance we get:

$$\begin{aligned} P = E[ee^T] = E[(x - \hat{x})(x - \hat{x})^T] = \\ E[[(I - KH)(x - x^{-}) - Kv][(I - KH)(x - x^{-}) - Kv]^T] \end{aligned} \tag{A.6}$$

This equation is difficult to calculate in practice as we would not know the *True* state x, however the linear estimator recursive approach assumes that $(x - x^{-})$ is the error of the prior estimate.

$$P = (I - KH)P^{-}(I - KH)^T + KRK^T \tag{A.7}$$

Expansion of A.7 gives:

$$\begin{aligned} P = \\ = [I - KH]P^{(-)}[I - KH]^T + KRK^T = \\ = P^{(-)} - KHP^{(-)} - P^{(-)}H^TK^T \\ +K(HP^{(-)}H^T + R)K^T \end{aligned} \tag{A.8}$$

The equation is the error covariance update equation. The diagonal of the covariance matrix contains the mean squared errors. MMSE is also minimization of the sum of the MSE, hence the trace, (noted in the equations as $Trace$), of the

covariance matrix P.

$$\begin{aligned} Trace[P] = \\ Trace[P^{(-)}] \\ -2Trace[KHP^{(-)}] \\ +Trace[K(HP^{(-)}H^T + R)K^T] \end{aligned} \tag{A.9}$$

To find the minimum condition we would differentiate the trace of $cov(x|z \in A)$ with respect to K and set the equation to zero:

$$\frac{dTrace[P]}{dK} = -2(HP^{(-)})^T + 2K(HP^{(-)}H^T + R) \tag{A.10}$$

Solving for K gives:

$$K = P^{(-)}H^T \left(HP^{(-)}H^T + R\right)^{-1} \tag{A.11}$$

where K is the same time-varying weighting matrix (the minimum variance gain) for Gaussian random variables in 2.15 and that been used by the author in the [27].

Quantized Observations

The derivation of the Kalman filter type estimator in [27] devised that the optimal gain, K, remains the same for the quantized measurements as it is originally derived for linear Kalman filter. In this section, we propose to examine the MSE

(Mean Squared Error) followed by the QKF derivation. Expansion of 2.24 gives:

$$\begin{aligned} cov(x|z \in A) = \\ = [I - KH]P^{(-)}[I - KH]^T \\ +KRK^T + Kcov(z|z \in A)K^T = \\ = P^{(-)} - KHP^{(-)} - P^{(-)}H^TK^T \\ +K(HP^{(-)}H^T + R)K^T \\ +Kcov(z|z \in A)K^T \end{aligned} \tag{A.12}$$

The equation is the error covariance update equation. The diagonal of the covariance matrix contains the mean squared errors. MMSE is also minimization of the sum of the MSE, hence the trace,(noted in the equations as $Trace$), of the covariance matrix $cov(x|z \in A)$.

$$\begin{aligned} Trace[cov(x|z \in A)] = Trace[P^{(-)}] \\ -2Trace[KHP^{(-)}] \\ +Trace[K(HP^{(-)}H^T + R)K^T] \\ +Trace[Kcov(z|z \in A)K^T] \end{aligned} \tag{A.13}$$

To find the minimum condition we would differentiate the trace of $cov(x|z \in A)$ with respect to K and set the equation to zero:

$$\begin{aligned} \frac{dTrace[cov(x|z \in A)]}{dK} \\ = -2(HP^{(-)})^T + 2K(HP^{(-)}H^T + R) \\ +2Kcov(z|z \in A) \end{aligned} \tag{A.14}$$

Solving for K gives:

$$K = P^{(-)}H^T \left(HP^{(-)}H^T + R + cov(z|z \in A) \right)^{-1} \tag{A.15}$$

Notice that the innovation has an associated measurement prediction covariance and now an additional associated quantized measurement prediction covariance.

A.1.7 MAP Parameters Approximation

The the theoretical formulas (2.4,2.6) can be hard to apply in real-time situations. Applying them directly would require infinite samples and a high computational load.

The key in a real-time estimator implementation is to use a recursive estimator to incrementally update the posterior probability distribution of the state vector based on most recent data. MAP estimation fuses both the a-priori observation and the new one to come up with an estimate. It repeats that process by using the previous estimate as a-priori and incorporating it with a fresh observation. Assuming the observations produced by the sensor sequentially, it would compute the estimate values in each time step.

One can find a closed form solution for the general MAP equations, but it cannot be applied in here since the observation is bounded.

The probability is conditioned on measurement z, which is bounded on the region $[a, b]$. Following the general description in equation A.16, the parameters for the posterior probability are a product of the conditional probability and the prior:

$$P^+(\theta|z \in [a, b]) = P(z \in [a, b]|\theta) \times P^-(\theta) \tag{A.16}$$

The first term on the left side is the likelihood (also the CDF), and the second

term is the prior:

$$P(z \in [a,b]|\theta) = \Phi(\frac{b-\mu}{\sqrt{2}\sigma}) - \Phi(\frac{a-\mu}{\sqrt{2}\sigma}) \tag{A.17}$$

where Φ is the CDF bounded from left or right ($[a,b]$), and the difference between them is the probability mass contained in the interval $\mu \pm \sigma$. Equation A.16 may be expended to give:

$$\begin{aligned} P^+(\theta|z \in [a,b]) = \\ = \left(\Phi(\frac{b-\mu}{\sqrt{2}\sigma}) - \Phi(\frac{a-\mu}{\sqrt{2}\sigma})\right) \times \phi(\frac{z-\mu}{\sqrt{2}\sigma}) \end{aligned} \tag{A.18}$$

where, ϕ is the pdf and z is our current measurement.

To find the maximum a-posteriori one would vary the parameters θ until it reaches maximum probability. Alternately, this can be done by using optimization methods to search effectively, differentiating the cost function, setting the derivative to zero, and solving the partial derivatives of the posterior probability, $P^+(\theta|z \in [a,b])$, for the best parameters θ.

$$\frac{\partial}{\partial\sigma} P^+(\hat{\theta}|z \in [a,b]) = 0 \tag{A.19}$$

$$\frac{\partial}{\partial\mu} P^+(\hat{\theta}|z \in [a,b]) = 0 \tag{A.20}$$

$$\frac{\partial}{\partial\sigma} P^+(\hat{\theta}|z \in [a,b]) = \frac{\partial}{\partial\sigma} P(z \in [a,b]|\theta) \times P^-(\theta) + P(z \in [a,b]|\theta) \times \frac{\partial}{\partial\sigma} P^-(\theta) \tag{A.21}$$

$$\frac{\partial}{\partial\mu}P^{+}(\hat{\theta}|z\in[a,b]) = \frac{\partial}{\partial\mu}P(z\in[a,b]|\theta)\times P^{-}(\theta) + P(z\in[a,b]|\theta)\times\frac{\partial}{\partial\mu}P^{-}(\theta) \tag{A.22}$$

$$\begin{aligned}\frac{\partial}{\partial\sigma}P^{+}(\hat{\theta}|z\in[a,b]) = \\ \frac{1}{2\sigma}\left\{\Phi(\frac{b-\mu}{\sqrt{2}\sigma}) - \Phi(\frac{a-\mu}{\sqrt{2}\sigma})\right\}\phi(\frac{a-\mu}{\sqrt{2}\sigma}) + \\ \left\{\Phi(\frac{b-\mu}{\sqrt{2}\sigma}) - \Phi(\frac{a-\mu}{\sqrt{2}\sigma})\right\}\left\{-\frac{1}{\sigma}\phi(\frac{a-\mu}{\sqrt{2}\sigma}) + \frac{(z-\mu)^2}{\sigma^3}\phi(\frac{a-\mu}{\sqrt{2}\sigma})\right\}\end{aligned} \tag{A.23}$$

where ϕ is the standard normal pdf.

$$\begin{aligned}\frac{\partial}{\partial\mu}P^{+}(\hat{\theta}|z\in[a,b]) = \\ \frac{1}{2\mu}\left\{\Phi(\frac{b-\mu}{\sqrt{2}\sigma}) - \Phi(\frac{a-\mu}{\sqrt{2}\sigma})\right\}\phi(\frac{a-\mu}{\sqrt{2}\sigma}) + \\ \left\{\Phi(\frac{b-\mu}{\sqrt{2}\sigma}) - \Phi(\frac{a-\mu}{\sqrt{2}\sigma})\right\}\left\{\frac{(z-\mu)}{\sigma^2}\phi(\frac{a-\mu}{\sqrt{2}\sigma})\right\}\end{aligned} \tag{A.24}$$

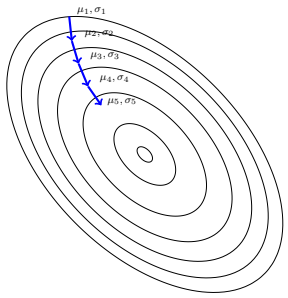


Figure A.3: This is n example of numerical implementation of the Gradient Descent optimization method. The indifference curves represent a simple strongly-convex function. The process search is for the best match of $\hat{\mu}, \hat{\sigma}$, which tends to be on the minimal or maximal point.

Where the models are non-linear and have no closed-form solution, a general optimization search problem in a continuous space is advisable. Such problems can be addressed by a Gradient Descent/Ascent algorithm that follows the gra-

dient of the function to be optimized. After applying the transformation to the local coordinates system of the candidate CP state vector, it chooses the resulted predicted parameters as a starting-point for the parameters μ, σ search. Next, the algorithm moves to a neighboring point that is uphill, repeating that process until it converges on a maximum probability value for the given new interval. The

Algorithm 3 Gradient descent method.

Given a starting point $x \in \mathbf{dom} f$
repeat
1. $\Delta x := -\nabla f(x)$.
2. *Line search.* Choose step size t.
3. *Update.* $x := x + t\Delta x$

until stopping criterion is satisfied.

Gradient descent algorithm 3, as it has been applied, starts from the predicted CP statistical characteristics. In step 2 of the Gradient Descent method, it chooses a constant increment to seek along the line search. Figure A.3 explains how the algorithm works, reaching the minimal point.

In this implementation, the cost function is the CDF (cumulative distribution function) defined on the interval $[a, b]$ between the observation and the CP. The interval is determined from interpreting the observation in the local coordinate system. The algorithm evaluates the probability for each iteration of updated parameters. The gradient $\nabla f(x)$ shall be the partial derivatives determine in A.23 and A.24.

Algorithm 4 Estimation with MAP Approximation.

Given a CP state vector x_i^{CP} and observations y_i

repeat

1. *Predict.* $x_{x,y}^{CP(+)} := \Phi \cdot x_{x,y}^{CP(-)}$
2. *Transformation.* $T = T_{x,y}^{a,b}$.
3. *Translation.* $x_a^{CP} := Dist(x_{x,y}^{CP}, x_{x,y}^{DP})$.
4. *Covariance Evaluation.* $P_{a,b}^{CP} := T \cdot P_{x,y}^{CP} \cdot T^T$.
5. *1D Evaluation.* $\mu_a := x_a^{CP}, \sigma_a := P_{a,b}^{CP}(1,1)$.
6. *Set search Bounds.* $(\lfloor a \rfloor, \lceil b \rceil)$.
7. *MAP.* $[\hat{\mu_a}, \hat{\sigma_a}] := \arg\max_{\mu_a, \sigma_a} p(z|x)p(x)$.
8. *1D Update.* $x_a^{CP} := \hat{\mu_a}, P_{a,b}^{CP}(1,1) = \hat{\sigma_a}$.
9. *Correlation Correction.* $P_{a,b}^{CP} = P_{a,b}^{CP} \cdot \frac{\hat{\sigma_a}}{\sigma_a}$.
10. *Inverse Transformation.* $P_{x,y}^{CP} := T^T \cdot P_{a,b}^{CP} \cdot T$.
11. *Inverse Translation.* $\hat{x}_{x,y}^{CP} := x_{x,y}^{CP(+)} + T^T \cdot (\hat{\mu_a} - x_{x,y}^{CP(+)})$

until stopping criterion is satisfied.

A.1.8 MAP Conditional Covariance Approximation

The approximation for the conditional covariance, cov_{MAP}, is treated in the same way as the deriving the approximated mean value.

$$cov_{MAP} = E(z - z^*)^2 \tag{A.25}$$

The covariance integral is by definition an integral of the product of squared errors and the PDF of the prior ($p(z)$):

$$cov_{MAP} = \int_a^{\infty} (z - z^*)^2 \cdot \frac{p(z)}{P(z \in A)} dz \tag{A.26}$$

The presented case in Eq. A.26 is for the non-conflicted observation where a is the lower bound and the upper bound is ∞. The term $P(z \in A)$ is used to scaling the density so that it integrates to one over the range of A and it evaluated in Eq. 2.27. By expanding the equation to three integrals :

$$cov_{MAP} = \int_a^{\infty} z^2 \cdot \frac{p(z)}{P(z \in A)} dz - 2z^* \int_a^{\infty} z \cdot \frac{p(z)}{P(z \in A)} dz + z^{*2} \cdot \int_a^{\infty} \frac{p(z)}{P(z \in A)} dz \tag{A.27}$$

Note that the integral of the second term is the conditional mean $E(z|z \in A)$ and that the integral in the third term is by definition one. By using basic exponential integral solutions:

$$\int z \cdot e^{-c \cdot z^2} dz = -\frac{1}{2c} e^{-c \cdot z^2} \tag{A.28}$$

$$\int z^2 \cdot e^{-c \cdot z^2} dz = \frac{1}{4} \sqrt{\frac{\pi}{c^3}} erf(z\sqrt{c}) - \frac{z}{2c} e^{-c \cdot z^2} \tag{A.29}$$

where, the variable of integration is z and c is a constant variable. Utilizing the formulas and substituting the limits of the first two terms in Eq. A.27 is given by:

$$\begin{aligned} cov_{MAP} &= \frac{1}{P(z \in A)} \cdot d \cdot \left[\int_a^{\infty} z^2 \cdot e^{-\frac{z^2}{2\sigma_z^2}} dz - 2z^* \int_a^{\infty} z \cdot e^{-\frac{z^2}{2\sigma_z^2}} dz \right] + z^{*2} \\ &= \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4} \sqrt{\frac{\pi}{c^3}} \cdot (1 - erf(a\sqrt{c})) + \frac{a}{2c} e^{-c \cdot a^2} - z^* \cdot \frac{1}{c} e^{-ca^2} \right] + z^{*2} \end{aligned} \tag{A.30}$$

where, c, d are constant variables which represent the scaling factors for the normal *pdf*: $c \equiv \frac{1}{2\sigma_z^2}$ and $d \equiv \frac{1}{\sqrt{2\pi}\sigma_z}$.

In the conflicted case, where expected is not as been observed, the truncation

for upper limit, b, is:

$$cov_{MAP} = \frac{1}{P(z \in A)} \cdot d \cdot \left[\frac{1}{4}\sqrt{\frac{\pi}{c^3}} \cdot (1 + erf(b\sqrt{c})) - \frac{b}{2c} e^{-c \cdot b^2} + z^* \cdot \frac{1}{c} e^{-cb^2} \right] + z^{*2} \tag{A.31}$$

A.1.9 The Unscented Transformation

The unscented transformation described in [50] is founded on the understanding that it is easier to estimate Gaussian statistical properties than to approximate nonlinear mapping function. To calculate the statistics of the mapping output y_k, the method determines a matrix χ that includes $2L + 1$ candidate points, where L is the dimension of the state vector x_k. The candidate points are also known as the sigma points, and each of them is a vector and the ith row in the matrix $\chi^{(i)}$:

$$\begin{aligned} \chi_0 &= \hat{x}_{k-1} \\ \chi_{k-1}^{(i)} &= \hat{x}_{k-1} + (\sqrt{(L+\lambda)P_{x_{,k-1}}})_i, \quad i = 1, ..., L \\ \chi_{k-1}^{(i)} &= \hat{x}_{k-1} - (\sqrt{(L+\lambda)P_{x_{,k-1}}})_{i-L}, \quad i = L+1, ..., 2L \end{aligned} \tag{A.32}$$

where λ, α and β are scaling parameters, $\hat{x}$ is the prior state estimate, and $P_{x_{,k-1}}$ is prior state covariance.

Next, the sigma points are propagated through the state prediction function:

$$\chi_{k|k-1}^{(i)} = f(\chi_{k-1}^{(i)}), \quad i = 1, ..., 2L \tag{A.33}$$

and the transformed sigma points sent through the observation model:

$$Y_{k|k-1}^{(i)} = h(\chi_{k|k-1}^{(i)}), \quad i = 1, ..., 2L \tag{A.34}$$

Note, the subscript $k|k-1$ means that this is the predicted value based on the information from the last step. To evaluate the relative contribution of the sigma points to the current time-step, the method determines their corresponding weights W_i:

$$\begin{aligned} W_0^m &= \lambda/(L+\lambda) \\ W_0^c &= \lambda/(L+\lambda) + (1-\alpha^2+\beta) \\ W_i^c &= W_i^m = 1/2(L+\lambda) \end{aligned} \tag{A.35}$$

The weights and their associated scale parameters (λ, α, β) are used to incorporate prior knowledge of the distribution of the state or the parameter being estimated.

$$\hat{x}_{k|k-1} \approx \sum W_i^m \chi_{k|k-1}^{(i)} \tag{A.36}$$

$$\bar{y} \approx \sum W_i^m Y_{k|k-1}^{(i)} \tag{A.37}$$

To calculate the Kalman gain matrix, the procedure is to evaluate the covariance matrices. The innovation covariance is:

$$P_k^{yy} \approx R + \sum W_i^c (Y_{k|k-1}^{(i)} - \bar{y}_i)(Y_{k|k-1}^{(i)} - \bar{y}_i)^T \tag{A.38}$$

where Y_i is the outcome of the nonlinear function that maps the sigma point to the observation space, and R is the noise covariance matrix (assuming the observation noise is additive and independent).

The state measurement cross correlation matrix is:

$$P_k^{xy} \approx \sum W_i^c (\chi_{k|k-1}^{(i)} - \hat{x}_{k|k-1})(Y_{k|k-1}^{(i)} - \bar{y}_i)^T \tag{A.39}$$

The final step in UKF is the update step. This involves calculating the estimated output based on the covariance and the cross-covariance matrices:

$$K_k = P_{xy}(P_{yy})^{-1} \tag{A.40}$$

This Kalman gain (K_k) is then used to update the state and covariance matrix:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - \bar{y}) \tag{A.41}$$

$$P_{k|k} = P_{k|k-1} - K_k P_k^{yy} K_k^T \tag{A.42}$$

A.2 System Design

The implemented autopilot is AUAV3 board with the SLUGS II software. This board became popular amongst research applications and is new compared to the SLUGS board or the APM board. The AUAV3 board runs 16-bit software architecture compared to the limited 8-bit used by the APM. Moreover, the AUAV3 board is compatible with the MatrixPilot software and with the SLUGS II software. The SLUGS II is tailored for a 16-bit platform, and it is compatible to use the full capacity of the AUAV3 board.

The advantage of the SLUGS II software is that it supports rapid redesign process. The downside of using the SLUGS II software is that it is relatively new software developed and used only in the ASL lab. Hence, it is not tested as MatrixPilot software or ArduPilot software, and therefore the software is tending to make errors.

A.2.1 QGroundControl

The SLUGS II is responsible for stabilizing the UAV toward a planned destination. The destination is generated by the GCS software, which is part of the

overall generated mission plan. The GCS support the users in generating the requested path and communicating with the UAV software. Most of the GCS utilizes the MAVlink (Micro Air Vehicle Communication) protocol to communicate with the UAV. The QGroundControl is an open-source of a popular GCS which support customized MAVlink messages, and since Mavlink protocol is suited for SLUGS II autopilot, the communication layer has been tailored for the specific messages.

The QGroundControl includes many features that have been adopted and others that have been added to support the Monitoring System. The GCS provides information about the location of fire origin and location of the predicted periphery in real-time. In simulated scenario, it also presents the animated propagated boundary.

The GUI architecture of the QGroundControl includes c++ classes that are called widgets. For the Monitoring System, a widget is customized to present the deployment phase and the pre-planned path for the fleet of UAVs. The customized widget send the measured status to the UAVs, according to there location. In a simulated configuration the measurement is based on the location relative to the polygon of the simulated periphery, in real-time the widget receive a customized message that includes the measured information.

The executed background process is an implementation of the estimation technique and the coordination method. This process is embedded in the QGroundControl. The predicted periphery is presented based on the estimation results, and the UAVs' updated mission plan is being sent as MAVlink messages based on the coordination method calculation.

The connection between the QGroundControl and the SLUGS II software and between QGroundControl and the multi-UAV controller is initially established

when the QGroundControl receive heartbeat messages from the UAV (SLUGS II) and the multi-UAV controller. The heartbeat messages specify which UDP port is open for response message and continuous communication.

A.2.2 Bixler 2 Parameters

The physical parameters of Bixler 2 aircraft are described in Table A.1

Wingspan	59.05 inches
Flying Weight	2 lbs.
Material EPO	Foam
Motor	1400 kV, 160 W
Battery	11.1 V, 2200 mAh, LiPo
Wing Area	2.65 ft^2

Table A.1: RC model plane - the physical parameters of Bixler 2 aircraft.

The wingspan is almost 1 meter, elliptical wings. The electric motor is brushless with a power of 1200kV. The power source battery used is a 2200mAh Lithium Polymer. It provides sufficient current for the power plant, servo motors and the AUAV3 board, through the Electronic Speed Controller (ESC). The ESC provides a 5.0 volt supply to the servos and the AUAV3 autopilot and also provides a control signal and power to the motor. An additional Battery Eliminator Circuit (BEC) is added in parallel to the ESC, to introduce redundancy.

A.2.3 Sensor

An essential part of the system design is the sensing capability. The sensor need the ability to obtain visual information from the environment quickly (e.g., distant temperature changes, local gradient, etc.)

A.3 Results

A.3.1 Flight Tests

After successful completion of the MSIL and MHIL Simulations, flight testing of the coordination algorithm begins. For testing the GUS controller, the airplanes are first stabilized and trimmed for level flight. The safety-pilot then turns on the autonomous mode. In this flight mode, the radio control transmitter works in pass-through mode, enables the pilots to keep controlling the airplanes. The GSC is used to send flight path updates (next WP), while the airplane headed to there on-board destinations (current mission).

The first stage of autonomous flight mode the SLUGS II autopilot gains need to be tuned for changes caused in preparing the hardware from the last tuned configuration. The key tuning parameters for the SLUGS II autopilot are the Proportional, Integral and Derivative (PID) gains of the longitudinal and latitudinal channels (roll, yaw, and pitch).

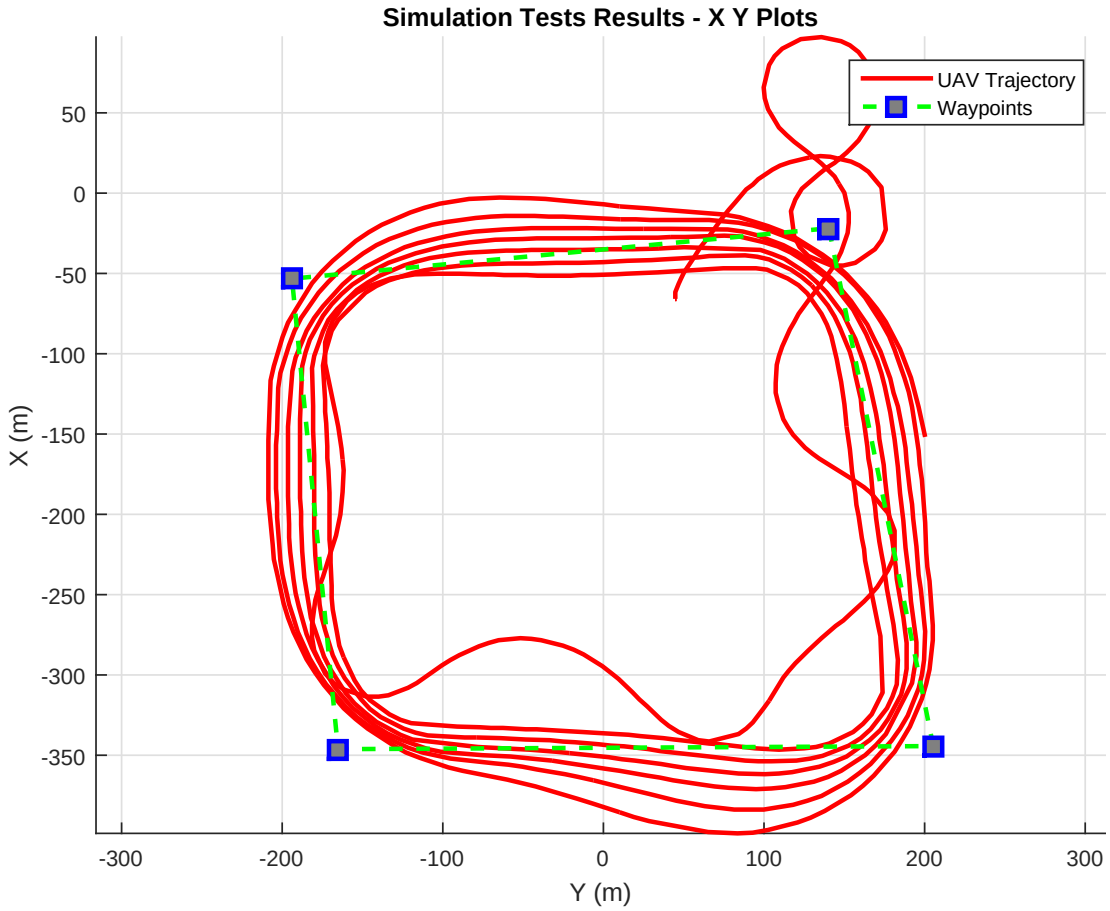


Figure A.4: Simulated scenario with a single UAV is presented. The UAV trajectory is in X Y Cartesian coordinate frame and is relative to the Home position. The first segment of the trajectory starts from take-off controlled manually by the safety-pilot (with the RC), he switched to autonomous mode after 30 seconds. Six laps were tested with different gains setting for tuning the roll command.

A.3.2 Martin Incident

The Martin fire occurred in 2008, and the recorded data were analyzed. The goal was to study the dynamics of the propagated phenomenon thoroughly, identify the dominant factors and their relation. Based on the analyzed boundaries of the map the models employed in this thesis were verified and tuned to have closer dynamics properties. For instance, the spreading rate is bounded, and it is

a function of terrain slope, wind direction, and the vegetation type.

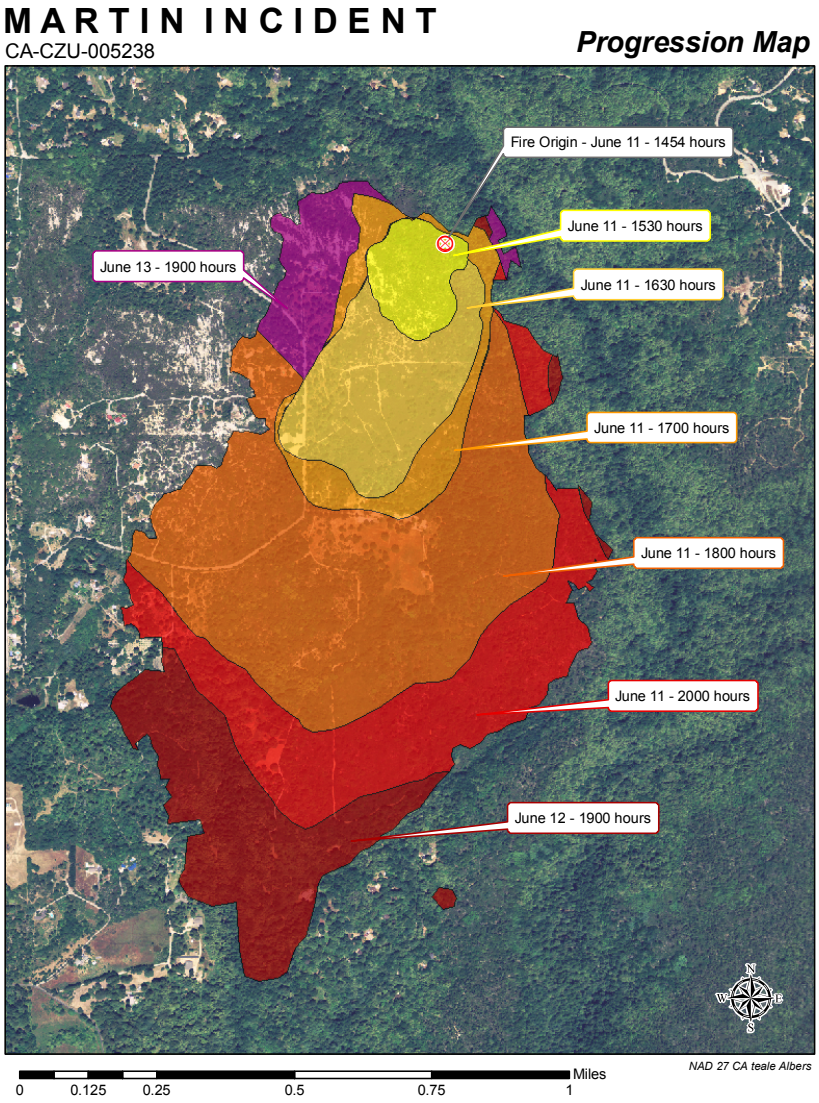


Figure A.5: Progression map of the Martin incident. The image is processed after the incident and relies on number of sources (from Cal-Fire)

Figure A.5 shows the progression map of the Martine incident. The periphery grows up in time, and the boundary of the wildfire spreads. Knowing that the terrain is a dominant factor helps to understand the outcome in relation to the vegetation type, knowing the actual weather explains the direction of propagation.

The incident last for almost two days and spread out over a distance of 2km in a spreading rate of 0.2m/sec.

Bibliography

[1] UAV Over-the-Horizon Disaster Management Demonstration Projects Project Manager : Steve Wegener February 2000 Contents. (February), 2000.

[2] D Adalsteinsson and J.a Sethian. The Fast Construction of Extension Velocities in Level Set Methods. *J. Comput. Phys.*, 148(1):2–22, 1999.

[3] K Alexis, G Nikolakopoulos, A Tzes, and L Dritsas. Coordination of Helicopter UAVs for Aerial Forest-Fire Surveillance. *Applications of Intelligent Control to Engineering Systems*, 39:169–193, 2009.

[4] K Alexis, G Nikolakopoulos, a Tzes, and L Dritsas. Coordination of Helicopter UAVs for Aerial Forest-Fire Surveillance. *Appl. Intell. Control to Eng. Syst.*, 39:169–193, 2009.

[5] Keith. Arnold. Uses of aerial photographs in control of forest fires. *Journal of Forestry*, 49:26–31, 1951.

[6] Nick Arsov. AUAV3.

[7] a Nalyses P Art, Forest Service, Martin E Alexander, David a Thomas, and Dale Bosworth. Fire Management W ILDLAND F IRE S TUDIES AND. *Management*, 63(3):1–96, 2003.

[8] Tuncer Can Aysal, Mark J Coates, and Michael G Rabbat. With Dithered Quantization. *October*, 56(10):4905–4918, 2008.

[9] Randal W Beard. Cooperative forest fire surveillance using a team of small unmanned air vehicles . Int . J . 37(April 2016):351–360, 2006.

[10] W. Randal Beard and W. Timothy Mclain. *Small Unmanned Aircraft: Theory and Practice*. 2012.

[11] J S Bellingham, M Tillerson, M Alighanbari, and J P How. Cooperative path planning for multiple UAVs in dynamic and uncertain environments. *Proc. 41st IEEE Conf. Decis. Control 2002*, 3(December):2816–2822, 2002.

[12] S. Boyd and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. *IPSN 2005. Fourth International Symposium on Information Processing in Sensor Networks, 2005.*, pages 63–70, 2005.

[13] Arthur Earl Bryson. *Applied optimal control: optimization, estimation and control*. CRC Press, 1975.

[14] Fancesco Bullo, Jorge Cortés, and Sonia Martínez. *Distributed Control of Robotic Networks : A Mathematical Approach to Motion Coordination Algorithms*. 2009.

[15] Wolfram Burgard, Mark Moors, and Frank Scheider. Collaborative Exploration of Unknown Environments with Teams of Mobile Robots. *Advances in Plan-Based Control of Robotic Agents*, 2466:52–70, 2002.

[16] CAL FIRE. Cal fire aviation program. Technical Report September, 2007.

[17] CAL FIRE. Strategic Plan Goals. Technical report, 2010.

[18] CAL FIRE. Firefighting Aircraft. Technical report, 2016.

[19] Y.U. Cao, A.S. Fukunaga, A.B. Kahng, and F. Meng. Cooperative mobile robotics: antecedents and directions. *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, 1:226–234, 1997.

[20] David W Casbeer, Randal W Beard, Timothy W Mclain, Randal W Beard, and Timothy W Mclain. Forest fire monitoring with multiple small UAVs Vehicles. 2005.

[21] David W Casbeer, Derek B Kingston, Randal W Beard, and Timothy W Mclain. Cooperative Forest Fire Surveillance Using a Team of Small Unmanned Air Vehicles. 00(00):1–18, 2005.

[22] Haiyang Chao, Yongcan Cao, and Yangquan Chen. Autopilots for small unmanned aerial vehicles: A survey. *International Journal of Control, Automation and Systems*, 8(1):36–44, 2010.

[23] Rama Chellappa, Ashok Veeraraghavan, and Gaurav Aggarwal. Pattern Recognition in Video. *Pattern Recognition and Machine Intelligence*, 3776:11–20, 2005.

[24] Jongeun Choi and Dejan Milutinovic. Tips on Stochastic Optimal Feedback Control and Bayesian Spatiotemporal Models: Applications to Robotics. *J. Dyn. Syst. Meas. Control*, 137(3):30000, 2014.

[25] X Cui, T Hardin, R K Ragade, and a S Elmaghraby. A swarm-based fuzzy logic control mobile sensor network for hazardous contaminants localization. *Mobile Ad-hoc and Sensor Systems, 2004 IEEE International Conference on*, pages 194–203, 2004.

[26] Renwick Curry, Mariano Lizarraga, Bryant Mairs, and Gabriel Hugh Elkaim. L+2, an improved line of sight guidance law for uavs. *Am. Control Conf. (ACC), 2013*, pages 1–6, 2013.

[27] Renwick E Curry. *Estimation and control with quantized measurements*. MIT press, 1970.

[28] Renwick E Curry, Mariano Lizarraga, and Gabriel Hugh Elkaim. The Design of Rapidly Reconfigurable Filters for Attitude and Position Determination. (April):3509, 2010.

[29] Jurek Czyzowicz, Leszek GaÌğsieniec, Adrian Kosowski, and Evangelos Kranakis. Boundary patrolling by mobile agents with distinct maximal speeds. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6942 LNCS:701–712, 2011.

[30] Navid Dadkhah and Bérénice Mettler. Survey of motion planning literature in the presence of uncertainty: Considerations for UAV guidance. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 65(1-4):233–246, 2012.

[31] David A. Goodman. DUBINS PATH PLANNER PROJECT REPORT. (December), 2015.

[32] D.J. Yost J.E. Kain. Command to line-of-sight guidance: A stochastic optimal control problem,. *Journal of Spacecraft,14(7):438444,*, 1977.

[33] Drone Deploy. Preparing for Takeoff Table of Contents. pages 1–44, 2017.

[34] Zhansheng Duan, Vesselin P. Jilkov, and X. Rong Li. State estimation with quantized measurements: Approximate MMSE approach. *Proceedings of the 11th International Conference on Information Fusion, FUSION 2008*, pages 1067–1072, 2008.

[35] Gabriel Hugh Elkaim. System identification for precision control of a wingsailed GPS-guided catamaran. *ProQuest Dissertations and Theses*, 3040012(December):323–323 p., 2002.

[36] Wilfried Elmenreich. An introduction to sensor fusion. *Austria: Vienna University Of Technology*, (February):1–28, 2002.

[37] Mariano I. Lizarraga Fernandez. DESIGN, IMPLEMENTATION AND FLIGHT VERIFICATION OF A VERSATILE AND RAPIDLY RECONFIGURABLE UAV GNC RESEARCH PLATFORM. 2009.

[38] João Fortuna. Search Patterns. 2012.

[39] Walter Gander, Gene H Golub, and Rolf Strebel. Least-squares fitting of circles and ellipses. *Bit*, 34(4):558–578, 1994.

[40] Richard Garcia and Laura Barnes. Multi-UAV simulator utilizing x-plane. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 57(1-4):393–406, 2010.

[41] Edward Geaney. UC Santa Cruz Electronic Theses and Dissertations. 2012.

[42] A.R. Girard, A.S. Howell, and J.K. Hedrick. Border patrol and surveillance missions using multiple unmanned air vehicles. *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, pages 620–625 Vol.1, 2004.

[43] W. H. Greene. Limited dependent variables - truncation, censoring, and sample selection. *Econometric Analysis*, pages 833–902, 2011.

[44] Fredrik Gustafsson and Rickard Karlsson. Generating dithering noise for maximum likelihood estimation from quantized data. *Automatica*, 49(2):554–560, 2013.

[45] Adam Hoover. Analysis of Tracking Systems - lecture-notes, 2015.

[46] Syed a. Imtiaz, Kallol Roy, Biao Huang, Sirish L. Shah, and Phanindra Jampana. Estimation of States of Nonlinear Systems using a Particle Filter. *2006 IEEE International Conference on Industrial Technology*, (5):2432–2437, 2006.

[47] Isabel Ribeiro. Autonomous Systems - class notes, 2004.

[48] Lucas Janson, Edward Schmerling, and Marco Pavone. Monte Carlo Motion Planning for Robot Trajectory Optimization Under Uncertainty.

[49] Simon Julier, Jeffrey Uhlmann, and Hugh F. Durrant-whyte. Technical Notes and Correspondence. *IEEE Transactions on Automatic Control*, 45(3):477–482, 2000.

[50] Simon J Julier and Jeffrey K Uhlmann. A New Extension of the Kalman Filter to Nonlinear Systems. *Spie*, 3068:182–193, 1985.

[51] Simon J. Julier and Jeffrey K. Uhlmann. A general method for approximating nonlinear transformations of probability distributions. *Upublished*, pages 1–27, 1996.

[52] Lubin Kerhuel. Simulink - embedded target for pic.

[53] Do-Myung Kim. Development of near-real-time simulation environment for multiple UAVs. In *Automation and Systems, 2007. ICCAS'07. International Conference on. IEEE, 2007.*, 2007.

[54] Jongrae Kim and Yoonsoo Kim. Moving ground target tracking in dense obstacle areas using UAVs. *IFAC Proc. Vol.*, 17(1 PART 1), 2008.

[55] V. I. Kortunov, O. V. Mazurenko, A. V. Gorbenko, Watheq Mohammed, and Ali Hussein. Review and comparative analysis of mini- and micro-UAV autopilots. *2015 IEEE 3rd International Conference Actual Problems of Unmanned Aerial Vehicles Developments, APUAVD 2015 - Proceedings*, pages 284–289, 2015.

[56] G. Kowadlo and R. A. Russell. Robot Odor Localization: A Taxonomy and Survey. *The International Journal of Robotics Research*, 27(8):869–894, 2008.

[57] Parth Kumar and James E. Steck. System Identification, HIL and Flight Testing of an Adaptive Controller on a Small Scale Unmanned Aircraft. *AIAA Modeling and Simulation Technologies Conference*, (January):1–10, 2015.

[58] L. E. Dubins. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics,*, vol. 79, n, 1957.

[59] Jonathan Las Fargeas, Pierre Kabamba, and Anouck Girard. Cooperative surveillance and pursuit using unmanned aerial vehicles and unattended ground sensors. *Sensors (Switzerland)*, 15(1):1365–1388, 2015.

[60] Thomas W Malone and Kevin Crowston. The interdisciplinary study of coordination. *ACM Computing Surveys*, 26(1):87–119, 1994.

[61] Jan Mandel, Jonathan D Beezley, Lynn S Bennethum, Soham Chakraborty, Janice L Coen, Craig C Douglas, Jay Hatcher, Minjeong Kim, and Anthony Vodacek. A dynamic data driven wildland fire model. *Computational Science*, 4487(1):1042–1049, 2007.

[62] MatrixPilot. MatrixPilot Auto-Pilot, 2016.

[63] P S Maybeck. *Stochastic models, estimation, and control*, volume 1. 1979.

[64] Ivan Maza and Anibal Ollero. Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms. *Distrib. Auton. Robot. Syst. 6*, pages 221–230, 2007.

[65] K McGrattan, S Hostikka, Je Floyd, H Baum, R Rehm, W Mell, and R McDermott. Fire dynamics simulator (version 5), technical reference guide. Volum1: Mathematical Model. *NIST Spec. ...*, 3(Version 5):92, 2004.

[66] William Mell, Anthony Bova, and Glenn Forney. Models for Fire Spread in the Wildland-Urban Interface. pages 1–61, 2012.

[67] J Melorose, R Perroy, and S Careas. *No Title No Title*, volume 1. 2015.

[68] National Interagency Fire Center. Federal Firefighting Costs (Suppression only). page 1987, 2016.

[69] NIFC. National Interagency Coordination Center Wildland Fire Summary and Statistics. Technical report, 2013.

[70] NIFC. Fire Initial Attack. Technical report, NIFC, 2016.

[71] NIFC. Interagency Standards for Fire and Fire Aviation Interagency Standards for Fire. Technical Report January, NIFC, 2016.

[72] N Nigam and I Kroo. Persistent Surveillance Using Multiple Unmanned Air Vehicles. In *IEEE Aerospace Conference*, pages 1–14. IEEE, 2008.

[73] Nikhil Nigam. The Multiple Unmanned Air Vehicle Persistent Surveillance Problem: A Review. *Machines*, 2(1):13–72, 2014.

[74] John Porrill. Fitting ellipses and predicting confidence envelopes using a bias corrected Kalman filter. *Image Vis. Comput.*, 8(1):37–41, 1990.

[75] Sharon Rabinovich. Doctoral Thesis Proposal Multi-UAV Coordination for Uncertainty Suppression. Technical report, UCSC, 2016.

[76] Sharon Rabinovich. A Methodology for Estimation of Ground Phenomena Propagation. 2018.

[77] Ioannis Rekleitis. Cooperative localization and multi-robot exploration. *McGill University, Montreal, Que.,*, 2003.

[78] Wei Ren and Randal W Beard. *Distributed consensus in multi-vehicle cooperative control*. 2008.

[79] Mi Ribeiro. Gaussian probability density functions: Properties and error characterization. *Institute for Systems and Robotics, Technical Report*, (February):1–30, 2004.

[80] Elaine. Rich, Kevin. Knight, and Shivashankar B. Nair. Artificial Intelligence-Rich-Knight-Nair.pdf, 2009.

[81] Arthur Richards and John Bellingham. Coordination and control of multiple UAVs. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, (August):1–11, 2002.

[82] Arthur Richards and John Bellingham. Coordination and control of multiple UAVs. *AIAA Guid. Navig. Control Conf. Exhib.*, (August):1–11, 2002.

[83] Branko Ristic, Daniel Angley, Bill Moran, and Jennifer L. Palmer. Autonomous multi-robot search for a hazardous source in a turbulent environment. *Sensors (Switzerland)*, 17(4):1–17, 2017.

[84] G Ronald. A111D7 IDhnD Technical Note 1611 Fire-Front Propagation Using the Level Set Method VIm7I.

[85] Stuart Russell and Peter Norvig. *Artificial Intelligence A Modern Approach*. 2013.

[86] S. Park, J. Deyst and J. P. How. Performance and Lyapunov stability of a nonlinear path- following guidance method,. *Journal of Guidance, Control and Dynamics, vol. 30, no. 6, p. 1718, 2007.*

[87] K Sain. Book reviews. *Prometheus*, 21(1):120–139, 2003.

[88] Simo Sarkka. Bayesian Filtering and Smoothing. 2013.

[89] Randall C Smith and Peter Cheeseman. On the Representation and Estimation of Spatial Uncertainty. *The International Journal of Robotics Research*, 5(4):56–68, 1986.

[90] H. W. Sorenson. *Parameter Estimation, Principles and Problems*. 1980.

[91] Alexander Stepanov and James Mac Gregor Smith. Modeling wildfire propagation with Delaunay triangulation and shortest path algorithms. *European Journal of Operational Research*, 218(3):775–788, 2012.

[92] a. L. Sullivan. A review of wildland fire spread modelling, 1990-present 3: Mathematical analogues and simulation models. *International Journal of Wildland Fire*, pages 1–42, 2009.

[93] Yang Tang, Huijun Gao, Wenbing Zhang, and Jürgen Kurths. Leader-following consensus of a class of stochastic delayed multi-agent systems with partial mixed impulses. *Automatica*, 53:346–354, 2015.

[94] Victoria State. Fire and Emergency Response Procedures and Training Framework. Technical Report December, 2001.

[95] Eric Wan and Rudolph van der Merwe. The unscented Kalman filter for nonlinear estimation. *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, 2000.

[96] David R. Weise and Gregory S Biging. Effects of Wind Velocity and Slope on Fire Behavior. *Fire Safety Science*, 4:1041–1051, 1994.

[97] Brian Yamauchi. Frontier-based exploration using multiple robots. *Proceedings of the second international conference on Autonomous agents - AGENTS '98*, (May):47–53, 1998.

[98] Angela J. Yu. Natural Computation - lecture-notes, 2010.

[99] Chi Yuan, Youmin Zhang, and Zhixiang Liu. A survey on technologies for automatic forest fire monitoring, detection, and fighting using unmanned aerial vehicles and remote sensing techniques. *Can. J. For. Res.*, 45(7):783–792, 2015.

[100] Zhengyou Zhang. Parameter estimation techniques: a tutorial with application to conic fitting. *Image Vis. Comput.*, 15(1):59–76, 1997.