Fall 12-17-2011

An Information Value Approach to Route Planning for UAV Search and Track Missions

Ryan R. Pitre
University of New Orleans, rrpitre@uno.edu

An Information Value Approach to Route Planning for UAV Search and Track Missions

A Dissertation

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy
in
Engineering and Applied Science

by

Ryan R. Pitre

B.S. University of New Orleans, 2002
M.S. University of New Orleans, 2004

December, 2011

Dedication

This dissertation is dedicated to everybody who I have ever met. Seriously.

Acknowledgments

I would like to give my sincere thanks to my friends and classmates who studied with me while I was an undergraduate. These friends, who played an integral part of my education, are Chris Duggan, Errol Schmitt, Ryan Thiel, Richard Hassell, Zubair Ahmad, Steve Bourg, Edwin Herasymiuk, Cuong Tran, Henry Bodden, Jerry Berndsen, Donald Bonck, George Klink, Miguel Cruz, and Holly McDerment. They all made my undergraduate experience a great one. Despite our many all-night study sessions and countless hours in the engineering labs, we had fun. Between the lakefront BBQs and renting out rooms at bars, I still smile looking back. Those memories will forever be held close to my heart. All of you helped me in ways you'll never know.

I would also like to give my sincerest thanks to Dr. Zhanlue "George" Zhao. His knowledge and understanding helped me through my most challenging year of school. I am forever grateful.

I express my gratitude to Dr. Vesselin Jilkov, Dr. Huimin Chen, Dr. Keshu Zhang, Dr. Jifeng Ru, Dr. Ming Yang, Dr. Anwer Bashi, Dr. Zhansheng Duan, Don Delbalzo, Xiaomeng Bian, and Xiong Lie for all of your helpful suggestions, insightful comments, and time that you have provided. All of your questions during our seminars helped shape my research.

Finally, I would like to thank my major professor Dr. X. Rong Li for maturing me academically and for teaching me the art of critical thinking. It is because of his training that I have truly learned to learn. Thank you.

Contents

List of Figures

List of Tables

Abstract

This dissertation has three contributions in the area of path planning for Unmanned Aerial Vehicle (UAV) Search And Track (SAT) missions. These contributions are: (a) the study of a novel metric, G, used to quantify the value of the target information gained during a search and track mission, (b) an optimal planning horizon that minimizes time-error of a planning horizon when interrupted by Poisson random events, and (c) a modified Particle Swarm Optimization (PSO) algorithm for search missions that uses the prior target distribution in the generation of paths rather than just in the evaluation of them.

UAV route planning is an important topic with many applications. Of these, military applications are the best known. This dissertation focuses on route planning for SAT missions that jointly optimize the conflicting objectives of detecting new targets and monitoring previously detected targets. The information theoretic approach proposed here is different from and is superior to existing approaches. One of the main differences is that G quantifies the value of the target information rather than the information itself. Several examples are provided to highlight G's desirable properties.

Another important component of path planning is the selection of a planning horizon, which specifies the amount of time to include in a plan. Unfortunately, little research is available to aid in the selection of a planning horizon. The proposed planning horizon is derived in the context of plan updates triggered by Poisson random events. To our knowledge, it is the only theoretically derived horizon available making it an important contribution. While the proposed horizon is optimal in minimizing planning time errors, simulation results show that it is also near optimal in minimizing

the average time needed to capture an evasive target.

The final contribution is the modified PSO. Our modification is based on the idea that PSO should be provided with the target distribution for path generation. This allows the algorithm to create candidate path plans in target rich regions. The modified PSO is studied using a search mission and is used in the study of G.

Route Planning, Path Planning, Information Value, Planning Horizon, Particle Swarm Optimization, UAV

1

Introduction

In order to mitigate the loss of human life, many agencies have removed man from the driver's seat and have begun employing Unmanned Aerial Vehicles (UAVs) to serve as the information gathering workhorses to patrol the world's skies. Many of these UAVs are operated remotely and this separation between the operator and the UAV saves lives, which results in reducing the overall risk into a strictly monetary one. Another noteworthy benefit of using UAVs becomes apparent during long surveillance missions. In these situations, pilots on the ground can operate the UAVs in shifts allowing for longer missions. In recent years, the demand for unmanned vehicles has increased and the research community has been called upon to solve the many issues that arise from using unmanned systems.

This dissertation has three contributions in the area of path planning for UAV Search And Track (SAT) missions. These contributions are: (a) the study of a novel metric, G, used to quantify the value of the target information gained during a search and track mission, (b) an optimal planning horizon that minimizes time-error of a planning horizon when interrupted by Poisson random events, and (c) a modified particle swarm optimization algorithm for search missions that uses the prior target distribution in the generation of paths rather than just in the evaluation of them. Three conference papers [39, 51, 52] and a journal article [53] (to appear) have been published reporting the work contained in this dissertation.

The main purpose of this dissertation is to propose an information-theoretic approach to route

planning for SAT missions. The cornerstone of this approach—our objective function—is novel: it integrates naturally the conflicting objectives of target detection, target tracking, and team survivability into a single scalar performance index that quantifies the value of target information. Two distinctive features distinguish this approach from other information theoretic approaches: The integration of the conflicting objectives is much more natural and coherent than existing approaches, and what the objective function quantifies is the *value* of target information, not the information itself. This makes more sense than treating all information equally. When a path maximizes our objective function, the vehicle following that path is expected to gain the most valuable information by detecting the most important targets and tracking them during the most critical times.

In a real mission, the proposed approach to path planning would be used for the initial plan and then again to update the plan using the most up-to-date information. A major task for plan update is the selection of a planning horizon, which defines how far into the future a plan includes. Unfortunately, there is very little research available on the selection of a planning horizon causing planning horizons to be selected either arbitrarily or based on limited computational capabilities. This dissertation proposes an optimal planning horizon that minimizes time-error of a planning horizon when interrupted by Poisson random events. To our knowledge, it is the only theoretically derived planning horizon available making it an important contribution.

The final contribution is the modified Particle Swarm Optimization (PSO) algorithm. It is tailored specifically for SAT missions in that it uses the prior target distribution in the *generation* of paths rather than just in the *evaluation* of them. This is a sensible modification in that the optimization algorithm should be allowed to use all available information when creating candidate solutions. In practice, a major drawback to PSO is that the prior distribution of targets is only used to *evaluate* the search paths rather than to guide the optimization algorithm in the generation of candidate search paths.

The proposed objective function, modified particle swarm optimization algorithm, and the planning horizon are key components of path planning. When used together, they can yield good

path plans for search and track missions while maximizing the expectation of the information value gained.

This dissertation is organized as follows. Chapter 2 provides a literature review in route planning for unmanned systems. In Chapter 3, we define our objective function and explain our approach. Chapter 4 derives an optimal planning horizon that minimizes time-errors when interrupted by Poisson random events. Chapter 5 presents a modified particle swarm optimization algorithm. Finally, we provide conclusions in Chapter 6.

2

Literature Review

2.1 Approaches to UAV Path Planning

Considerable effort has been dedicated to UAV route planning with numerous approaches developed for three primary classes of problems: (a) vehicle routing to visit known locations/targets, (b) search for targets, and (c) search for and then track targets. A variety of optimization techniques have been used to solve the path planning problem including: gradient search [75, 76], evolutionary algorithm [46, 54–56], particle swarm [1, 18, 22, 28, 36, 37, 41, 48, 50–53, 66, 70, 73], neural networks [23] , genetic algorithm [10] , differential evolution [45], ant colony algorithm [77], dynamic programming [3, 16, 17], mixed integer linear programming [5, 6, 61], Eppstein's k-best [4], and myopic approaches [12, 13, 20, 27, 32, 40, 47, 65]. The most commonly used optimization constraints are speed and turning radius. Other constraints that have been incorporated into objective functions include communication, detection range, and safety.

A review of published research in path planning is provided next and is arranged by category. Section 2.1.1 reviews vehicle routing strategies, Section 2.1.2 reviews search strategies, and Section 2.1.3 reviews search and track strategies. This dissertation focuses on search and track missions.

2.1.1 Vehicle Routing Strategies

Most research in UAV path planning has focused on vehicle routing strategies (see, e.g., [4, 6, 10, 12, 13, 15, 18, 22, 28, 37, 44–46, 48, 56, 58, 61, 66, 70, 74–76]), which are primarily concerned with minimizing distance traveled, energy, or time to complete a mission. Vehicle routing approaches are related to the Traveling Salesman Problem (TSP), which is an optimization problem interested in finding the shortest route to visit a set of cities exactly once [49]. The vehicle routing approach of class (a) is similar to TSP with the addition of other constraints and objectives. Fundamentally, vehicle routing is directing a UAV from point A to point B. Gu et al. provided a short study of path selection algorithms in [21] but it is far from complete.

There are three common ways to route UAVs. The first way is to optimize the objective function by determining the best sequence of control inputs to provide to the UAV. These inputs can include changes in heading, altitude, or speed to name a few. The second way to generate paths uses graph theory, which assigns nodes to target locations and then connects those nodes with arcs. The arcs are then assigned a value, which is often based on distance. Then, an optimization algorithm determines the best sequence of nodes to visit based on the arcs' values, hence creating a path. The third way is to grid the environment into equally spaced cells for the UAV to move through like a game piece. The cells are usually square and are small enough to for the cell's properties to be constant. Two examples of cell properties are terrain altitude or threat level.

Obstacle and threat avoidance is a common theme in vehicle routing strategies [4, 6, 10, 13, 18, 28, 45, 56, 61, 66, 77]. In some cases, the threats and obstacles are unknown to the UAVs before the mission and their appearance may require a plan update. Another danger is a collision between UAVs. This can be avoided if they are aware of each other's locations via communication or being equipped with the appropriate sensors.

Some vehicle routing missions require that the UAVs perform tasks such as monitoring, photographing, or killing targets. When multiple UAVs are involved, the route planner must determine how to distribute the workload and it becomes a task assignment problem. Tasks are usually as-

signed to the team member that most benefits the team but this may not always be possible as in the case of limited communication or conflicts. An example of a conflict is when a UAV is only able to complete a single task.

In dynamic environments, communication between UAVs becomes important when there is a probability of failing to perform the assigned task. Team members must be updated with information such as a task's state, in case the task needs to be reassigned. Communication is also useful if a plan changes due to the loss of a UAV or the appearance of new targets or threats.

Another issue involving communication is deciding where the plans are generated. Ideally, a centralized planner would be implemented under perfect communication conditions. This would be optimal because decisions would be made with complete information and each UAV would be directed to act in the best interest of the team. Under limited communication or limited computation constraints, greedier distributed controller strategies have been implemented where each member behaves in its own best interest.

Voronoi diagrams were used in [4] to ensure that the UAV can maximize its distance from the nearest threat by providing path segments that are equidistant from nearby threats. The work of [6] is unique because the UAVs are allowed to begin and end the mission at different times. This is extended by [61] by including a safety constraint and adding a feature that allows the UAV to circle, possibly indefinitely, as it determines its next move. In [15], policies were developed to minimize the amount of time needed to visit pop-up targets. In [58], lower and upper bounds were derived for a multiple depot UAV routing problem. They present a lower bound for the traveling salesman problem by viewing it as a multiple depot UAV routing problem. They show that all of the solutions of this problem are contained in the simplified problem of a constrained forest problem. Then they show that the constrained forest problem can be solved within a certain bounds and the multiple depot UAV routing problem cannot be solved with a lower cost than the constrained forest problem. They also set up the problem to be solved.

2.1.2 Search Strategies

Search missions, as in [3, 7–9, 16, 17, 20, 23, 32, 40, 41, 47, 51, 54, 55, 59, 67–69] require UAVs to locate a possibly unknown number of targets with unknown or partially known locations. The objective for search missions is usually to maximize the probability of detecting the targets. In many search strategies, once a target has been detected, it is no longer considered. A commonly used performance measure is the Cumulative Detection Probability (CDP), which is the probability that the searcher will have detected the target(s) by the end of the mission. CDP is equivalent to the expected number of targets detected during the mission and is the basis for search theory and search techniques. CDP emphasizes the detection of targets but it completely ignores frequent observations of the same targets, which is helpful for tracking.

Search path planning has received considerable attention and some of the works are discussed next, due to their relevance to SAT missions. The work of [7–9] derived the minimum number of times to search a cell. They use a Beta distribution to account for the uncertainty surrounding the probability of a target being in a cell. They also account for the uncertainty of target motion using transition probability matrices. Later, they further account for the uncertainty in the transition probability matrix itself using a Dirichlet distribution and they propagate the distribution in a manner similar to what the particle filter does. In [16, 17], a limited communication cooperative search approach was proposed. They try to maximize a search-to-go gain, which sums the one-step gain over their planning horizon while considering the UAV's survival in addition to the affects of interference by a team member. The work of [41] followed the SAT approach of [65] without considering the tracking problem. They encouraged UAVs either to search new areas or to search old areas from viewing angles orthogonal to the viewing angle that the cell was originally viewed. In [47], a search method was proposed without using a prior target distribution. They modeled their UAVs as state machines with the following states: global search, locate target, approach target, find lost target. The last three states are very similar to each other in that they simply circle the best estimate of the target at some radius. In [54,55], a method was proposed that

assigns and schedules tasks in order to maximize the team's overall score. A special emphasis of this approach is on the communication structure, which they refer to as a market-based approach. It allows each team member to bid on a task and then the task is assigned to the team member that will best improve the team's score. The work of [59] used particle filters to predict the state of a single target with a known initial location and three possible final locations. It attempts to solve the problem by decreasing the relative variance of the distribution by choosing trajectories that minimize the posterior variance of the set of particles. In [69], an approach was proposed for a search and destroy mission under the constraints of limited sensor range and no communication. They search for target with unknown locations and must kill the target. They do this by evaluating the state of the target (alive, damaged, or dead). Because of the lack of communication, they can only guess what their teammates will do. They claim that their results using team theory and no communication is able to outperform an approach with full communication.

2.1.3 Search and Track Strategies

The final category for UAV route planning is the Search And Track (SAT) mission, as in [19, 26, 27, 35, 39, 52, 53, 57, 60, 64, 65, 71, 73], which first finds and locates targets and then attempts to track them for the remainder of the mission. SAT missions are the focus of this dissertation.

Furukawa et al. used a Recursive Bayesian approach in [19] to search and track moving targets. Given the initial number of targets and their prior distributions, they find the set of control inputs that minimize the variance of the targets' states over the planning horizon. Tisdale et al. in [72] extended the work of [19] using a particle filter with improved results.

The approach of Hoffmann et al. in [26, 27] searches for a single stationary target using an information theoretic approach by minimizing the entropy of the target distribution at each time step. A particle filtering technique was proposed to update the state distribution of a target after each measurement, which is then used to calculate the information gained from each measurement. This is useful for search missions because it updates the target's state distribution at each sample

allowing the planner to use the most up-to-date distribution.

In [60], Ryan and Hedrick extended the work of [26, 27] by the addition of a receding horizon. This work proposed a method to approximate the future distribution of the target state using particle filtering. This is useful because it allows a plan to be longer than the single step proposed by [26, 27].

Sinha et al. proposed in [65] a myopic approach that maximizes the information gained from an environment while searching for new targets and tracking detected targets—the next area to search is selected by predicting how much information can be gained by searching that area. This idea of information gain, which inspired our work, is based on the information filter.

A slightly different and simpler approach than [65] was proposed by Sinha et al. in [64] that does not predict the potential gain from undetected targets. It is heavily focused on improving track accuracy without including directly the search objective. However, it is not clear if it can generate search plans because their solutions seem to depend upon following previously detected targets. This was made easy for them since their simulated sensors had a long detection range capable of detecting a target anywhere in the search region.

In [71], Tian et al. proposed a look-ahead policy for autonomous cooperative surveillance. This work uses a layered decision framework rather than integrating multiple objectives into a single objective function. In other words, the objectives are prioritized and then evaluated in order of priority. If the minimum criteria of an objective are not satisfied, the next objective will not be evaluated. In no particular order, their mission objectives are detecting new targets, classification, tracking, and UAV safety.

In the work [35], Lee et al. presented a path planning strategy for a UAV to track a previously detected ground vehicle. The system was tested on a real UAV and was able to track a ground target successfully.

Our SAT work in [39, 52, 53] were authored on the work contained in Chapter 3. Willigen et al. extended our work using simple plan updates that allow the UAV to circle newly detected targets

and then continue with the original plan [73]. They admitted that the updated plans did not always outperform the preplanned paths.

SAT approaches can also be useful for mapping terrain changes over time. The work in [57] proposed an approach for a UAV to search for a river and then track and map its banks. Also, [11] proposed an approach for UAVs to monitor and track the spread of a forest fire.

2.2 Planning Horizons

Very few resources are available on the proper selection of a planning horizon. We have observed that the use of planning horizons is widespread yet they are usually selected because of limited computational resources. The following works only hint on the topic of selecting a planning horizon.

In [34], conditions were provided for which a planning horizon can be determined and then searches the control space to determine when the control input becomes constant. The point in time when the control input becomes constant is considered the planning horizon. Herbon et al. proposed in [24, 25] that information about future events beyond the planning horizon should be used in a plan because it has value. They refer to this window of time beyond the planning horizon as the *effective information horizon* with information in the near future having more value than information farther into the future. This is similar to the idea behind the receding horizon controller [42]. A receding horizon control generates a plan with the intent of only executing a portion of it. The portion of the planning horizon that is executed before a plan is automatically updated is called the *action horizon*. The action horizon usually has a fixed length and is also referred to as the *execution* horizon.

A benefit of using the receding horizon controller over a fixed horizon controller is that it is a means to provide feedback to the controller. This can help UAVs avoid getting trapped in an obstacle field [5]. In [38], Li and Cassandras proposed a cooperative receding horizon controller for a team of UAVs to visit target points. They provide conditions in which their UAVs will visit

all the target sites without actually assigning the sites to the UAVs. In [43], Nabbe pointed out that planning horizons (for obstacle avoidance in unknown terrain) are limited by the sensing capability of the unmanned systems. This is known as the *sensing horizon*. As part of Nabbe's approach to route planning, the vehicles are put in positions that maximize their sensing capabilities. Despite the importance of planning horizons, little research is available to aid in the selection of a planning horizon. That is why the planning horizon proposed in Chapter 4 is an important contribution.

2.3 Particle Swarm Optimization

Particle swarm optimization is a form of evolutionary computation that iteratively improves its population of candidate solutions. Since first published by Kennedy and Eberhart in 1995 [33], PSO has been used to solve countless problems. In its first eight years, more than 300 papers related to PSO were published [29] and many modified versions of PSO are available today. For example, Shi and Eberhart [62, 63] modified PSO and were able to improve its performance in three benchmark problems. They did so by using a fuzzy controller that dynamically adapts the inertia weight in the particle's velocity update equation. See Chapter 5 for an explanation of PSO and its inertia weight.

Among its many applications, PSO is well suited to solve path planning problems [1, 18, 22, 28, 36, 37, 41, 48, 50–53, 66, 70, 73] and a few modified algorithms are discussed next. Sun et al. in [70] proposed using skeletonization to initialize the particle set with reasonable solutions that avoids obstacles. Skeletonization uses a set of vertices similar to that of a Voronoi diagram. In [66], an "anytime algorithm" version of PSO was used. It allowed the optimization of a plan to be interrupted in time for a UAV to avoid an obstacle by using its best candidate path. The work of [36] applied the fuzzy PSO approach of [62, 63] to route planning. Fu [18] proposed a phase angle-encoded and quantum-behaved particle swarm optimization (θ-QPSO) algorithm and then applied it to UAV route planning. Fu gave a detailed explanation of several other modified PSO algorithms and provided a good evaluation of them by comparing them with other forms of

optimization using several benchmarks. The θ-QPSO algorithm outperformed the other forms of PSO, the genetic algorithm, and differential evolution algorithm in the benchmarks and in route planning simulations. To our knowledge, our modified PSO is the only one to incorporate the target distribution into the algorithm itself.

3

UAV Route Planning for Joint Search and Track Missions—An Information-Value Approach

3.1 Introduction

The purpose of this chapter is to propose an information-theoretic approach to route planning for Search and Track (SAT) missions. The cornerstone of our approach—our objective function—is novel: it integrates naturally the conflicting objectives of target detection, target tracking, and team survivability into a single scalar performance index that quantifies the value of target information. Two distinctive features distinguish our approach from other information theoretic approaches: Our integration of the conflicting objectives is much more natural and coherent than existing approaches, and what our objective function quantifies is the *value* of target information, not the information itself. More specifically, our method is superior to the existing information theoretic approaches [26, 27, 60, 64, 65] in several ways, as elaborated next.

First, our objective function integrates the conflicting objectives of target detection and target tracking much more naturally and coherently than the existing approaches. In our objective function, target detection performance is reflected in the number of terms—each term corresponds to one target detected—while tracking performance is dictated by how large each term is. This makes much better sense than the ideas underlying the existing approaches.

Second, we assign a value to the information gain rather than treating all information equally.

The value can be a function of target type, target kinematic state, and time of observation, to name a few. Note that targets are in general not equally important and therefore information from different targets should not be equally valued. The same is true for information obtained at different times. Existing approaches ignore all these important distinctions. For example, our objective function's design parameters can be used to emphasize early detections for time-critical rescue missions or to track targets near strategic locations. These are desirable attributes for an objective function. When a path maximizes our objective function, the vehicle following that path is expected to gain the most valuable information by detecting the most important targets and tracking them during the most critical times.

Another improvement lies in the type of information used and how it is used. The work of [26, 27, 60] tries to maximize Shannon information in order to reduce the uncertainty of the state distribution of a *single unknown target*. Its generalization to the case of multiple targets, or more importantly an unknown number of targets, is not trivial. In contrast, our approach handles any number of targets with ease because our basis is average estimation error for *detected targets* in terms of Fisher information, which is additive and easily obtained from the output of a target tracker. Note that Fisher information is variant w.r.t. the order in which elements are arranged, which is desirable, while Shannon information is not. For example, two independent random variables x and $\hat{x}$ could have the same distribution but vastly different realizations. This is reflected well in Fisher information but not in Shannon information. Fisher information, but not its value, was also used in [64, 65]. Using our information value, a path plan can be generated to provide the right balance of the search and track objectives regardless of the number of targets in the region or the type of sensors used by the UAVs.

Additionally, our approach provides a framework for plan updates. A path update is again path planning except that it is over an updated time horizon and it uses the most up-to-date information, which includes any detections and target tracks as well as the knowledge of regions that have been searched. How the target distribution is updated depends on many things. For example, over the

regions that have been searched without a detection, the spatial density can be reduced to a value depending on the probability of missed detection of the sensor suite onboard the UAVs. In our simulations, for simplicity once a target is detected, it will be tracked using a (Kalman) tracking filter without updating the UAVs' search paths, meaning that they will continue to follow their original, preplanned path regardless of what the target is doing. In a real mission, the proposed approach to path planning would be used for the initial plan and then again to update the plan using the most up-to-date information.

Our objective function also offers valuable flexibility not found in existing approaches.

Next, we reveal two potential drawbacks of our approach. Our objective function is too complex to maximize analytically, making a simulation based approach necessary. Also, it may be difficult to set the design parameters to reflect the relative value of the information obtained from different targets or at different times.

This chapter is organized as follows. In Section 3.2, we define our objective function and describe its parameters. Section 3.3 highlights the desirable properties of our objective function. Section 3.4 modifies our objective function to include a penalty for a loss of a team member. In Section 3.5, we provide several examples of how our objective function can be used. Then, in Section 3.6, we describe an illustrative scenario. Section 3.7 presents the results of the simulation, and finally, Section 3.8 contains our conclusions.

3.2 Objective Function

Our objective function, G, quantifies the value of the target information gained during the mission and is defined as

$$G = E\left[\sum_{k=1}^{K}\sum_{n=1}^{N}\alpha_{n,k}\lambda_{n,k}\text{tr}\left(I_{n,k}\right)\right] \tag{3.1}$$

where

- $E[\cdot]$ is the expectation with respect to all uncertainty, including target distributions and detec-

tions.

- K is the length of the mission measured by discrete time intervals.
- N is the total number of targets detected during the mission, which is random before the mission.
- $\alpha_{n,k}$ is the importance factor for target n at time k.
- $\lambda_{n,k}$ is the time factor for tracking target n at time k once it has been detected.
- tr$(I_{n,k})$ is the trace of the information matrix $I_{n,k}$ for target n at time k.

G assigns a value to the information accumulated during the mission based on when, where, and from which target the information is obtained. The importance and time factors control this and can be set according to the mission planner's goals. To maximize G, we need to select the routes that make more detections (i.e., a large N) and in the meantime track the targets more accurately so that each term is large (i.e., $I_{n,k}$ is large), especially for important targets, such as high-profile targets and major threats, at critical times.

3.2.1 Expectation operator

G is the expectation of the total value of the information gained using a particular solution. By comparing the G values from two different path plans, one can tell, on the average, which of the path plans will yield more valuable information.

3.2.2 Total number of targets detected, N

The most unusual aspect of G is that the number of terms, N, in the second summation is path dependent and ***random***—it is equal to the number of targets to be detected. This is so due to the uncertainty associated with the distribution of targets and missed detections. Maximizing the number of detected targets is not critical when maximizing G because maximizing formula (3.1) is not equivalent to maximizing N.

3.2.3 *Importance factor,* $\alpha_{n,k}$

It is a design parameter that is necessary when all targets are not equally important. The importance factor depends on what is known about the targets and may also be a function of a target's kinematic state, such as location or velocity. For example, a target may be more valuable in one location than in another; a moving target could be more important than a stationary target. The importance factor can be time dependent—a particular target may become more valuable at a particular time. If a solution with a high CDP is more desirable than solutions with quality tracks, the importance factor can be used to reduce the benefit of multiple observations on a target.

3.2.4 *Time factor,* $\lambda_{n,k}$

It is used to specify how valuable information is as a function of time. It can control when it is most important to know where the targets are—e.g., it can reduce the value of information with age.

3.2.5 *Information matrix,* $I_{n,k}$

It quantifies the accuracy of the estimates and is the *fused* information matrix for target n at time k from *all* of the UAVs. In the Fisher sense, information is the inverse of uncertainty and a reduction in uncertainty results in an increase in information [2]. By improving the accuracy of the state estimates, we are increasing the amount of information about the target. The value of this information is determined by $\alpha_{n,k}$ and $\lambda_{n,k}$.

3.3 Desirable Properties of G

3.3.1 *Jointly optimizes detection and tracking*

Jointly optimizing conflicting objectives is quite challenging [30, 31]. Our objective function (3.1) contains elements of both detection and tracking and can be used to optimize these objectives

$jointly$.

3.3.2 Differentiates information from information value

Fisher information only quantifies the accuracy of a track without considering the usefulness of the information. G assigns value to the information according to when, where, and from which target the information was obtained.

3.3.3 Easily compares different solutions

When using our objective function to evaluate different solutions, the best solution is the one with the largest value. This greatly simplifies determining which path, among many candidate paths, is the best solution for the problem at hand.

3.3.4 Naturally prefers early detections

G naturally encourages early detection of targets. When targets are detected early in a mission, they can be tracked for a longer time, which will increase the total amount of information gained from those targets. This is supported by the (Kalman) tracking filter's ability to continue tracking a target after the last time it was observed. In general, early detection is preferred since it gives more time for response.

3.3.5 Encourages repeated observations of the same targets

It is beneficial to make multiple observations of the same target so as to track it more accurately. G encourages tracking targets for a longer time since it will lead to a larger gain in information value as a result of more terms in the summation.

3.3.6 Useful for ranking solutions of many practical problems

G quantifies the joint objectives of detection and tracking naturally and coherently. It is flexible and can be used to rank solutions to many related problems well, such as sensor management—by determining the best locations to take measurements over time—and resource allocation. In fact, route planning itself is essentially sensor placement over time.

There are numerous possible enhancement techniques that can be ranked well by G including, for example, revisiting the same target, optimizing the field of view, and viewing targets from different angles. Also, G can handle different numbers of UAVs, different platforms, different sensors types, and different communication schemes. All these enhancements can be applied without changing our overall approach to path planning.

3.4 Accounting for Team Survival

Our objective function indirectly promotes team survival because UAVs that live longer will have more opportunities to detect and track targets. Formula (3.1) reflects this without directly accounting for the loss. This is because G only keeps track of what is gained in terms of information value and does not penalize for the loss of a UAV directly. As a result, G does not directly care about how many UAVs survive. A formula that explicitly accounts for the loss of a team member is presented next.

Here, a term is added to formula (3.1) that penalizes the team for the loss of team members:

$$G_s = E\left[\sum_{k=1}^{K}\sum_{n=1}^{N}\alpha_{n,k}\lambda_{n,k}\text{tr}\left(I_{n,k}\right) - \sum_{m=1}^{M}G_{m,k_m}\right] \tag{3.2}$$

Here, M is the total number of UAVs lost during the mission, which is random before the mission. G_m is the cost of the searcher in terms of information value even though it may be difficult to quantify. When there is no knowledge about G_m, we may choose it to be equal to the difference in

the contributions of the UAV if it is lost or not, which can be obtained by a simulation-based study.

3.5 Examples and Analysis

In this section, we include five simple scenarios to illustrate how different solutions score when using G. There are no actual UAV paths in these examples. Instead, detection sequences were assigned to the targets, as will be explained. These examples will highlight the desirable properties of G. In each figure, the solution that tracks multiple targets will be plotted using a blue solid line and the solution that tracks a single target will be plotted using a red dot-dash line.

In all the examples, the targets will move at a constant velocity (CV) in a two dimensional space and we use a Kalman filter with a matching CV motion model to track the targets. Each example takes place over a 100-s time period and is sampled at a rate of 10Hz for a total of 1000 samples per scenario. These examples are easily reproducible because the estimation error covariance does not depend on measurements. In fact, no measurements were simulated for these examples.

The motion and measurement models are

$$\begin{aligned} x_k &= Fx_{k-1} + Gw_{k-1} \\ z_k &= Hx_k + v_k \end{aligned} \tag{3.3}$$

with w and v being zero-mean Gaussian white noise with covariances Q and R, respectively, and

$$F = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad G = \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix}$$

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \frac{m^2}{s^4}$$

$$R = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix} m^2, \quad T = 0.1s$$

When viewing the figures in this section, the Φ_k label on the y-axis is the trace of the information matrix at sample k:

$$\Phi_k = \text{tr}(I_k),$$

to reflect how well the targets are being tracked. However, if multiple targets are included, then

$$\Phi_k = \sum_{n=1}^{N} \text{tr}(I_{n,k}),$$

unless otherwise stated. Furthermore, we always set $\text{tr}(I_{n,k}) = 0$ before target n has been detected. Lastly, the legend in each figure shows each solution's score in terms of

$$G = \sum_{k=1}^{1000} \Phi_k.$$

Thus, G can be thought of as the area under the Φ_k curve.

3.5.1 *G prefers early detections to late detections*

This example shows that it is better to detect and track a target early rather than later in the mission when $\alpha_{n,k}$ and $\lambda_{n,k}$ are set to unity. In Figure 3.1, two sets of scores are shown. The first solution (blue solid line) detects a target at sample number 200 and then observes it until sample 700. The second case (red dot-dash line) is the first case delayed by 200 samples. In both cases, the target is observed and tracked for 500 samples and then tracked using prediction until sample 1000. Note that without the observations, the value of Φ_k decreases but is still positive valued because it is estimating the target's state using prediction.

The blue solid line solution has a higher score of $G = 16,848$ while the red dot-dash line solution only had a score of $G = 16,057$. This is because the blue solution was able to predict the target's state for 300 samples after the last observation was made, as apposed to the red solution's 100 samples.

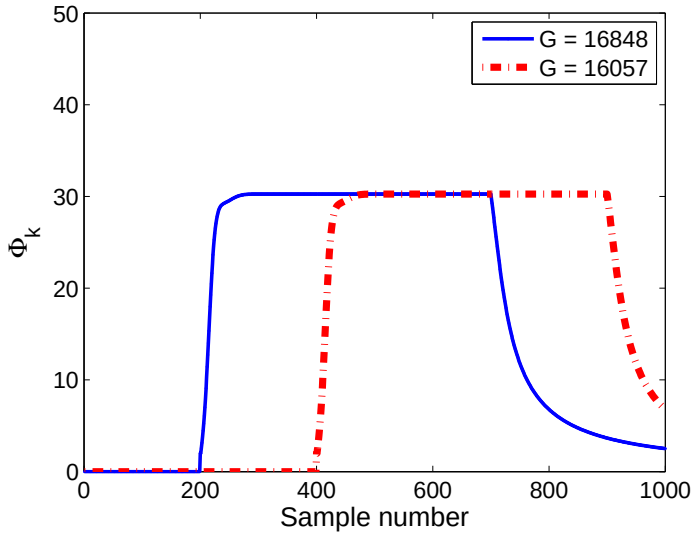

Figure 3.1: Normal case: early detection is better

3.5.2 *Use time factor to encourage late information gain*

This example uses the same scenario as in Figure 3.1 but changes the time factor. See Figure 3.2 for the time factor vs. sample number. For clarity, λ_k in Figure 3.2 is the same λ_k as in formula (3.1), meaning that information gained towards the end of the mission is more valuable. Figure 3.3 shows the time factor applied to the scenario from Figure 3.1. Here,

$$\Phi_k = \lambda_k \text{tr}(I_k).$$

In Figure 3.3, the later solution has a greater G value showing that $\lambda_{n,k}$ can be used to encourage later information gain, if desired.

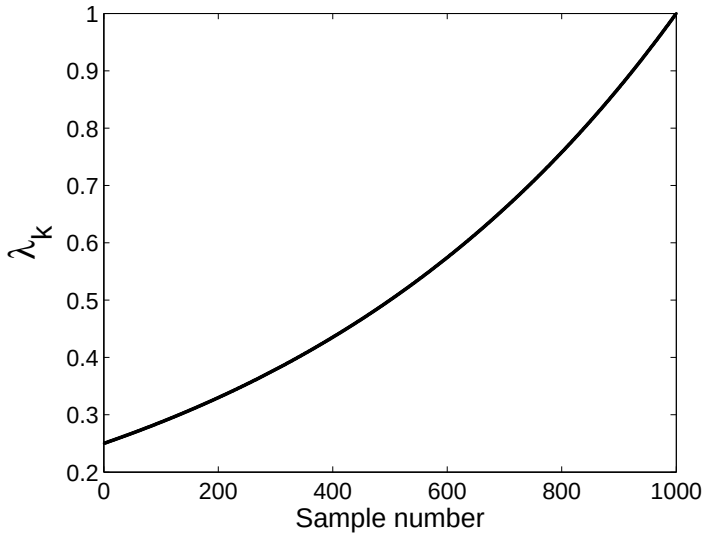

Figure 3.2: Time factor vs. time for Section 3.5.2

3.5.3 *G prefers multiple targets to a single target*

Next, we compare tracking two targets to tracking a single target when the total number of observations is fixed. This will demonstrate that it is better to distribute the fixed number of observations over multiple targets than to focus on a single target. For simplicity, we set $\alpha_{n,k}$ and $\lambda_{n,k}$ to unity.

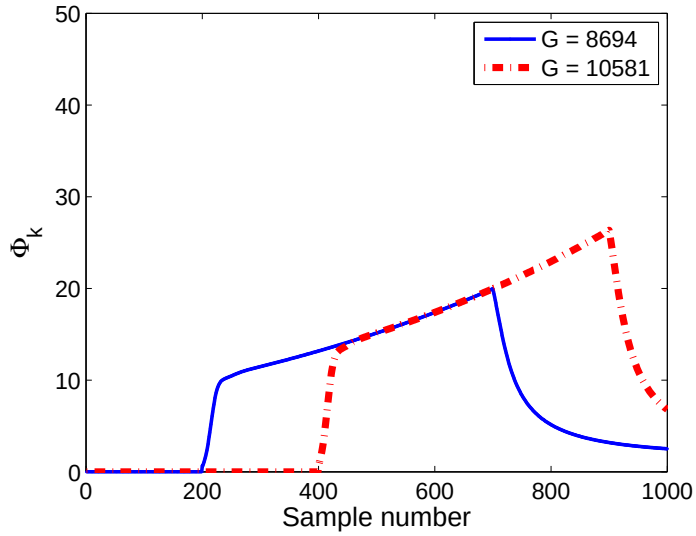

Figure 3.3: Use time-increasing time factor to encourage later detection

Figure 3.4 plots the gain in information value vs. time and shows that G prefers detecting multiple targets to detecting a single target. Here, we plot the multiple target case in blue with

$$\Phi_k = \sum_{n=1}^{N} \text{tr}(I_{n,k}).$$

At sample 100, a searcher detects and actively tracks a target until sample 200. Between samples 200 and 400, the searcher does not observe the target and tracks it using prediction.

At sample 400, the searcher can either begin tracking a new target (blue solid line) or it can reacquire the original target (red dot-dash line). In both cases, the second sequence of observations will continue until sample 600. As seen in Figure 3.4, the two cases are equal until sample 400 when Φ_k becomes greater for the multiple targets case. This is due to the (Kalman) tracking filter's ability to continue tracking the first target using prediction.

3.5.4 Use importance factor to encourage detecting more targets

In some situations, it is possible for a solution that continuously tracks a single target to have a higher G value than a solution that tracks multiple targets for a shorter time. This is because con-

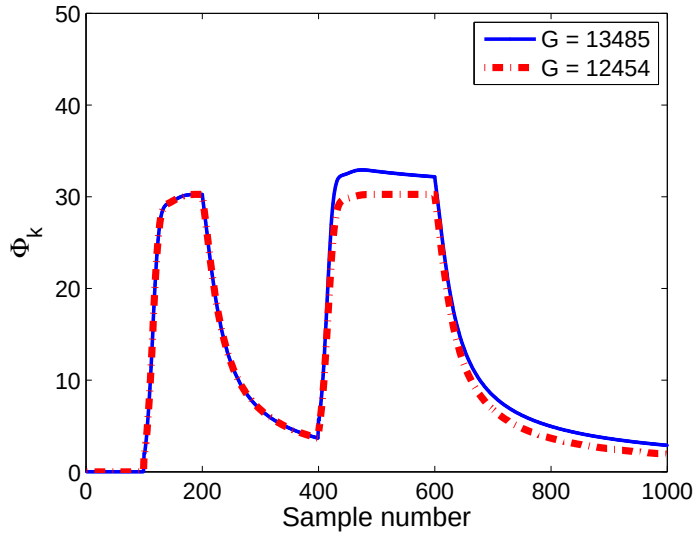

Figure 3.4: Normal case: G prefers detecting more targets

tinuously tracking a single target can result in many more positive valued terms in the summation of formula (3.1). Figure 3.5 shows the information gained from continuously tracking a single target (red dot-dash line) compared with a solution that briefly tracks two different targets (blue solid line). If we want to overcome this, $\alpha_{n,k}$ can be used to reduce the importance of a target once it has been detected so as to encourage more detections.

Figure 3.6 shows the effect that an importance factor has when applied to the scenario depicted in Figure 3.5, which shows the information gained from continuously tracking a single target. Here,

$$\Phi_k = \sum_{n=1}^{2} \alpha_{n,k} \text{tr}(I_{n,k}),$$

where $\alpha_{n,k}$ was arbitrarily designed to reduce the importance of a target after its first detection. This is an example of how $\alpha_{n,k}$ can influence UAVs to track more targets.

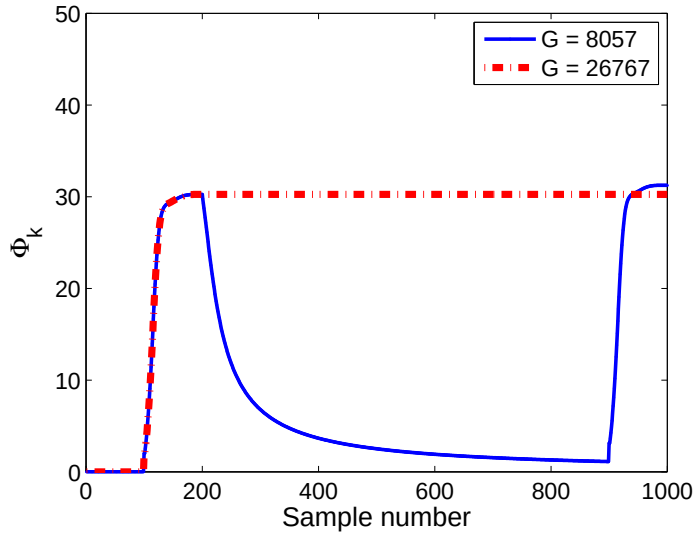

Figure 3.5: A case that prefers better tracking to more detections

3.6 Simulation

We begin this section with some simplifying assumptions. We describe the terrain, the search vehicles, and their sensor's detection performance followed by a description of the Kalman filter as a target-tracking algorithm. We then describe the parameters used in our objective function. Next, we describe several sets of search paths. Finally, we explain how we evaluate G. Section 3.7 contains the results of the simulations.

3.6.1 Simplifying assumptions

The main contribution of this chapter is the proposal of an information-value approach based on a novel objective function. Applying it to a real-world problem would involve many practical issues. While those that are unavoidable and will affect our overall approach significantly should be considered and discussed here, the others need not and should not be discussed, even though they may affect the results of our approach. In order to focus on the main theme—how the method works—it is better to simplify things for clarity so as to avoid distracting the reader from the real purpose of the chapter. Thus we do not include some of the practical issues that may be

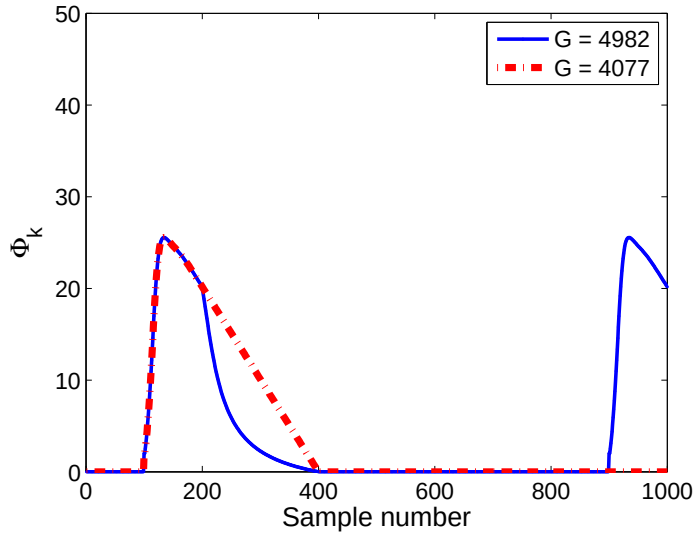

Figure 3.6: Use importance fact to encourage more detections

encountered in a real mission, although they are important. Also, including these issues would complicate things unnecessarily and make it hard to apply our common sense in the judgment and analysis of the outcome. In addition, space limitation would not allow us to discuss such issues in detail.

Therefore, we assumed perfect communication and used centralized fusion during the simulation. We also assume some prior knowledge of the distribution of target locations rather than assuming that their locations are known. For a fair comparison between the optimized paths and the ladder pattern solutions, we do not update the path plans. Finally, for simplicity we set both $\alpha_{n,k}$ and $\lambda_{n,k}$ to unity for all targets. Furthermore, our simulation did not consider the problem of false alarm, data association, or model mismatch. These problems are not present in all situations and their incorporation would complicate things substantially. In fact, their treatment is not trivial and calls for separate studies, which is beyond the scope of this chapter.

The simulation setup presented here is general enough to handle most practical issues discussed above, but doing so would not change the overall approach; rather, it would complicate things and make it harder for us to make sense out of the results. For example, changing the sensor, the UAV

properties, target distribution and behaviors, etc. would change the outcome but the method itself would remain the same. We chose to use a simple sensor, because using a more sophisticated sensor model would not change the approach. Our target model is simple yet reasonable in that every terrain type received its own percentage of the targets. Changing it would also not change the approach.

3.6.2 Terrain

We simulate a square region that is 60 nautical miles (nmi) wide. The resolution is 162 cells by 162 cells with each cell having its own terrain type and altitude. Figure 3.7 shows the environment contains three terrain types: mountain, desert, and forest. In the colorbar, the different colors labeled "Mount." indicate different altitudes of the mountainous terrain. The region also contains a road network, shown by grey lines. The altitudes in this region are shown in Figure 3.8 and they vary from 2808ft to 9295ft.

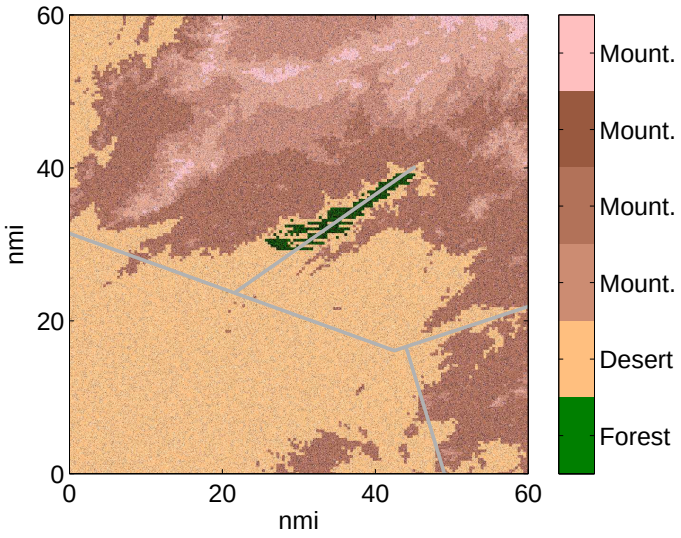

Figure 3.7: Terrain map of the search region

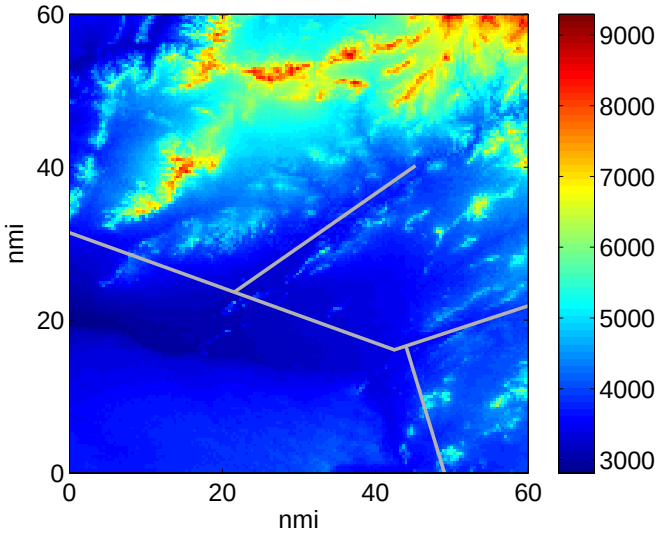

Figure 3.8: Altitude of the search region in feet

3.6.3 Search vehicles

The search team consists of two UAVs. Searchers 1 and 2 begin with the initial coordinates of (1.25, 1.25)nmi and (58.75, 1.25)nmi, respectively. They will navigate the region at a constant altitude of 9,842.5ft (3000m) above sea level so that there is no concern of colliding with the terrain. The UAVs fly at a constant speed of 100kt and are limited to a maximum turning rate of $2deg/s$.

3.6.4 Sensor model and detection function

The searchers use a simple optical sensor aimed directly below the vehicle and takes a snapshot every 10s. At each discrete-time step, an image is captured for the purpose of detecting targets. The sensor has a viewing angle, θ, set to be about 51deg in every direction. This viewing angle is sufficient for the searcher to detect a target up to 2nmi away while flying at 9,842.5ft above the target.

To determine if a target is within the detection range of a searcher, we need to determine the

difference in altitude between the target and the searcher, Δ_{alt}. A target cannot be detected if the sensor's maximum viewing angle θ is smaller than the angle to the target, θ_t, given by

$$\theta_t = \tan^{-1}\left(d/\Delta_{alt}\right) \tag{3.4}$$

where d is the horizontal ground distance between the searcher and the target.

The detection function is used to determine the probability of detecting a target within the sensor's field of view. The detection function is expressed as

$$P_D = \begin{cases} P_{D,\text{terrain}} + \delta_r B_r, & \theta_t < \theta \\ 0, & \theta_t > \theta \end{cases} \tag{3.5}$$

It is the sum of the probability of detection associated with the target's terrain, $P_{D,\text{terrain}}$, and the probability bonus B_r, because it is assumed easier to detect an on-road target. The binary operator δ_r is

$$\delta_r = \begin{cases} 1, & \text{Target is on a road} \\ 0, & \text{Target is off the road} \end{cases} \tag{3.6}$$

See Table 3.1 for the probability of detection associated with each type of terrain. The values consider the possibility of occlusions and the target blending in with its surroundings.

Table 3.1: Target detection probabilities

	$P_{D,\text{terrain}}$	$Road\ Bonus$	$P_{D,\text{terrain}}$ $+Road\ Bonus$
Desert	0.90	0.05	0.95
Mountain	0.50	0.25	0.75
Forest	0.10	0.40	0.50

3.6.5 *Target distribution and description*

The probability that a target is located in a particular terrain type follows the distribution in Table 3.2.

Table 3.2: Percent of targets in each type of terrain

Terrain Type	% of targets
Mountain	90
Desert	7
Forest	2
Road	1

Figure 3.9 illustrates the probability density for the distribution of a target's location. The percentages from Table 3.2 are uniformly distributed over their respective regions. The numbers appear to be very small because there are a total of $162 \times 162 = 26,244$ cells used to create the map. Summing the values from each cell results in unity.

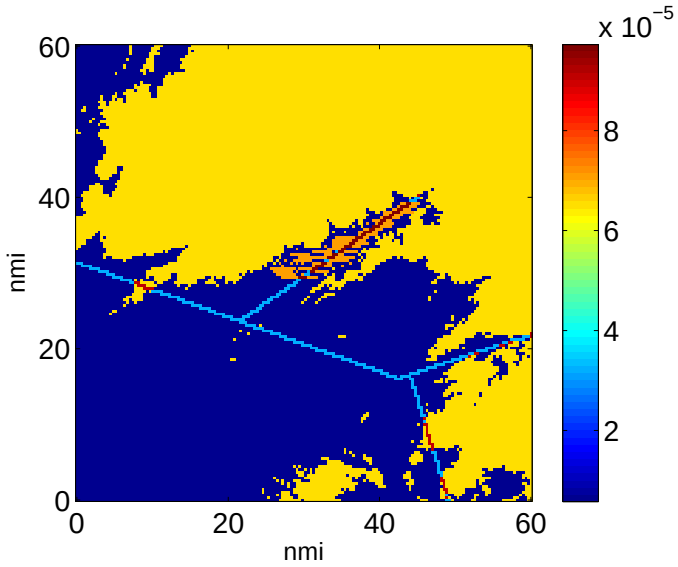

Figure 3.9: Probability density for a target's location

The targets move at a constant speed of 35kt on a road and travel off the road more slowly,

with a speed given in Table 3.3. Road targets are allowed to leave the area when they encounter the edge of the region and they are replaced with a new target headed in the opposite direction. If an off-road target encounters the edge of the region, it turns around at some random angle. These steps guarantee that there is a constant number of targets in the region at all times. Furthermore, the targets remain in their initial terrain type. Otherwise, the targets would leave the target rich terrain types, which would cause the distribution of targets to vary over time.

Table 3.3: Target speed (knots) in various types of terrain.

Terrain Type	speed (kt)
Road	35.00
Desert	33.25
Mountain	8.75
Forest	3.50

3.6.6 Target tracker

Even though the targets can change directions as described above, for simplicity, we used a Kalman filter with the nearly constant velocity motion model to track the targets and to gain information once the target has been detected. The motion and measurement models used in the Kalman filter are given by system (3.3) with

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \frac{nmi^2}{hour^4}, \quad R = \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_v^2 \end{bmatrix} ft^2, \quad T = 10s.$$

Figure 3.10 shows the standard deviation of the position measurement error along each axis vs. Δ_{alt}.

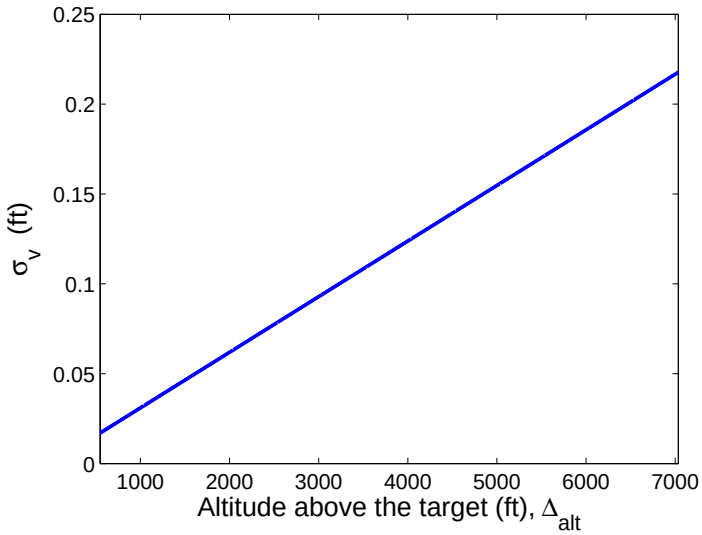

Figure 3.10: Measurement noise vs Δ_{alt}

3.6.7 Search paths

For this scenario, we evaluate the expected performance of five sets of paths using our objective function. Three of them were generated using a modified particle swarm optimization algorithm and the other two are simple ladder pattern solutions commonly used in search missions.

Optimized search paths without threats

This section does not suggest that our modified particle swarm optimization (PSO) algorithm must be used to optimize G. Other path optimizers can use G as their objective function. This section is meant to display how G assigns value to different solutions in a realistic setting. A description of the algorithm used to generate these solutions is provided in Chapter 5 and is summarized next.

PSO is a form of evolutionary computation that attempts to improve its solution iteratively. Our modified PSO uses the prior distribution of the targets' spatial density not only to evaluate the solutions but also as a guide in creating the solutions. To do so, a population of targets is generated. Then, a target path is generated for each of those targets and it is sampled at the same

times that the UAVs' paths are sampled. So, for every discrete-time location the UAV has, every target will have a corresponding location. Then, the modified PSO will have every target pull the UAVs toward its corresponding sampled position, which means that the UAVs will be drawn to the regions of the map with a high target density although those regions are time varying. This force of attraction for each of those samples has a magnitude proportional to the inverse square of the distance between the searcher and the target. These forces, in addition to the traditional terms in the PSO update equations, make up our modified PSO. This results in more path plans being created in the target rich regions than elsewhere. Once a generation of search paths is created using the modified update equations, G is used to evaluate them. Details of our modified PSO algorithm can be found in Chapter 5 or [51]. For further reading about PSO, see [33], [14], and [29].

The path plans in Figures 3.11 and 3.12 were generated using our modified PSO algorithm. To optimize each of these plans, we used 256 particles for 20 iterations, which required about one hour of computation time. To reduce the computation time, the sample rate of each sensor was reduced to 120 samples/hour during the optimization, which is below the 360 samples/hour used in the scenario. This reduces the dimension of the problem. Then, once the optimizer has finished running, the best path will be upsampled to the rate of 360 samples/hour. To avoid overfitting the paths to a particular realization of targets, the paths were generated using six 500-target sets. Three of the six sets were used for evaluation and the other three sets were used as guides. For both cases, one of the three sets was randomly selected for each iteration. These targets are only used by the optimization algorithm and a larger set of targets will be used to provide the average G value for the Results section.

After a path is generated but before it is evaluated, it is tested for any violations of its motion constraints. If the speed is not a constant 100kt or if the heading changes by more than 2deg/s, the path is corrected. This process is time consuming. The computation time could again be reduced by improving the path correction process but the means of optimization is not the focus of this work.

The difference in how Figures 3.11 and 3.12 were generated is their use of the importance factor. Figure 3.11 was optimized with an $\alpha_{n,k}$ equal to unity. This type of setting encourages the searchers to track the targets more accurately by spending more time loitering over targets. Maintaining longer visibility on a single target conflicts with detecting new targets and requires the necessary tradeoffs. The UAVs travel faster than the ground targets and the optimization algorithm handles the issue of maintaining visibility automatically. The path plan that provides the most valuable information will maximize G by satisfying these conflicting objectives.

Figure 3.12 was generated by reducing the importance factor of a target after each time it is observed. This causes G to prefer solutions that detect more targets by discouraging too many observations of a single target. For Figure 3.12, $\alpha_{n,k}$ was set to 1% of its initial value $\alpha_{n,0}$ after 10 observations:

$$\alpha_{n,k} = \eta^{d_{n,k}} \alpha_{n,0}$$

where $\eta \approx 0.631$ from $\eta^{10} = 0.01$ and $d_{n,k}$ is the total number of observations on target n before time k.

The path plans in Figures 3.11 and 3.12 focus on searching the low lying, mountainous region because of a combination of two factors: (a) there is an abundance of targets in the mountains and (b) detection range in low lying areas is better than in high altitude regions.

Optimized search paths with threats

Figure 3.13 is a path plan generated for the same environment with threats added. These threats are assumed to be surface-to-air missile launchers with known locations. The paths were generated using formula (3.2) with G_m set to 20,000. The threats, marked in red, are guaranteed to kill a UAV when within 3nmi of it and zero chance of killing the UAV outside of that range. The threats were strategically placed in the low lying areas to encourage the UAVs to search elsewhere.

Ladder pattern search paths

Ladder patterns are commonly used search paths and are simple to generate. See Figures 3.14 and 3.15 for two examples. Figure 3.14 shows a pair of ladder patterns that use the prior knowledge of the possible target locations. This solution was generated to search for targets in the mountain where 90% of the targets are known to be located. Figure 3.15 shows a pair of ladder patterns generated without the prior knowledge of the targets' spatial distribution.

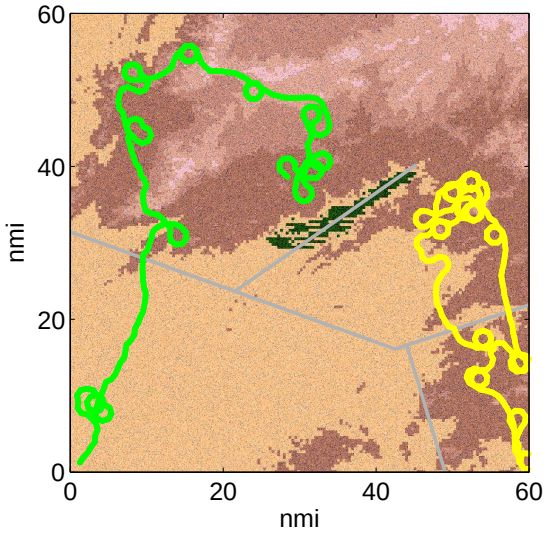

Figure 3.11: Optimized search paths with unity importance factor.

3.6.8 Evaluating G

G is too complex to maximize analytically, making a simulation based approach necessary. The following steps were taken to evaluate G: 1) Simulate the searchers following their specified paths. 2) Determine which targets have been detected. This will create a sequence of observations. 3) Input the coordinates of the observations into the target tracking algorithm and an output of the target tracking algorithm is the information matrix. Every target will have its own information matrix at each time-step. 4) Use the set of information matrices to evaluate formula (3.1).

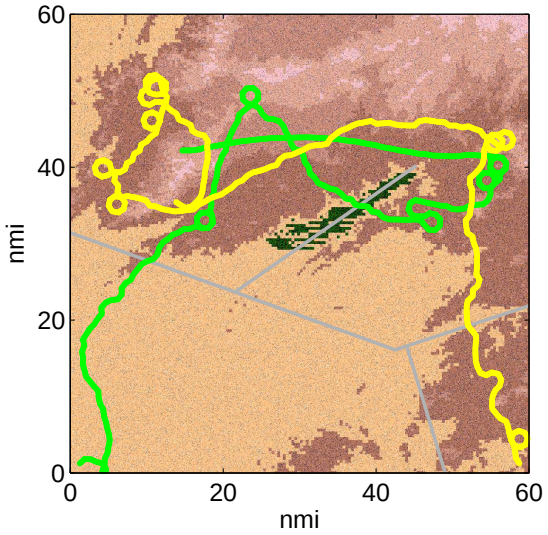

Figure 3.12: Optimized search paths with an importance factor that decreases once a target is detected.

3.7 Results and Discussion

In this section we present the results for the five sets of search paths for the two-hour mission described in Section 3.6. The results from this simulation are presented in Table 3.4. The values were obtained by averaging over 100 runs using 1000 targets per run for a total of 100,000 targets. Our objective function is linear in this regard, so the following cases are equivalent (all using the same target distribution): 1) 100 runs with 1,000 targets, 2) a single run with 100,000 targets, and 3) 100,000 runs with a single target.

The "G_1" column contains the average value of our objective function using a constant importance factor to encourage tracking. The "G_2" column was calculated using an importance factor that decreases after each time it is detected. The "Dets." column indicates the average number of unique targets detected per run. Dividing the number in the detections column by 1000 is equivalent to the cumulative detection probability (CDP). The "Obs." column contains the average *total number* of observations, including repeated observations of the same target. The final column is

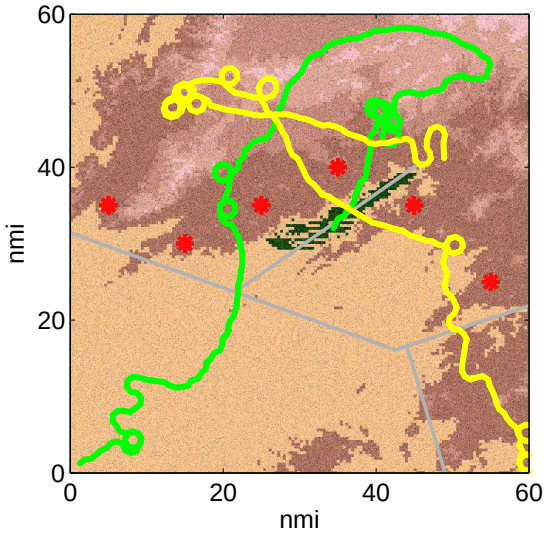

Figure 3.13: Optimized search paths with unity importance factor and six threats (shown in red) having a kill radius of 3nmi.

Table 3.4: Tabulated results for the SAT simulation.

	G_1 (10^{12})	G_2 (10^{11})	Dets.	Obs.	$\frac{obs.}{target}$
Optimized with $\alpha = 1$ (Fig. 3.11)	5.2	5.8	203	996	4.9
Optimized with decaying α (Fig. 3.12)	4.7	6.5	251	1015	4.0
Optimized with $\alpha = 1$ and threats (Fig. 3.13)	4.4	5.6	202	834	4.1
Ladder with prior (Fig. 3.14)	3.7	6.1	270	917	3.4
Ladder without prior (Fig. 3.15)	2.5	4.3	195	663	3.4

the average number of observations per target, which is the total number of observations divided by the number of targets detected.

The PSO solutions had the most observations per target because they spent more time loitering in the low lying, mountainous regions, which resulted in the best combination of detection range and the abundance of targets. Both factors are favorable conditions for tracking targets, especially because they make it easier for the searchers to reacquire previously detected targets. Once stated, these reasons become obvious but optimization algorithms often generate solutions using environmental relationships that may not be considered beforehand or even overlooked. Ladder patterns

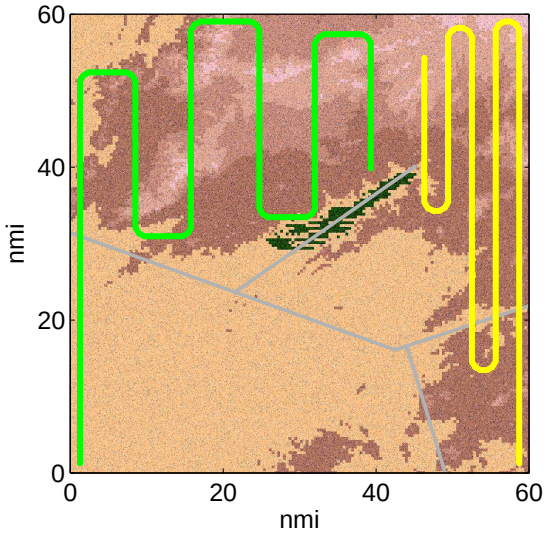

Figure 3.14: A conventional ladder search path with prior

often overlook or neglect these relationships.

Column G_1 shows that the solution in Figure 3.11 has the highest average value for G when a constant important factor is used. This was expected and it explains why Figure 3.11 averaged the most observations per detected target. According to G_1, this solution performed best because it distributed its observations over fewer targets in order to improve tracking performance.

Table 3.4 also shows that Figure 3.12 maximizes G_2 when the importance of a target is reduced after it is detected. Even though Figure 3.12 detected 19 less targets than the informed ladder patterns of Figure 3.14, the team made 98 more observations and that is why its G_2 value is greater than that of the informed ladder. When comparing the total number of detections, the solutions in Figures 3.12 and 3.14 fared best because they were designed to favor detections. The fact that Figure 3.12 maximizes G_2 is in line with Section 3.5.4 that the importance factor can be used to find more targets.

Figure 3.13 contains the only path plan facing threats. The placement of the threats forced the search team to fly in between the threats to search the low-lying, target rich regions of the map. The modified PSO algorithm chose to avoid the threats even though the 20,000 loss in information

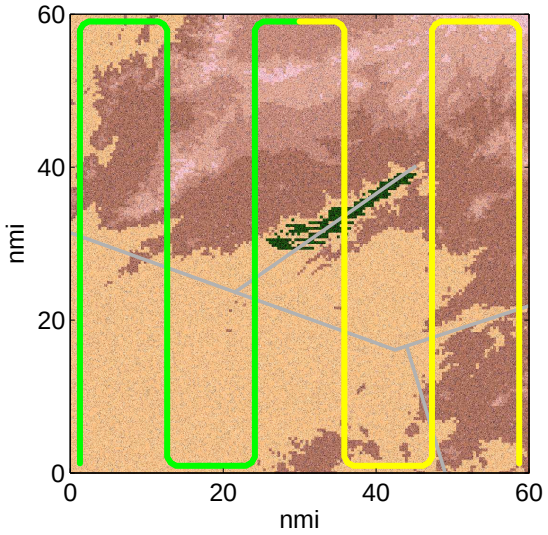

Figure 3.15: A conventional ladder search path without prior

value is relatively small compared to the values of G in Table 3.4. This was probably due to the amount of potential information that would be lost if a UAV were unable to detect new targets.

An interesting point is that the total number of detections is almost the same for Figures 3.13 and 3.11. However, their G_1 values vary significantly. This can be attributed to the UAVs of Figure 3.13 being forced to search the higher altitude regions of the mountain, which resulted in a shorter detection range and fewer repeated observations.

Table 3.4 also shows how a poor solution, as in Figure 3.15, might score. This poor solution detected fewer targets along with only 3.4 observations per target. Since these two metrics are important to G, this solution received a much lower value for G in both G_1 and G_2. Even though the observations/target for the informed ladder pattern was identical to the uninformed ladder pattern, the informed ladder pattern performed much better in G because it was able to detect many more targets.

The differences in the values of G_1 and G_2 bring to light the trade-off between the conflicting objectives of detecting new targets and tracking previously detected targets. These objectives are conflicting because, due to limited resources, a searcher cannot simultaneously follow one target

while searching elsewhere for new targets. Not only do these differences appear in G, but they also show up in the detections and observations/target columns. Though the settings of both G_1 and G_2 consider the objectives of detection and tracking, G_1 emphasizes better tracking while G_2 encourages detecting more targets. In general, it is hard to say how much better one solution is than another without knowing exactly which of the objectives is more desirable but Table 3.4 does provide a starting point in understanding how different solutions are valued.

3.8 Summary

This chapter proposes a novel approach for path planning that maximizes the quantified value of target information for search and track missions.

The cornerstone of the approach is the introduction of a new scalar-valued objective function G that integrates naturally and coherently the objectives of target detection, target tracking, and vehicle survivability: In G, the number of terms reflects the number of targets detected and how large each term is reflects how well each detected target is tracked. As such, our G makes good sense for jointly optimizing the conflicting objectives of detection and tracking. Also, G quantifies the value of information rather than the information itself. This makes G convenient and flexible—its factors can be chosen to fit a user's particular needs and are simple to use. The time factor behaves as a window in time when assigning a value to information. The importance factor is used to assign a relative value to the information of a target. These factors allow G to quantify the value of the information that can be obtained by following a particular path.

The simplicity of G makes it promising for many applications. We have provided several simple examples to highlight the attributes of G. These examples show how the value of G can be affected by its parameters. We also have included a realistic simulation of a search and track mission using G with and without threats. Using G, we have compared and discussed the simulation results for five different path choices for a team of cooperative UAVs. The results show that G can be used to determine which search path is better at detecting and tracking targets based on which

targets are important and when they are important. Whether a mission's focus is track oriented or detection oriented, tradeoffs must be made and G is able to handle well the tradeoffs between the conflicting objectives of search and track missions.

4

Minimum Time-Error Planning Horizon for Plan Updating Triggered by Poisson Random Event

4.1 Introduction

A major task for plan update is the selection of a planning horizon. A planning horizon is how far into the future a plan includes. Here, we attempt to answer this question,

What is the optimal planning horizon?

There is very little research, if any, regarding the selection of planning horizons. This chapter derives a planning horizon that is optimal in the sense of minimizing errors in planning time when the plan updates are triggered by Poisson random events. This is the first step towards finding "the" optimal planning horizon if such a horizon exists. It is an important contribution because it is the only available planning horizon that has been derived. It is suitable for an unknown number of targets with unknown locations.

Despite the widespread use of planning horizons [6–9, 13, 16, 17, 19, 26–28, 35, 41, 45, 47, 51, 52, 54, 55, 59, 60, 64–66, 71], very little research is available on the selection of a planning horizon. In general, they are usually selected either arbitrarily or based on limited computational capabilities. In [7–9] a lower bound is set for a planning horizon based on the minimum number of looks to determine whether or not a target is in a region. Unfortunately, much work leaves the planning horizon for the end-user to decide. This causes horizons to be selected for convenience rather

than optimality. To help in this selection, this chapter derives an *optimal planning horizon* that minimizes errors in planning time, namely, overshoot and undershoot.

Throughout this chapter, an event is defined as anything that triggers a plan to be updated. Often, these events occur in an unknown number and at unknown times. With this in mind, we modeled this random number of events by a Poisson distribution while assuming that the rate parameter is exactly known. Therefore, the corresponding exponential distribution is also known, which defines the distribution of the time between events. Moreover, it is these events that change the dynamics of a scenario and require that a plan be updated. Hence, the benefits of using a Poisson model are twofold in that it describes both the expected number and frequency of events. In this light, the purpose of this chapter is to show the relationship between the planning horizon that minimizes planning time errors and the aforementioned rate parameter.

We proceed in Section 4.2 by supporting the need to update a plan when it is interrupted. In Section 4.3, we define the planning time errors, undershoot and overshoot. Then, in Section 4.4, we derive the planning horizon that minimizes these errors. Section 4.5 describes a predator-prey simulation and Section 4.6 presents its results. Finally, we conclude with Section 4.7.

4.2 Optimality of Plans

When change occurs during the mission, there are two options. The first is to ignore the change and continue with the original plan and the second is to update the plan. This work handles the second case by proposing a planning horizon that minimizes the time-error when a plan is changed. When a plan uses the optimal planning horizon, it avoids two pitfalls. The plan can be either too short or too long. If the plan is too short, it does not consider the big picture. Even though a myopic plan is less likely to be interrupted, its associated computational savings come at the expense of not being big-picture optimal. Among these short-sighted plans are the so-called greedy solutions. On the other hand, when a plan is too long relative to the rate of events, it is not best suited to achieve the goal because it is very likely to be interrupted. Then, once a plan is interrupted, there

is no guarantee that the completed portion of the plan is still optimal over the abbreviated horizon. This is true even if the plan was optimal over its original horizon. Support for using the optimal planning horizon is provided next.

4.2.1 Longer plans are not always better plans.

Let a plan beginning at time t with a planning horizon length τ be denoted as $P_\tau(t)$. Then, let $P^*_\tau(t)$ be optimal over the interval $[t, t+\tau]$ such that

$$J(P^*_\tau(t)) \geq J(P_\tau(t)) \qquad \forall P_\tau(t)$$

where J is the objective function to be maximized.

Next, suppose that we have two plans $P^*_{\tau_1}(t)$ and $P^*_{\tau_2}(t)$ that are optimal over their respective planning horizons, with

$$\tau_1 < \tau_2.$$

Then, if plan $P^*_{\tau_2}(t)$ is interrupted at time $t+\tau_1$, it cannot be better than plan $P^*_{\tau_1}(t)$ over the interval $[t, t+\tau_1]$ because plan $P^*_{\tau_1}(t)$ is optimal over that interval. Hence, the completed portion of an interrupted plan may not be optimal, which is why plans should not be too long relative to the rate of interrupting events.

4.2.2 The remainder of an interrupted plan is not optimal.

Let τ_2 be the optimal planning horizon among all possible horizons. Using the above reasoning, we explain why an entirely new plan of length τ_2 should be generated following an interruption, rather than using the remainder of the interrupted plan directly. The key point here is that simply appending to the remaining part of the current plan is not optimal even when using the optimal horizon. This is explained next.

Suppose plan $P^*_{\tau_2}(0)$ is optimal over the interval $[0, \tau_2]$ and is interrupted at time τ_1. The

question here is,

> *If a plan is interrupted, should the new plan include the unfinished portion of the original plan?*

The answer is no. $P^*_{\tau_2}(0)$ was optimal over the interval $[0, \tau_2]$ because it used all information available at time $t = 0$. Now that a new plan is needed, the most up-to-date information should be used to generate the new plan.

With $P^*_{\tau_2}(0)$ being interrupted at time τ_1, let $P_{\tau_{12}}(\tau_1)$ denote the unfinished part of the plan over the interval $[\tau_1, \tau_2]$ with

$$\tau_{ij} = \tau_j - \tau_i.$$

This is the part of the plan that is in question.

Instead of using $P_{\tau_{12}}(\tau_1)$ as the beginning part of the next plan, we suggest generating an entirely new plan, $P^*_{\tau_{13}}(\tau_1)$, that is optimal over the interval $[\tau_1, \tau_3]$ with $\tau_3 = \tau_1 + \tau_2$. This is better than a plan that combines $P_{\tau_{12}}(\tau_1)$ with another plan, $P_{\tau_{23}}(\tau_2)$, that is over the remainder of the interval $[\tau_2, \tau_3]$.

Since $P^*_{\tau_{13}}(\tau_1)$ is optimal over the interval $[\tau_1, \tau_3]$,

$$J(P^*_{\tau_{13}}(\tau_1)) \geq J(P_{\tau_{12}}(\tau_1)) + J(P_{\tau_{23}}(\tau_2)) \;\; \forall P_{\tau_{23}}(\tau_2).$$

That is, simply appending to the end of the current plan cannot be better than an optimal update, which justifies generating a new plan.

4.3 Errors in Planning Time

This section describes two types of timing errors: undershoot and overshoot. An undershoot occurs when a plan ends before an event takes place, thus requiring that a new plan be generated. An overshoot occurs when an event interrupts a plan, hence wasting the remainder of the plan. In

other words, a good plan might be end-loaded—a lot of important actions will be undertaken towards the end of the plan after spending time learning. An overshoot means the plan is cut short without these important actions. An undershooting plan is a myopic one.

4.3.1 Overshoot.

Once $P^*_{\tau_2}(0)$ in Section 4.2.2 has been interrupted, the amount of time that is wasted is equal to τ_{12}. This *overshoot* is denoted as δ (Figure 4.1). In cases like this, it is easy to calculate the amount of plan that is wasted as

$$\delta = \tau - t.$$

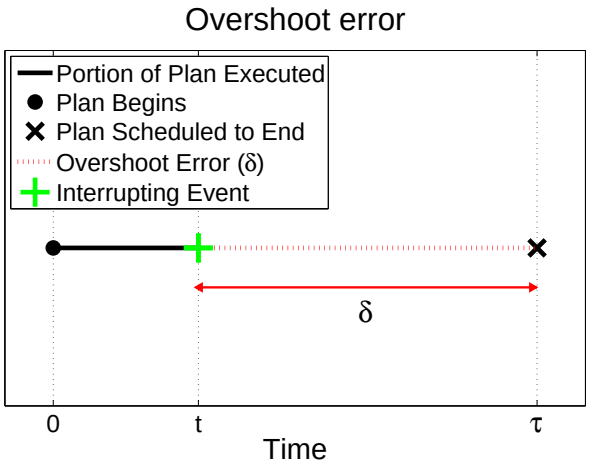

Figure 4.1: An example of overshoot error.

4.3.2 Undershoot.

If an event has yet to occur and the plan has ended, a new plan will have to be made. Then, when the event occurs at some time t in the future, the undershoot can be calculated as

$$\epsilon = t - \tau.$$

as seen in Figure 4.2.

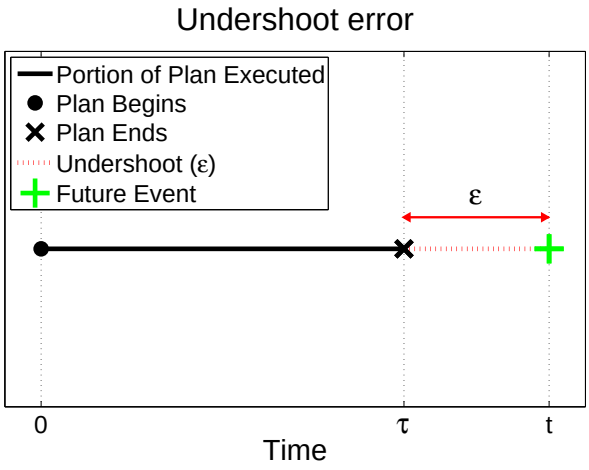

Figure 4.2: An example of undershoot error.

Furthermore, additional penalties for under-planning may be needed if the event occurs outside multiple planning horizons. For example, if an event will occur five time-steps into the future, a planning horizon that only plans for a single step will undershoot the event several times.

4.3.3 Accumulating errors.

Counting both overshoot and undershoot is important. If only the overshoot were considered, a greedy plan using a single-step horizon would never over-plan. If both overshoot and undershoot errors are counted, every greedy plan will be penalized because its plans are too short. On the other hand, if only the undershoot were considered, planning horizons that are too long would never be penalized. As explained in Section 4.2.1, plans that are prematurely terminated can no longer guarantee their optimality over the abbreviated time-horizon. These long plans should be penalized for the amount of plan that is wasted.

4.3.4 *Example of accumulated undershoot.*

Suppose a plan has a horizon of length τ. Now suppose that an event will occur at the time 5τ, which is unknown to the planner. Figure 4.3 shows that after plan #1, the event will occur four planning horizons into the future. The undershoot for the first plan is

$$\epsilon[P_\tau(0)] = 4\tau,$$

which is the amount of time between the end of plan #1 and the event. After plan #2, the event will occur three planning horizons into the future, which is an undershoot of $\epsilon[P_\tau(\tau)] = 3\tau$. Continuing on to plan #5, we have $\epsilon[P_\tau(2\tau)] = 2\tau$, $\epsilon[P_\tau(3\tau)] = \tau$, and $\epsilon[P_\tau(4\tau)] = 0$. These errors are shown in Figure 4.3. Summing these errors yields the total undershoot error:

$$\sum_{i=0}^{4} \epsilon[P_\tau(i\tau)] = 10\tau.$$

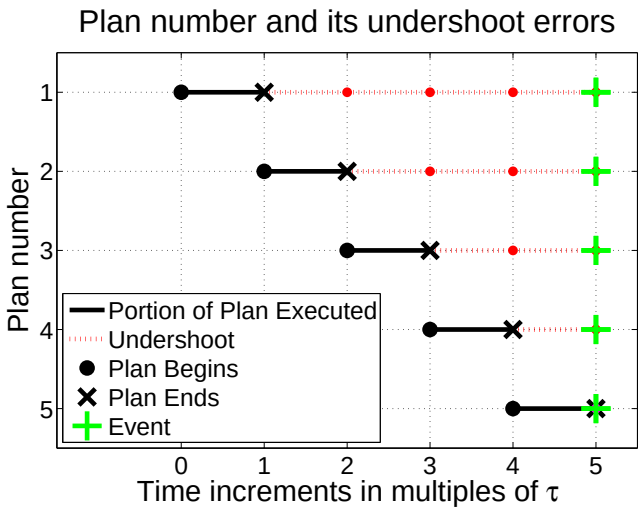

Figure 4.3: An example of accumulated undershoot error for a plan of length τ when the interrupting event occurs at a time five planning horizons (5τ) into the future.

4.4 Optimal Planning Horizon

4.4.1 Poisson assumption

Though optimal planning horizons for other distributions can be derived in a similar fashion, we use the Poisson distribution to derive the optimal planning horizon. The benefits of using a Poisson distribution are as follows:

i. Interrupting events are relatively rare, and the Poisson distribution is a widely used good model for such events.

ii. A Poisson distribution expresses the probability of k events occurring over an interval. Its probability mass function is given by

$$p(k, \lambda) = e^{-\lambda} \frac{\lambda^k}{k!} \quad k = 0, 1, 2, \ldots$$

Using this distribution, the unknown number of events over an interval can be estimated.

iii. If the rates of different types of interruptions are independent and all Poisson, then the total rate of interruptions is still Poisson and its rate parameter is equal to the sum of the rate parameters.

iv. If the rate parameter for a Poisson random variable is λ, then the time between events is exponentially distributed with the density

$$f(t, \lambda) = \begin{cases} \lambda e^{\lambda t}, & t > 0 \\ 0, & t < 0 \end{cases}$$

See Figure 4.4.

v. For a search and track mission, if the number of targets is Poisson, so are the number of

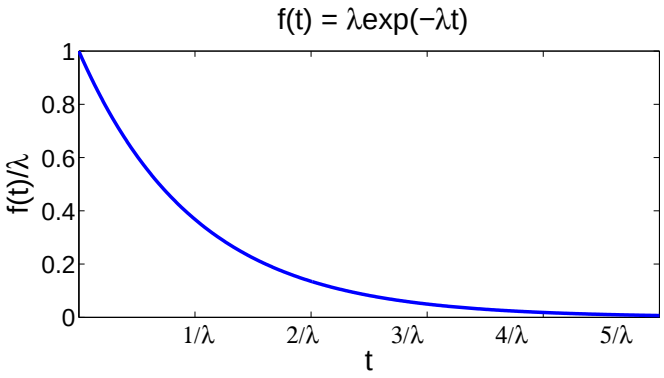

Figure 4.4: Normalized exponential function vs. units of λ^{-1}. This figure holds for any λ.

detected targets. Because of property iii, this is true even with several types of targets and each has its own rate parameter or probability of detection.

4.4.2 Derivation

We now derive the optimal planning horizon, τ, that minimizes the expected error (undershoot and overshoot errors) in planning time. Suppose the number of interrupting events follows a Poisson distribution with a rate parameter of λ. Then the time between events follows an exponential distribution with a mean time of $1/\lambda$ as shown in Figure 4.4.

First, partition the time axis of Figure 4.4 in multiples of τ, as in Figure 4.5. Here, τ is arbitrary and is used as boundaries when calculating the probability of an event occurring. Each of these sections will be referred to by its index number. These indices refer to the number of updates used to arrive in this section. For example, the first plan will have an index of n_0 because it is not an update while the index of the first update is n_1 even though it is the second section, as shown in Figure 4.5.

Using the boundaries in Figure 4.5, we can obtain the probability of an event occurring in the n^{th} section relative to the current planning time. See equation (4.1). The event time is denoted by

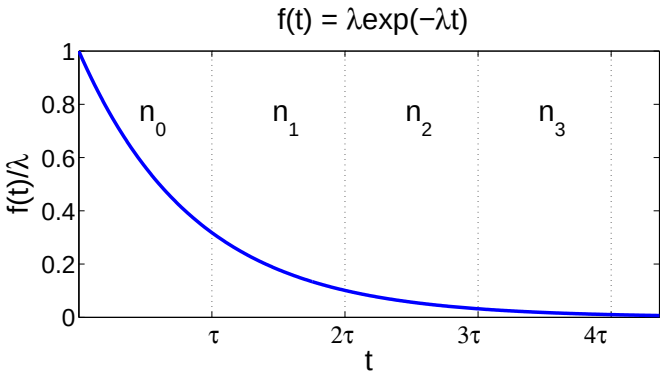

Figure 4.5: Partitioned exponential distribution of Figure 4.4. The section for each index n_i has a width of τ.

t and the current planning time is always $t = 0$.

$$
\begin{aligned}
P\{n\tau < t < (n+1)\tau\} \qquad & \text{for } n = 0, 1, 2, \ldots \\
&= P\{t < (n+1)\tau\} - P\{t \leq n\tau\} \\
&= [1 - \exp(-(n+1)\lambda\tau)] - [1 - \exp(-n\lambda\tau)] \\
&= \exp(-n\lambda\tau) - \exp(-(n+1)\lambda\tau) \\
&= \exp(-n\lambda\tau)(1 - \exp(-\lambda\tau)) \qquad (4.1)
\end{aligned}
$$

We can also calculate the average amount of overshoot, δ_n, in the n^{th} section. See Figure 4.6 in the derivation of δ_n in equation (4.2).

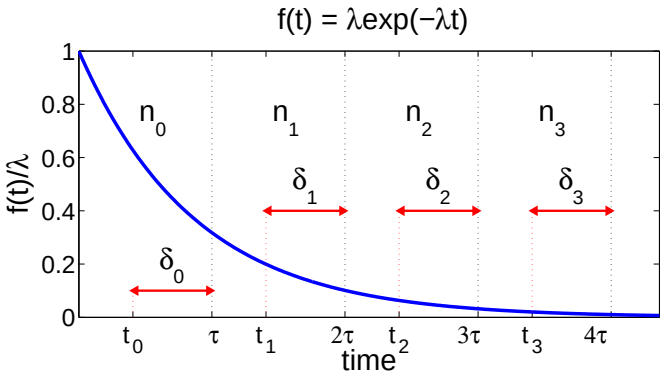

Figure 4.6: Shows δ_n for each of the sections shown in Figure 4.5. Here, t_i is the mean event time in section n_i. equation (4.2) shows that $\delta_0 = \delta_1 = \delta_2 = \ldots = \delta_n$

$$
\begin{aligned}
\delta_n &= E[(n+1)\tau - t | n\tau < t < (n+1)\tau] \\
&= (n+1)\tau - E[t | n\tau < t < (n+1)\tau] \\
&= (n+1)\tau - \int_0^{\infty} t f(t | n\tau < t < (n+1)\tau) dt \\
&= (n+1)\tau - \int_{n\tau}^{(n+1)\tau} t \frac{f(t)}{P(n\tau < t < (n+1)\tau)} dt \\
&= (n+1)\tau - \frac{\lambda \int_{n\tau}^{(n+1)\tau} t \cdot \exp(-\lambda t) dt}{P(n\tau < t < (n+1)\tau)} \\
&= (n+1)\tau - \frac{\lambda \int_{n\tau}^{(n+1)\tau} t \cdot \exp(-\lambda t) dt}{\exp(-n\lambda\tau) - \exp(-(n+1)\lambda\tau)} \\
&= \frac{\{\tau - \frac{1}{\lambda}[1 - \exp(-\lambda\tau)]\} \exp(-n\lambda\tau)}{\exp(-n\lambda\tau) - \exp(-(n+1)\lambda\tau)} \\
&= \frac{\tau - \frac{1}{\lambda}[1 - \exp(-\lambda\tau)]}{1 - \exp(-\lambda\tau)} \\
&= \frac{\tau}{1 - \exp(-\lambda\tau)} - \frac{1}{\lambda} \\
&= \delta(\tau, \lambda)
\end{aligned}
\tag{4.2}
$$

Note that δ_n is independent of n, meaning that δ_n is a constant given τ and λ. This makes sense in view of the memoryless property of the exponential distribution.

The average undershoot for section n is denoted as ϵ_n. Note that the corresponding section

index is relative to the beginning of this particular plan rather than the very first plan. As shown in Figure 4.7,

$$\epsilon_1 = \tau - \delta(\tau, \lambda)$$

$$\epsilon_2 = \tau + \epsilon_1$$

$$\epsilon_2 = 2\tau - \delta(\tau, \lambda)$$

Then, by induction, we have

$$\epsilon_3 = \tau + \epsilon_2$$

$$\epsilon_3 = 3\tau - \delta(\tau, \lambda)$$

$$\vdots$$

$$\epsilon_n = \tau + \epsilon_{n-1}$$

$$\epsilon_n = n\tau - \delta(\tau, \lambda)$$

By default,

$$\epsilon_0 = 0$$

because an event that interrupts the current plan can only cause an overshoot error. That is why Plan # 3 in Figure 4.7 does not have an undershoot ϵ_0.

Let $\Delta_\tau(t)$ denote the total planning-time error for a sequence of plans of length τ beginning at time t. This total error is the sum of the overshoot and all the undershoots. Using Figure 4.7 for example, the accumulated error for the sequence of plans is

$$\Delta_\tau(0) = \epsilon_2 + \epsilon_1 + \delta. \tag{4.3}$$

Then, let $\overline{\Delta}_\tau$ denote average accumulated error for a sequence and $\overline{\Delta}_{\tau,n}$ denote the expected error

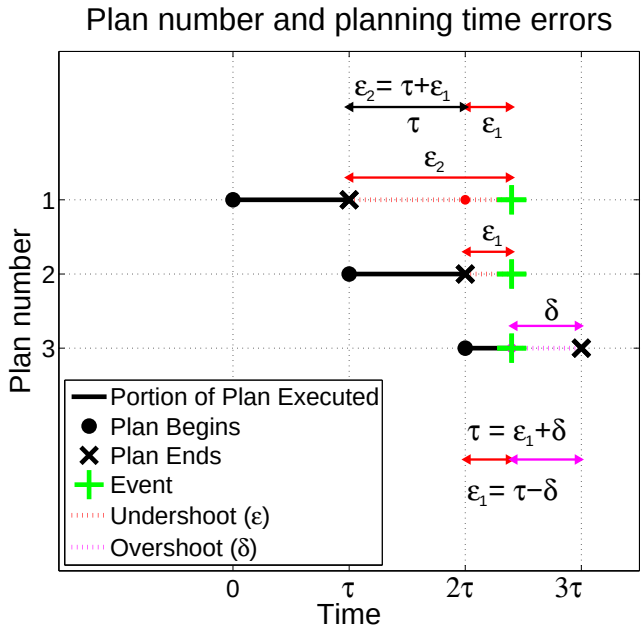

Figure 4.7: This example shows that the planning-time error for the first and second plans is equal to ϵ_2 and ϵ_1, respectively. The third plan has a planning-time error equal to δ.

given that the event occurs in section n. This is defined as

$$\overline{\Delta}_{\tau,n} = E\left[\Delta_\tau | n\tau < t < (n+1)\tau\right]$$

resulting in

$$
\begin{aligned}
\overline{\Delta}_0 =& \delta(\tau,\lambda) \\
\overline{\Delta}_1 =& \delta(\tau,\lambda) + \epsilon_1 \\
\overline{\Delta}_2 =& \delta(\tau,\lambda) + \epsilon_1 + \epsilon_2 \\
\overline{\Delta}_3 =& \delta(\tau,\lambda) + \epsilon_1 + \epsilon_2 + \epsilon_3 \\
& \vdots \\
\overline{\Delta}_n =& \delta(\tau,\lambda) + \epsilon_1 + \epsilon_2 + \ldots + \epsilon_{n-1} + \epsilon_n \\
=& \delta(\tau,\lambda) + (\tau - \delta(\tau,\lambda)) + (2\tau - \delta(\tau,\lambda)) + \ldots \\
& + [(n-1)\,\tau - \delta(\tau,\lambda)] + (n\tau - \delta(\tau,\lambda)) \\
=& \delta(\tau,\lambda) + \tau \sum_{i=1}^{n} i - n\delta(\tau,\lambda) \\
=& \tau \sum_{i=1}^{n} i + (1-n)\delta(\tau,\lambda)
\end{aligned}
$$

We now have (a) the probability of an event occurring in the n^{th} section, $P\{n\tau < t < (n+1)\tau\}$, and (b) the average error associated with the n^{th} section. Then using the total probability theorem,

we have the *expected total planning-time error*:

$$
\begin{aligned}
\overline{\Delta} =& \sum_{n=0}^{\infty} E\left[\Delta | n\tau < t < (n+1)\tau\right] P\left\{n\tau < t < (n+1)\tau\right\} \\
=& \sum_{n=0}^{\infty} \overline{\Delta_n} P\left\{n\tau < t < (n+1)\tau\right\} \\
=& \sum_{n=0}^{\infty} \left\{ \left[\tau \sum_{k=1}^{n} k + (1-n)\delta_n \right] P\left\{n\tau < t < (n+1)\tau\right\} \right\} \\
=& \sum_{n=0}^{\infty} \tau \frac{n(n+1)}{2} \exp(-n\lambda\tau)(1-\exp(-\lambda\tau)) \\
& + \sum_{n=0}^{\infty} (1-n)\{\tau - \frac{1}{\lambda}[1-\exp(-\lambda\tau)]\} \exp(-n\lambda\tau) \\
=& \frac{\tau(1-\exp(-\lambda\tau))}{2} \sum_{n=0}^{\infty} n(n+1)\exp(-n\lambda\tau) \\
& + \{\tau - \frac{1}{\lambda}[1-\exp(-\lambda\tau)]\} \sum_{n=0}^{\infty} (1-n)\exp(-n\lambda\tau) \\
=& \frac{\tau(1-\exp(-\lambda\tau))}{2} \frac{2\exp(-\lambda\tau)}{(1-\exp(-\lambda\tau))^3} \\
& + \{\tau - \frac{1}{\lambda}[1-\exp(-\lambda\tau)]\} \frac{1-2\exp(-\lambda\tau)}{(1-\exp(-\lambda\tau))^2} \\
=& \frac{\tau\exp(-\lambda\tau)}{(1-\exp(-\lambda\tau))^2} + \frac{\{\tau - \frac{1}{\lambda}[1-\exp(-\lambda\tau)\}}{(1-\exp(-\lambda\tau))^2} \\
& - \frac{2\tau\exp(-\lambda\tau) - \frac{2}{\lambda}[\exp(-\lambda\tau) - \exp(-2\lambda\tau)]}{(1-\exp(-\lambda\tau))^2} \\
=& \frac{\tau\exp(-\lambda\tau) + \tau - \frac{1}{\lambda} + \frac{1}{\lambda}\exp(-\lambda\tau)}{(1-\exp(-\lambda\tau))^2} \\
& + \frac{-2\tau\exp(-\lambda\tau) + \frac{2}{\lambda}\exp(-\lambda\tau) - \frac{2}{\lambda}\exp(-2\lambda\tau)}{(1-\exp(-\lambda\tau))^2} \\
=& \frac{\tau - \frac{1}{\lambda} - (\tau - \frac{3}{\lambda})\exp(-\lambda\tau) - \frac{2}{\lambda}\exp(-2\lambda\tau)}{(1-\exp(-\lambda\tau))^2}
\end{aligned}
\tag{4.4}
$$

Its derivative is equal to

$$
\overline{\Delta}' = \frac{1 - (3+\lambda\tau)\exp(-\lambda\tau) + (2+\lambda\tau)\exp(-2\lambda\tau)}{(1-\exp(-\lambda\tau))^3} \tag{4.5}
$$

It can be shown that $\overline{\Delta}$ is minimized when $\overline{\Delta}'$ is equal to zero or when its numerator is equal to zero. As seen in Figure 4.8, the numerator of $\overline{\Delta}'$ is equal to zero when

$$\tau = \rho \frac{1}{\lambda} \tag{4.6}$$

with $\rho \approx 1.1462$. This is the relationship between the planning horizon that minimizes planning time errors and the rate parameter. It means that the optimal planning horizon is about 1.15 times the mean time between update-triggering events.

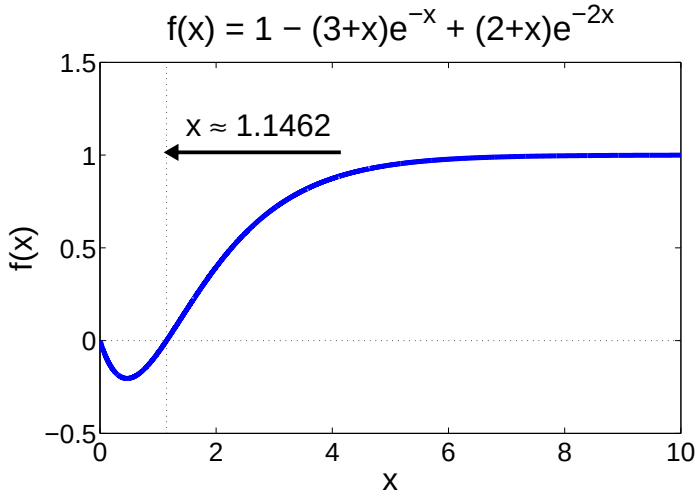

Figure 4.8: Letting $x = \lambda\tau$ in $\overline{\Delta}'$, this figure shows that the numerator of $\overline{\Delta}'$ is equal to zero when $x \approx 1.1462$.

4.5 Simulation

To test the proposed planning horizon, a predator-prey scenario is simulated to compare different planning horizons. Of the planning horizons compared, the best will be the one that minimizes the average time for the predator to capture the prey.

Let the speeds of the predator and prey be fixed at 2.5m/s and 2.0m/s, respectively. To evade the faster predator, the prey will randomly change its heading uniformly over 0–360deg with the

time between maneuvers following an exponential distribution with mean of $\bar{\tau} = 10$, meaning that $\lambda = 0.1$ from $\bar{\tau} = 1/\lambda$. When the prey changes its heading the predator must update its plan to account for the prey's maneuver. Hence, the prey's maneuver is an interrupting event as viewed by the predator.

For simplicity, at the instant the prey changes direction, the predator will generate a new plan that again tries to minimize the distance between them at the end of the new plan. When the predator's current plan ends, it will generate a new plan until it finally captures the prey. It is important to note that the predator's objective function is to minimize the distance between itself that the prey at the end of its plan and it does not consider beyond its own planning horizon.

The horizon factor ρ from equation (4.6) will be evaluated over the range 0.25 to 5 in increments of 0.25. We also evaluate the proposed horizon of $\rho = 1.15$. Each of these will be evaluated over 10,000 runs to provide a smooth curve. In addition, we repeat the evaluation using different initial distances, d_0. The initial distances used are 50m, 100m, 250m, 500m, 1000m, and 2000m.

4.6 Simulation Results

Figures 4.9–4.14 show the average time needed for the predator to capture the prey. For each case, the horizon yielding the minimum time is designated with a yellow triangle. Also, the location of the proposed planning horizon is indicated with a red vertical dashed line. For each case, the curves appear to have a minimum between the values of one and two. Using the minimum values from Figures 4.9–4.14, Figure 4.15 shows that the best horizon lies somewhere between $\rho = 1$ and $\rho = 1.75$ for the evaluated ranges and our proposed $\rho = 1.15$ is a good choice.

4.7 Conclusions

This chapter formulated the problem of an optimal planning horizon and derived a planning horizon that is optimal in the sense of minimizing errors in planning time when the plan updates

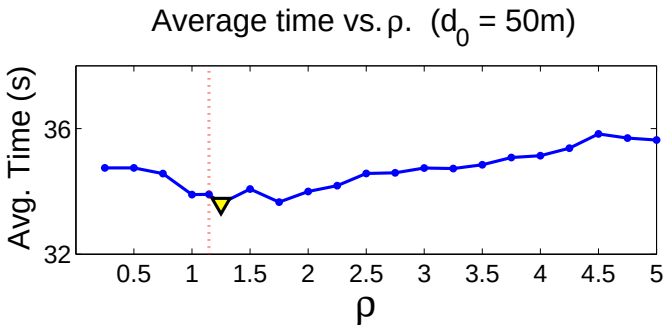

Figure 4.9: Average time to capture the prey with an initial separation of 50m. The minimum point occurs when $\rho = 1.25$.

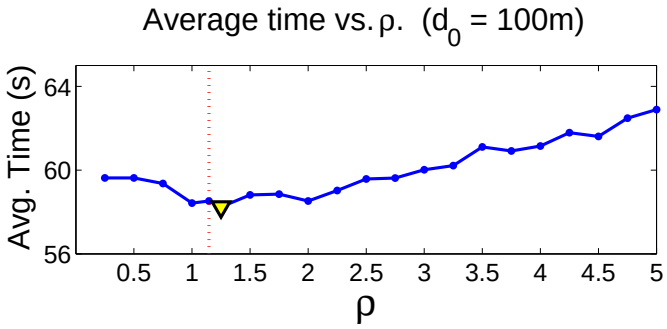

Figure 4.10: Average time to capture the prey with an initial separation of 100m. The minimum point occurs when $\rho = 1.25$.

are triggered by Poisson random events. It is a first step towards finding "the" optimal planning horizon if such a horizon exists. The predator-prey scenario was presented to compare different planning horizons. While the proposed horizon is optimal in minimizing planning time errors, our results show that the proposed horizon is also near optimal in minimizing the average time needed to capture the prey. The results show that there is not a single best horizon for our simulations but the test cases provide some evidence that the length of a plan (in time) should be close to the mean time between interrupting events. One may expect the planning horizon that minimizes the time to capture the prey be equal to the prey's mean maneuver time, but the results show that the length of a plan should lie somewhere between the factors of $\rho = 1$ and $\rho = 1.75$.

Optimal planning horizon depends on the nature and features of the mission and planning, and

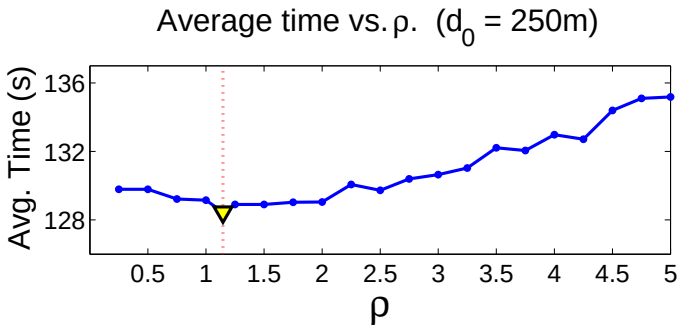

Figure 4.11: Average time to capture the prey with an initial separation of 250m. The minimum point occurs when $\rho = 1.15$.

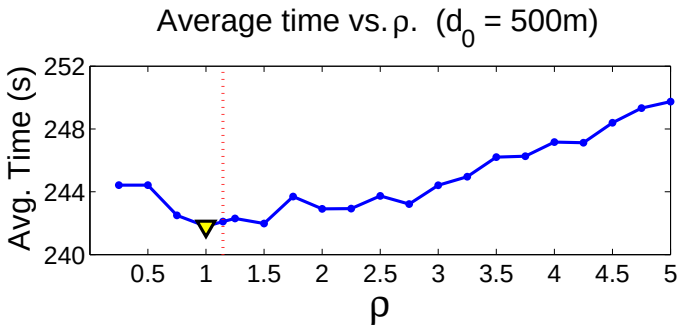

Figure 4.12: Average time to capture the prey with an initial separation of 500m. The minimum point occurs when $\rho = 1$.

it can be very complicated. Our simplifying treatment of minimizing the planning time error in terms of undershoot and overshoot provides a tractable means to handle this complex yet important problem. Since this treatment is not tailored specifically to any particular application, the results presented here are general and can be applied in many other areas. Moreover, we used time as an example but other units can be used to specify the rate parameter of the Poisson distribution.

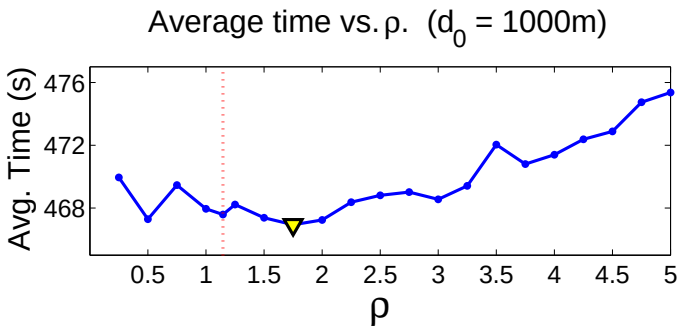

Figure 4.13: Average time to capture the prey with an initial separation of 1000m. The minimum point occurs when $\rho = 1.75$.

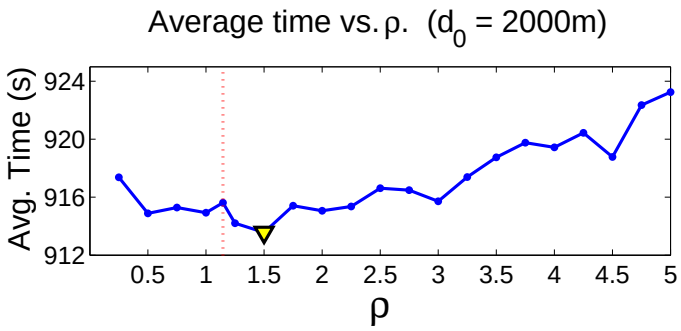

Figure 4.14: Average time to capture the prey with an initial separation of 2000m. The minimum point occurs when $\rho = 1.5$.

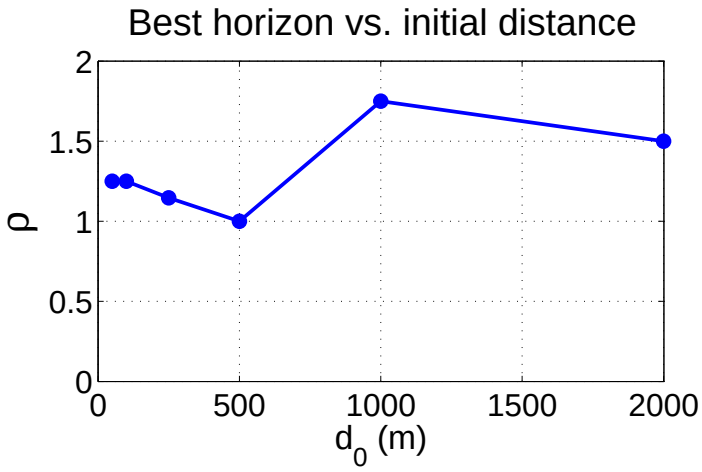

Figure 4.15: Best horizon versus initial distance with a mean value of 1.3160.

5

Modified Particle Swarm Optimization

5.1 Introduction

A common approach to route planning is to apply an optimization algorithm to obtain a good solution based on assumptions about the targets. In general, the path planner uses simulated targets to emulate the expected target behavior in order to evaluate candidate search paths. In practice, a major drawback is that the prior distribution of targets is only used to *evaluate* the search paths rather than to guide the optimization algorithm in the generation of the search paths. This chapter introduces the use of the prior target distribution in the *generation* of search paths rather than only for *evaluation*. Here, we present a modified particle swarm optimization algorithm for search missions that explicitly uses the sampled target distribution in the creation of candidate search paths, which naturally improves the results since the search paths directly depend on the time varying target locations. Results from a realistic cooperative path planning scenario show that the usage of target distributions can improve the performance of particle swarm optimization.

The chapter is organized as follows: Section 5.2 briefly summarizes particle swarm optimization in order to explain how to use the prior target distribution to create search paths. Section 5.3 introduces the modified particle swarm optimization algorithm. Section 5.4 describes an illustrative scenario. Section 5.5 presents results and Section 5.6 contains the conclusions.

5.2 Review of Particle Swarm Optimization

Particle swarm optimization (PSO) is an iterative optimization algorithm that uses a large number of candidate solutions, called particles, to sample the solution space. For the problem described here, the solution space is the set of all feasible search paths. Like other forms of evolutionary computation, PSO is useful when evaluating the entire set of feasible solutions is computationally intractable. It explores the solution space by evaluating candidate search paths and then creating new paths around the best paths evaluated so far. Particles are evaluated using an objective function and the cumulative probability of detection is used in this chapter for simplicity.

In search for better solutions, particles traverse the solution space using an equation similar to the kinematics equation for motion, where motion can be decomposed into position and velocity. A particle has a position in the solution space and that position represents a unique search path. A particle also has a velocity in the solution space. The velocity describes how fast the solution is moving in the solution space and does not reflect the actual velocity of the searchers in the real world. Rather, it reflects how fast the solution is changed in the real world. As a particle moves through the solution space, the real-world search path changes shape resulting in a new search path. The equations for particle swarm optimization are

$$p_{k+1}^{i} = p_{k}^{i} + v_{k}^{i} \tag{5.1}$$

and

$$v_{k}^{i} = w_{I}^{i} v_{I}^{i} + w_{GB}^{i} v_{GB}^{i} + w_{LB}^{i} v_{LB}^{i} \tag{5.2}$$

where

- i is the index of the particle.
- p_{k}^{i} is the position of particle i in the solution space at iteration k.
- v_{k}^{i} is the velocity vector of the particle in the solution space.

- w_I^i is the weight assigned to give inertia to this particle.
- w_{GB}^i is the weight that moves a search path towards the very best path discovered so far. It is the Global Best (GB) among the current set of solutions.
- v_{GB}^i is the point-wise distance vector between this path and the very best path discovered by the swarm so far.
- w_{LB}^i is the weight that moves a search path towards the best path discovered by this particle so far. It is the local best (LB) solution in the history of particle i.
- v_{LB}^i is the point-wise distance vector between this path and the best path discovered by this particle so far.

A search path has N waypoints and each waypoint has a coordinate in the real world. If a waypoint in the real world can be represented using two coordinates, such as latitude and longitude, then

$$p_k^i \in \Re^{N \times 2}$$

Basically, p_k^i contains a sequence of coordinates that describes the search path as a function of time. The vectors v_{GB}^i and v_{LB}^i have the same dimension as p_k^i and are calculated by

$$v_{GB}^i = p_{GB,k}^i - p_k^i \tag{5.3}$$

$$v_{LB}^i = p_{LB,k}^i - p_k^i \tag{5.4}$$

where

- $p_{GB,k}^i$ is the GB particle describing the best solution among all solutions at iteration k.
- $p_{LB,k}^i$ is the best path so far for particle i at iteration k.

The weights determine how each of the components are used to update the particle's velocity. The choice of weights affects the placement of the search paths for the next iteration. When the weights favor the best path so far, the other search paths will search closer to that solution. For

example, if w_I and w_{LB} are set to zero and w_{BSF} is set to one, the next search path will be identical to the solution that is best so far.

After a path is updated, it will be evaluated and then its score will be recorded. Any metric can be used to evaluate search paths but we use cumulative detection probability, which is the number of detected targets divided by the total number of targets. A search path will be declared the best so far if it detects more targets than any other search path is able to detect. This process is iterated for the desired number of iterations or some other terminating criteria. More can be read about particle swarm optimization in [14, 29, 33].

5.3 Using Prior Target Distributions

Particle swarm optimization provides a framework that allows search paths to be iterated in such a way that new search paths focus their energy near other search paths that have already detected targets. A subtle drawback to this is that the target-containing locations become known only after other search paths have found targets in these locations. Recall that these targets were generated using a prior target distribution. If the target distribution is known when generating the simulated targets, then why not use the prior distribution to create the search paths as well? This question introduces the modified particle velocity equation for PSO, which is

$$v_k^i = w_I^i v_I^i + w_{GB}^i v_{GB}^i + w_{LB}^i v_{LB}^i + \sum_{t=1}^{T} w_t^i v_t^i \tag{5.5}$$

where the new terms added in formula (5.5) are

- T is the total number of simulated targets.
- w_t^i is the weight that describes the attraction of the searcher to target t.
- v_t^i is a vector representing the distance in the solution space from this particle to target t.

The new term, v_t, attracts searchers to locations that contain targets. Furthermore, searchers will be more attracted to locations with a high density of targets than to locations that contain few targets.

Without this additional term, new search paths will only be attracted to targets that have already been detected. When this new term is used, the searchers are explicitly attracted to the targets as well as other candidate search paths.

As stated in the previous section, the weights determine how each component affects the motion of a particle. In formula (5.5), the added component encourages the placement of search paths near locations with high densities of targets, which is unlike the traditional method that waits for the best paths to find the targets before other paths can begin searching near those locations. It is important to note that the simulated targets used in the generation of new search paths should not be the same set of targets used to evaluate the search paths. Otherwise, the solution will be over-fitted to that specific realization of targets. In other words, the target set used to generate the search path should not be used again in evaluation.

5.4 Illustrative Scenario and Results

Suppose a hiker has not returned after several days of hiking in a park. The park, as shown in Figure 5.1, is a campground that is 60 miles by 60 miles. It is believed that there is an 85% chance that the hiker is on a trail, an 11% chance that the hiker is lost in the forest, and a 4% chance that the hiker is waiting at another campsite.

The search party consists of two helicopters that fly at a constant speed of 50mph for two hours. They are equipped with sensors that can easily detect the hiker within a two mile radius, i.e. probability of detection is unity when the hiker is within 2 miles of the helicopter and zero elsewhere.

The purpose of this scenario is to demonstrate that using the prior target distribution can improve the performance of PSO in terms of the cumulative probability of detection. The PSO algorithms will use 200 particles and will be terminated after 25 iterations. For a fair comparison, the PSO algorithm without prior was initialized with the identical initial conditions as the PSO that uses prior. This means that the algorithms begin with an identical set of 200 initial search paths

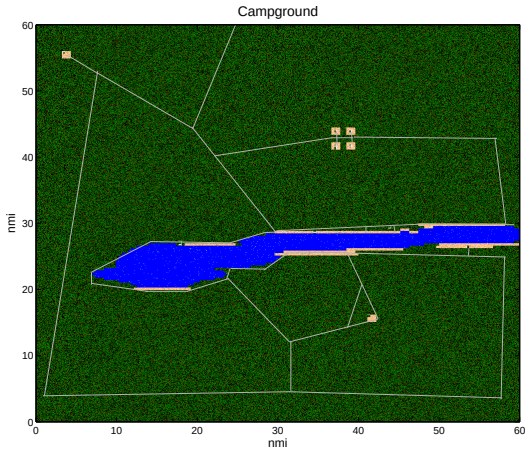

Figure 5.1: A fictional campground with hiking trails (grey), forest (green), campsites (tan), and water (blue).

and were evaluated using the same realization of the hiker's spatial distribution. Additionally, the set of sampled targets used to evaluate the search paths was not the same as the set used to update the search paths of the modified PSO in formula (5.5).

Figure 5.2 shows a solution generated using the modified PSO algorithm. This solution detects 67.0% of the simulated hikers during the two-hour search mission. Figure 5.3 shows a solution generated without using prior that was only able to provide a 58.5% chance of detecting the lost hiker. Clearly, a better solution would be to cover different sections of the trails rather than revisiting the same part of the trail within a short amount of time.

5.5 Results

Simulation results show that using the prior target distribution in the PSO algorithm can produce search paths that have a better chance of locating a lost hiker than the traditional PSO algorithm. The modified PSO detected 67.0% of the simulated hikers compared to the 58.5% chance of detecting the hiker using the unmodified PSO. For comparison, the hand-made solution shown in Figure 5.4 was able to detect 84.7% of the targets. This solution simply followed the trail.

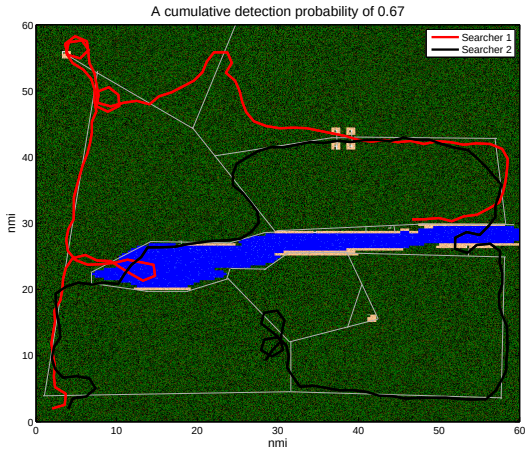

Figure 5.2: A search path generated using the PSO algorithm with prior that has a 67.0% chance of finding the lost hiker.

The PSO solutions did not perform as well as the hand-made solution and there is room for improvement. The PSO solutions could be improved to match the performance of the hand-made solution if the PSO was initialized with more particles and allowed more than 25 iterations. Another possibility for improvement could come by modifying the PSO to make the trail attractive to the search paths could possibly improve the solutions.

Figure 5.5 is a comparison of the percentage of hikers detected as a function of iteration number. It shows that the use of prior improves the performance of PSO. In this example, the unmodified PSO detected 58.5% of the hikers after 25 iterations, whereas the modified PSO that was able to detect 59% after only 3 iterations.

5.6 Conclusion

This work is an initial step towards using prior distributions to solve general problems given prior information. There are several considerations that must be addressed in future work. The first topic is the proper selection of the weights in order to improve convergence to a near optimal solution because the PSO solutions presented here are not as good as the hand-made solution.

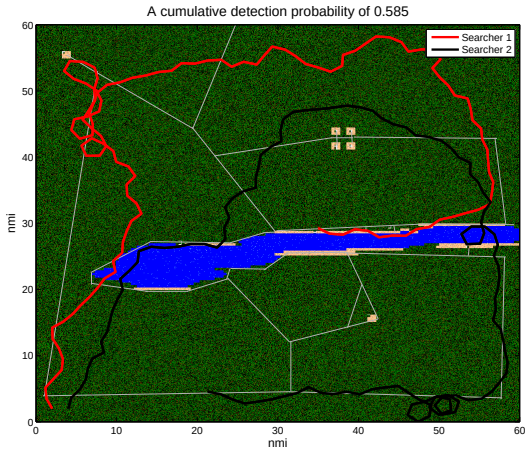

Figure 5.3: A search path generated using the PSO algorithm without prior that has a 58.5% chance of finding the lost hiker.

Other potential improvements include reducing the attraction to undetectable targets or modifying the weights to improve the safety of the searcher. As in all types of mission planning, the proposed method requires that the simulated targets accurately represent the true target distribution in order to create an effective solution. Additionally, if too few simulated targets are used, the solution will be over-fitted to that particular realization of the simulated targets. Future work may also include methods that encourage the searcher to cover more area rather than revisiting searched locations.

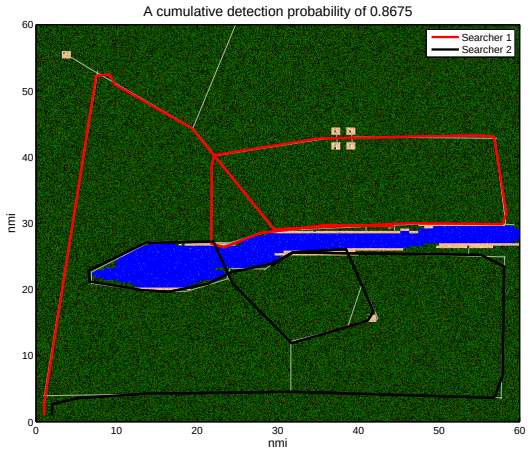

Figure 5.4: A hand-made search path that closely follows the hiking trails. This solution has an 84.7% chance of finding the lost hiker.

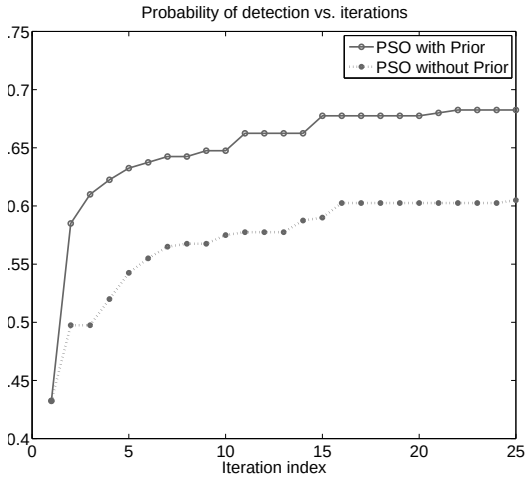

Figure 5.5: PSO with prior was able to detect more targets after 3 iterations than the PSO without prior was able to detect after 25 iterations.

6

Summary and Future Work

6.1 Summary

This dissertation has three contributions in the area of path planning for UAV SAT missions. These contributions are: (a) the study of a novel metric, *G*, used to quantify the value of the target information gained during a search and track mission, (b) an optimal planning horizon that minimizes time-error of a planning horizon when interrupted by Poisson random events, and (c) a modified particle swarm optimization algorithm for search missions that uses the prior target distribution in the generation of paths rather than just in the evaluation of them.

We proposed and studied the information-based metric G and its application to search and track missions using several examples and a realistic simulation of a search and track mission. It has been shown how our performance index can handle the tradeoffs between the conflicting objectives of a search and track mission. Since a searcher cannot simultaneously follow a target (to track it) and locate/track other targets, it is important to be able to balance these tradeoffs when generating a path plan. It also has been explained how G can be tuned to prefer a particular type of solution by adjusting the importance and time factors.

We also derived and studied an optimal planning horizon that is optimal in the sense of minimizing errors in planning time when the plan updates are triggered by Poisson random events. Our simplifying treatment of minimizing the planning time error in terms of undershoot and overshoot provides a tractable means to handle this complex yet important problem. To our knowledge, it

is the only theoretically derived planning horizon available making it an important contribution. Simulation results show that the proposed planning horizon is near-optimal in minimizing the time needed for the predator to capture its maneuvering prey.

A modified particle swarm optimization algorithm for search missions has also been proposed. This method explicitly uses the sampled target distribution to create search paths, which naturally improves the results since the search paths directly depend on the time varying target locations. This is a valuable contribution for search and track missions because in practice, the prior distribution of targets is only used in the *evaluation* of candidate search paths rather than to guide the optimization algorithm in the *creation* of search paths. Results from a realistic cooperative path planning scenario show that using the target distribution in this manner can improve the performance of particle swarm optimization. To our knowledge, our modified PSO is the only one to incorporate the target distribution into the algorithm itself.

The three contributions of this dissertation cover three key components in path planning, namely 1) an objective function, 2) a planning horizon, and 3) an optimization algorithm for path generation. When used together, they can use the target distribution to generate path plans that search for and track targets while maximizing the value of the expected information gain.

6.2 Future Work

Future work includes studying the performance of the proposed planning horizon in SAT missions rather than the predator-prey simulation provided here. Another useful study would be a comparison between the fixed and receding planning horizons. We should study how well our planning horizon compares to other planning horizons in maximizing G over an entire SAT mission. This type of simulation would include generating only a few targets from a Poisson distribution and then allowing the plan to be updated when a target is detected.

A priority in continuing this work is speeding up the PSO algorithm by streamlining the software and using parallel processors. As is, the software is too slow to be implemented in a real-time

system. Speeding up the software can also benefit simulations by allowing more particles to be used without excessive computational overhead. Once the software is sped up, it would be interesting to repeat the simulation in Chapter 5 with more particles and more iterations. Finally, the velocity weights of the modified PSO should be studied in terms of improving the convergence of the algorithm.

Bibliography

[1] Y. Bao, X. Fu, and X. Gao. Path planning for reconnaissance UAV based on particle swarm optimization. In *2010 Second International Conference on Computational Intelligence and Natural Computing*, pages 28–32, November 2010.

[2] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with Applications to Tracking and Navigation*. Wiley, New York, 2001.

[3] R. Beard and T. W. McLain. Multiple UAV cooperative search under collision avoidance and limited range communication constraints. In *Proc. of the 42nd IEEE Conference on Decision and Control*, pages 25–30, Maui, Hi, December 2003.

[4] R. W. Beard, T. W. McLain, M. A. Goodrich, and E. P. Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *IEEE Transactions on Robotics and Automation*, pages 911–922, 2002.

[5] J. Bellingham, A. Richards, and J. P. How. Receding horizon control of autonomous aerial vehicles. In *Proc. of the American Control Conference*, pages 3741–3746, Anchorage, AK, May 2002.

[6] J. Bellingham, M. Tillerson, M. Alighanbari, and J. How. Cooperative path planning for multiple UAVs in dynamic and uncertain environments. In *Proc. of the 41st IEEE Conference on Decision and Control*, pages 2618–2622, Las Vegas, NV, December 2002.

[7] L. F. Bertuccelli and J. P. How. Robust UAV search for environments with imprecise probability maps. In *Proc. of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, pages 5680–5685, Seville, Spain, December 2005.

[8] L. F. Bertuccelli and J. P. How. UAV search for dynamic targets with uncertain motion models. In *Proc. of the 45th IEEE Conference on Decision and Control*, pages 5941–5946, San Diego, CA, December 2006.

[9] L. F. Bertuccelli and J. P. How. Search for dynamic targets with uncertain probability maps. In *Proc. of the 2006 American Control Conference*, pages 737–742, Minneapolis, MN, June 2006.

[10] E. Besada-Portas, L. de la Torre, J. M. de la Crus, and B. de Andres-Toro. Evolutionary trajectory planner for multiple UAVs in realistic scenarios. *IEEE Transactions on Robotics*, pages 619–634, 2010.

[11] D. W. Casbeer, S. Li, R. W. Beard, R. K. Mehra, and T. W. McLain. Forest fire monitoring with multiple small UAVs. In *2005 American Control Conference*, pages 3530–3535, Portland, OR, June 2005.

[12] P. R. Chandler, M. Pachter, D. Swaroop, J. M. Fowler, J. K. Howlett, S. Rasmussen, C. Schumacher, and K. Nygard. Complexity UAV cooperative control. In *Proc. of the American Control Conference*, pages 1831–1836, Anchorage, AK, May 2002.

[13] A. Dogan. Probabilistic approach in path planning for UAVs. In *Proc. of the 2003 IEEE International Symposium on Intelligent Control*, pages 608–613, Houston, TX, October 2003.

[14] R. Eberhart and Y. Shi. Particle swarm optimization: developments, applications and resources. *Proc. of the 2001 Congress on Evolutionary Computation*, 1:81–86, 2001.

[15] J J. Enright, K. Savla, E. Frazzoli, and F. Bullo. Stochastic and dynamic routing problems for multiple uninhabited aerial vehicles. *Journal of Guidance, Control, and Dynamics*, 32(4):1152–1166, 2009.

[16] M. Flint, E. Fernandez-Gaucherand, and M. Polycarpou. Cooperative control for UAVs searching risky environments for targets. In *Proc. of the 42nd IEEE Conference on Decision and Control*, pages 3567–3572, Maui, HI, December 2003.

[17] M. Flint, M. Polycarpou, and E. Fernandez-Gaucherand. Cooperative control for multiple autonomous UAVs searching for targets. In *Proc. of the 41st IEEE Conference on Decision and Control*, pages 2823–2828, Las Vegas, NV, December 2002.

[18] Y. Fu and M Ding. Phase angle-encoded and quantum-behaved particle swarm optimization applied to three-dimensional route planning for UAV. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, PP(99):1–16, 2011.

[19] T. Furukawa, F. Bourgault, B. Lavis, and H. Durrant-Whyte. Recursive bayesian search-and-tracking using coordinated UAVs for lost targets. In *Proc. of the 2006 IEEE International Conference on Robotics and Automation*, pages 2521–2526, Orlando, FL, May 2006.

[20] D. Grundel. Searching for a moving target: Optimal path planning. *Networking, Sensing and Control, 2005*, pages 867–872, July 2005.

[21] D-W Gu, I. Postlethwaite, and Y. Kim. A comprehensive study on flight path selection algorithms. In *The lEE Seminar on Target Tracking: Algorithms and Applications*, pages 77 – 90, Birmingham, United Kingdom, March 2006.

[22] W. Guoshi, L. Qiang, and G. Lejiang. Multiple UAVs routes planning based on particle swarm optimization algorithm. In *2nd International Symposium on Information Engineering and Electronic Commerce*, pages 1 – 5, Ternopil, Ukraine, 2010.

[23] R. Henriques, F. Bacao, and V. Lobo. UAV path planning based on event density detection. In *2009 International Conference on Advanced Geographic Information Systems & Web Services*, pages 112 – 116, Cancun, Mexico, 2009.

[24] A. Herbon, E. Khmelnitsky, and O. Maimon. Effective information horizon length in measuring offline performance of stochastic dynamic systems. *European Journal of Operational Research*, pages 688–703, 2004.

[25] A. Herbon, E. Khmelnitsky, O. Maimon, and Y. Yakubov. Reduction of future information required for optimal control of dynamic systems: A pseudostochastic model. *IEEE Transactions on Automatic Control*, pages 1025–1029, 2003.

[26] G. M. Hoffmann and C. J. Tomlin. Mobile sensor network control using mutual information methods and particle filters. *IEEE Transactions on Automatic Control*, pages 32–47, 2010.

[27] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin. Distributed cooperative search using information-theoretic costs for particle filters, with quadrotor applications. In *AIAA Guidance,Navigation, and Control Conference and Exhibit*, August 2006.

[28] Z. Hongguo, C. Changwenl, H. Xiaohui, and L. Xiang. Path planner for unmanned aerial vehicles based on modified PSO algorithm. In *Proc. of the 2008 IEEE International Conference on Information and Automation*, pages 541–544, Hunan, China, June 2008.

[29] X. Hu, Y. Shi, and R. Eberhart. Recent advances in particle swarm. *2004 Congress on Evolutionary Computation*, 1:90–97, 2004.

[30] V. P. Jilkov and X. R. Li. On fusion of multiple objectives for UAV search & track path optimization. *Journal of Advances in Information Fusion*, 4(1):27–39, June 2009.

[31] V. P. Jilkov, X. R. Li, and D. DelBalzo. Best combination of multiple objectives for UAV search & track path optimization. In *Proc. 10th International Conference on Information Fusion*, pages 1–8, Québec City, Canada, July 2007.

[32] Y. Jin, A. A. Minai, and M. M. Polycarpou. Cooperative real-time search and task allocation in UAV teams. In *Proc. of the 42nd IEEE Conference on Decision and Control*, pages 7–12, Maui, Hawaii, December 2003.

[33] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proc. IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, Perth, WA, Australia, December 1995.

[34] J. B. Lasserre. Detecting planning horizons in deterministic infinite horizon optimal control. *IEEE Transactions on Automatic Control*, pages 70–72, 1986.

[35] J. Lee, R. Huang, A. Vaughn, X. Xiao, K. Hedrick, M. Zennaro, and R. Sengupta. Strategies of path-planning for a UAV to track a ground vehicle. In *Autonomous Intelligent Networks and Systems Conference*, Menlo Park, CA, July 2003.

[36] W. Lei, K. Qi, X. Hui, and W. Qidi. A modified adaptive particle swarm optimization algorithm. In *IEEE International Conference on Industrial Technology*, pages 209–214, Hong Kong, China, December 2005.

[37] S. Li, X. Sun, and Y. Xu. Particle swarm optimization for route planning of unmanned aerial vehicles. In *Proc. of the 2006 IEEE International Conference on Information Acquisition*, pages 1213–1218, Weihai, Shandong, China, August 2006.

[38] W. Li and C. G. Cassandras. A cooperative receding horizon controller for multivehicle uncertain environments. *IEEE Transactions on Automatic Control*, 51(2):242–257, 2006.

[39] X. R. Li, R. R. Pitre, V. P. Jilkov, and H. Chen. A new performance metric for search and track missions. In *The 12th International Conference on Information Fusion*, pages 1100–1107, Seattle, WA, July 2009.

[40] R. L. Lidowski, B. E. Mullins, and R. O. Baldwin. A novel communications protocol using geographic routing for swarming UAVs performing a search mission. In *2009 IEEE International Conference on Pervasive Computing and Communications*, pages 1–7, Galveston, TX, March 2009.

[41] S. R. Martin and A. J. Newman. The application of particle swarm optimization and maneuver automatons during non-markovian motion planning for air vehicles performing ground target search. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2605–2610, Nice, France, September 2008.

[42] D. Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. *IEEE Transactions on Automatic Control*, pages 814–824, 1990.

[43] B. Nabbe and M. Hebert. Where and when to look how to extend the myopic planning horizon. In *Proc. of the International Conference on Intelligent Robots and Systems*, pages 920–927, Las Vegas, NV, October 2003.

[44] N. Nigam and I. Kroo. Persistent surveillance using multiple unmanned air vehicles. In *IEEE Aerospace Conference*, pages 1–14, Big Sky, MT, June 2008.

[45] I. K. Nikolos. Coordinated UAV path planning using differential evolution. In *Proc. of the 13th Mediterranean Conference on Control and Automation*, pages 549–556, Limassol, Cyprus, June 2005.

[46] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras. Evolutionary algorithm based offline/online path planner for UAV navigation. *IEEE Transactions on Systems, Man, and Cybernetics– Part B: Cybernetics*, 33(6):898–912, 2003.

[47] D. Pack and G. York. An extended time horizon search technique for cooperative unmanned vehicles to locate mobile RF targets. In *The 2005 International Symposium on Collaborative Technologies and Systems*, pages 333–338, St Louis, MO, May 2005.

[48] F. Pan, X. Hu, R. Eberhart, and Y. Chen. A new UAV assignment model based on PSO. In *2008 IEEE Swarm Intelligence Symposium*, pages 1–7, St Louis, MO, September 2008.

[49] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization*. Dover, New York, 1998.

[50] H. Peng, F. Su, Y. Bu, G. Zhang, and L. Shen. Cooperative area search for multiple UAVs based on RRT and decentralized receding horizon optimization. In *Proceedings of the 7th Asian Control Conference*, pages 298–303, Hong Kong, China, August 2009.

[51] R. R. Pitre. Modified particle swarm optimization for search missions. In *40th Southeastern Symposium on System Theory*, pages 362–365, New Orleans, LA, March 2008.

[52] R. R. Pitre, X. R. Li, and D. DelBalzo. A new performance metric for search and track missions 2: Design and application to UAV search. In *The 12th International Conference on Information Fusion*, pages 1108–1114, Seattle, WA, July 2009.

[53] R. R. Pitre, X. R. Li, and D. DelBalzo. UAV route planning for joint search and track missions—an information-value approach. *IEEE Transaction on Aerospace and Electronic Systems*, To appear.

[54] A. Pongpunwattana. *Real-Time Planning for Teams of Autonomous Vehicles in Dynamic Uncertain Environments*. PhD thesis, University of Washington, 2004.

[55] A. Pongpunwattana and R. Rysdyk. Real-time planning for multiple autonomous vehicles in dynamic uncertain environments. *Journal of Aerospace Computing, Information, and Communication*, pages 580–604, 2005.

[56] D. Rathbun, S. KrageLund, A. Pongpunwattana, and B. Capozzi. An evolution based path planning algorithm for autonomous motion of a UAV through uncertain environments. In *Proc. to the 21st Digital Avionics Systems Conference*, pages 8D2–1–8D2–12, October 2002.

[57] S. Rathinam, P. Almeida, Z. W. Kim, S. Jackson, A. Tinka, W. Grossman, and R. Sengupta. Autonomous searching and tracking of a river using an UAV. In *Proc. of the 2007 American Control Conference*, pages 359–364, New York, NY, July 2007.

[58] S. Rathinam and R. Sengupta. Lower and upper bounds for a multiple depot UAV routing problem. In *Proc. of the 45th IEEE Conference on Decision and Control*, pages 5287–5292, San Diego, CA, December 2006.

[59] N. Roy and C. Earnest. Dynamic action spaces for information gain maximization in search and exploration. In *Proc. of the 2006 American Control Conference*, pages 1631–1636, Minneapolis, MN, June 2006.

[60] A. Ryan and J. K. Hedrick. Particle filter based information-theoretic active sensing. *Robotics and Autonomous Systems*, pages 574–584, May 2010.

[61] T. Schouwenaars, J. How, and E. Feron. Receding horizon path planning with implicit safety guarantees. In *Proc. of the American Control Conference*, pages 5576–5581, Boston, MA, June 2004.

[62] Y. Shi and R. C. Eberhart. Fuzzy adaptive particle swarm optimization. *Proceeding of the 2001 Evolutionary Computation*, pages 101–106, 2001.

[63] Y. Shi and R. C. Eberhart. Empirical study of particle swarm optimization. *Proc of the 1999 Evolutionary Computation*, pages 1945–1950, September 1999.

[64] A. Sinha, T. Kirubarajan, and Y. Bar-Shalom. Autonomous ground target tracking by multiple cooperative UAVs. In *2005 IEEE Aerospace Conference*, March 2005.

[65] A. Sinha, T. Kirubarajan, and Y. Bar-Shalom. Autonomous surveillance by multiple cooperative UAVs. In *Proc. of SPIE Signal and Data Processing of Small Targets*, pages 616–627, San Diego, CA, September 2005.

[66] P.B. Sujit and R. Beard. Multiple UAV path planning using anytime algorithms. In *2009 American Control Conference*, pages 2978–2983, St. Louis, MO, June 2009.

[67] P.B. Sujit and D. Ghose. Search using multiple UAVs with flight time constraints. *IEEE Transactions on Aerospace and Electronic Systems*, pages 491–510, 2004.

[68] P.B. Sujit and D. Ghose. Multiple UAV search using agent based negotiation scheme. In *2005 American Control Conference*, pages 2995 – 3000, August 2005.

[69] P.B. Sujit, A. Sinha, and D. Ghose. Multi-UAV task allocation using team theory. In *Proc. of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005*, pages 1497 – 1502, December 2005.

[70] T. Sun, C. Huo, S. Tsai, and C. Liu. Optimal UAV flight path planning using skeletonization and particle swarm optimizer. *IEEE Congress on Evolutionary Computation*, 1:1183–1188, 2008.

[71] X. Tian, Y. Bar-Shalom, and K. R. Pattipati. Multi-step look-ahead policy for autonomous cooperative surveillance by UAVs in hostile environments. In *Proc. of the 47th IEEE Conference on Decision and Control*, pages 2438–2443, Cancun, Mexico, December 2008.

[72] J. Tisdale, A. Ryan, Z. Kim, D. Tornqvis, and J. K. Hedrick. A multiple UAV system for vision-based search and localization. In *2008 American Control Conference*, pages 2438–2443, Seattle, WA, June 2008.

[73] W. H. van Willigen, M. C. Schut, A. E. Eiben, and L. J. H. M. Kester. Online adaptation of path formation in UAV search-and-identify missions. In *Proc. of the 10th international con-*

ference on Adaptive and natural computing algorithms, pages 1985–1990, Ljubljana, Slovenia, April 2011.

[74] G. Yang and V. Kapila. Optimal path planning for unmanned air vehicles with kinematic and tactical constraints. In *Proc. of the 41st IEEE Conference on Decision and Control*, pages 1301–1306, Las Vegas, NV, December 2002.

[75] U. Zengin and A. Dogan. Real-time target tracking for autonomous UAVs in adversarial environments: A gradient search algorithm. *IEEE Transactions on Robotics*, pages 294–307, April 2007.

[76] U. Zengin and A. Dogan. Real-time target tracking for autonomous UAVs in adversarial environments: A gradient search algorithm. In *Proc. of the 45th IEEE Conference on Decision & Control*, pages 697–702, San Diego, CA, December 2006.

[77] W. Zhenhua, Z. Weiguo, S. Jingping, and H. Ying. UAV route planning using multiobjective ant colony system. In *IEEE International Conference on Cybernetics and Intelligent Systems*, pages 797–800, Chengdu, China, September 2008.

Vita

Ryan Pitre was born in Los Angeles, California and was raised in Metairie, Louisiana. He attended Airline Park Elementary School, T. H. Harris Jr. High School, and graduated from East Jefferson High School. He obtained his Bachelor's degree in electrical engineering from the University of New Orleans in 2002. In 2003, he became Professor X. Rong Li's student and joined the Information Systems Laboratory at UNO. Ryan received his Master's degree in electrical engineering in 2004. He has authored or coauthored five conference papers and one journal paper.