Search and Pursuit with Unmanned Aerial Vehicles in Road Networks

Michael Dille

CMU-RI-TR-13-30

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

November 2013

Thesis Committee:
Sanjiv Singh, Chair
Benjamin Grocholsky
Maxim Likhachev
Paul Scerri
Thanasis Kehagias, Aristotle University, Greece

Abstract

Across many rescue, surveillance, and scientific applications, there exists a broad need to perform wide-area reconnaissance and terrain surveys, for which unmanned aerial vehicles (UAVs) are increasingly popular. This thesis considers the task of using one or more UAVs to locate an object of interest, provide continuous viewing, and rapidly re-acquire tracking should it be lost for any reason.

For both the common class of small field-launched UAVs considered as well as larger UAVs, this is a difficult task due to a small available sensor field of view, uncertain estimates of UAV pose, and limited maneuverability relative to the scale of the environment, requiring constant processing of observations and recomputation of flight paths or sensor aiming to best find the object or keep it in view. Existing strategies for accomplishing this provide poor estimates of the object's location and rely on grossly heuristic or computationally intensive trajectory generation for both pursuit and search.

This thesis proposes careful representation of observation uncertainty and exploitation of environmental structure—with particular focus on road networks typical of urban-like areas—as means to simplify and better model the problem. For the case of actively tracked objects, greatly improved location estimates are demonstrated through filter representations designed for high-uncertainty observations, as is increased pursuit performance by modeling terrain-constrained space reduction in object location and motion. Objects having no or only roughly known prior location require an initial search, for which both classical Bayesian probabilistic search (for stochastically-modeled moving objects) and novel road network coverage strategies (for stationary or slow-moving objects) are considered. Finally, this is extended to search and local recapture of evasive adversaries in road networks through novel mappings of pursuit-evasion tactics that are well-studied in abstract or ground-based domains but have yet to see use in physical, particularly aerial, applications.

Estimation and tracking aspects have been validated in extensive field trials using widely-fielded air vehicles, and other components have been evaluated in realistic simulation using similarly parameterized vehicle models and control interfaces, laying the groundwork to directly apply the demonstrated algorithms on real aircraft.

Acknowledgements

First and foremost, I owe a great debt of thanks to family and friends for their encouragement and patience throughout this long endeavor.

The support and guidance of the thesis committee in the preparation and improvement of this document and the underlying work, especially the always-enthusiastic efforts by Thanasis and many thoughtful discussions with Ben, are deeply appreciated.

Much of the work presented in Chapter 4 is joint work as part of several projects in the Field Robotics Center at the Carnegie Mellon University Robotics Institute. Portions of the described visual tracking and static geolocation effort were led by Stephen Nuske, and nonlinear estimation strategies were led by Ben Grocholsky. Additional thanks are owed to Ben and Steve for entertaining and stimulating discourse on all manner of topics.

Portions of this work received the support of US Army RDECOM-ARDEC (in collaboration with iRobot Corporation), the United Technologies Research Center, and a DoD National Defense Science and Engineering Graduate (NDSEG) fellowship.

Contents

CHAPTER 1

Introduction

1.1 Motivation

Wide-area reconnaissance and survey missions are a key element of many search and rescue, disaster response, military reconnaissance, law enforcement surveillance, and scientific exploration scenarios. Examples include searching for missing persons in the wilderness or at sea, detecting and tracking the spread of wildfires, locating and pursuing fleeing suspects, protecting a perimeter for a moving convoy, finding meteorites, mapping floating ocean debris, and identifying and tracking the movement of wildlife.

The character of such missions has evolved with social and computing trends, greatly affecting approach methodology. Typically, these are extremely data-driven and information rich, since a terrain map (e.g. from satellite imagery) and prior information on the relative priority of area regions are likely to be available, and mission results are frequently aggregated—often in real-time—for visualization or pattern detection. These missions are also extremely dynamic in that additional information—from witness reports, external sensors, or other responding agencies—may arrive at any time, and a well-executed mission plan must be able to react and adapt as necessary. Time-sensitivity is another probable characteristic, both because lives may be at stake and due to large operational costs, implying a need for rapidly field-deployed sensing, some form of optimal mission planning to minimize execution time, and automatic information passing. Finally, considerable risk-aversion is likely

(a) Wildfire detection and monitoring

(b) Flood survivor identification

(c) Tracking pirate activity

(d) Convoy perimeter protection

(e) Pursuit of fleeing suspects

Figure 1.1: Example scenarios requiring substantial reconnaissance and survey components over a wide area.

to play a role, requiring that mission actions be thoroughly documented, follow strict procedures, and minimize the use of human intervention to avoid introducing errors or placing further lives at risk, all of which suggest extensive use of automation.

Particular commonalities of such scenarios may be further distilled as a step towards formalization. In general, these include one or more *observers* seeking some object of interest (henceforth simply referred to as the *target*). Potential prior knowledge about the target's possible location that can be updated with every observation (or lack of observation), and if mobile, the target (by virtue of being an inanimate object, wild animal, fleeing suspect, or an out-of-communication victim) will be non-cooperative and its motions non-deterministic. Once detected, multiple observations or persistent observation is desirable, and each observation may have some quality or error dependent on the relative pose of the observer and target (for instance due to terrain occlusion, range to the target, or intrinsic sensor properties). Meanwhile, the environment in which the mission takes place is likely to be hazardous and poorly traversable, but its structure well known from existing maps. Together, these properties suggest a general mission structure and desirable observer characteristics.

Independently, unmanned aerial vehicles (UAVs) have come to form the basis of a rapidly growing field of study and are seeing increasing real-world use, frequently

(a) RQ-4 GlobalHawk
35m wingspan
3,800kg empty

(b) RQ-8A Fire Scout
8m rotor diam.
940kg empty

(c) Scout
80cm body diam.
1.3kg typical

(d) Hummingbird
16cm wingspan
19g total

Figure 1.2: Several unmanned aircraft demonstrating a spectrum of portability roughly parameterized by the magnitude of vehicle size, weight, cost, feature set, flight altitude, mission duration, and launch facilities. These include (a) the Northrop Grumman RQ-4 Global Hawk, (b) the Northrop Grumman RQ-8 Fire Scout, (c) the Aeryon Scout, and (d) the AeroVironment Nano Hummingbird.

for such missions. Their viability may largely be attributed to advances in sensor miniaturization, computing power, battery chemistry, and composite materials, coinciding with increases in commercial production volume in each area for use in consumer electronics. Simultaneously, a shift in defense strategy towards unmanned vehicles, particularly aircraft, [89] has resulted in a major infusion of research funding in the area and the development of a considerable variety of UAVs by new and well-established companies. Combined, these advances and industry shifts have made available a wide spectrum of reliable, capable, and relatively affordable UAVs ranging in size from tens of grams to over ten thousand kilograms.

UAVs are particularly well suited to the aforementioned scenarios, having a number of advantages over ground vehicles. Most are rapidly deployable from a distance, without a need to assemble and transport human teams into an area—after verifying it is safe to do so—for a manned mission or to deploy an unmanned ground vehicle (UGV) within the affected area. Further, they are fast moving and thus able to reach the area of interest and cover a wide space quickly. By virtue of being airborne, UAVs are not subject to terrain obstacles, debris, and hazards below their flight altitude, while a wide area view is possible with minimal occlusion from obstacles ground vehicles could not see beyond. Additionally, given this lack of occlusion, reliable and long-range line-of-sight communication from a ground station or satellite is feasible.

Within the broad category of reconnaissance UAVs, a further distinction may be made along a spectrum of what might generally be termed vehicle portability—parameterized by increasing size, weight, cost, feature set, flight altitude, mission duration, and facilities required for deployment—depicted in Figure 1.2. At one extreme lies, for instance, the high-altitude Northrop Grumman RQ-4 Globalhawk,

with a wingspan of 35 meters, 36-hour endurance, a complex sensor package including high-resolution Synthetic Aperture Radar, and a unit cost of approximately US$200 million, amortizing development expenses. [132] At the other end may be found the likes of the AeroVironment Nano Hummingbird, with a wingspan of 16 centimeters and weighing 19 grams, intended for short-duration hovering or perching surveillance in urban environments using only a low-resolution onboard camera. [4]

In between lies the class of aircraft considered as the focus of this thesis: small, human- or van-portable field-reconnaissance UAVs, having a wingspan of one to three meters, weighing one to ten kilograms, and flying at between 50 and several hundred meters above ground level. Commonly called small UAVs (SUAVs) or micro air vehicles (MAVs), these are most appropriate for a wide range of rapid-response reconnaissance missions in hazardous terrains for several reasons. First, they may be launched in minutes from a small clearing, with little to no infrastructure such as a runway or launch-propulsion equipment, very near the area of interest. Further, launch and operation need be performed by at most several lightly-trained operators using portable battery-powered ground stations. Finally, their relatively low cost—one to several tens of thousands of dollars—admits a substantial level of expendability, permitting particularly risky yet possibly high-reward missions over hostile or inaccessible terrain from which retrieval is impractical. However, design simplification necessary for weight, cost, and operator workload minimization results in a number of drawbacks. Control surfaces such as ailerons are often eliminated, limiting maneuverability and weakening control authority necessary to reject wind disturbances. Onboard autopilots may provide a simple (e.g. waypoint-based) high-level interface that benefits operators but which limits the complexity of algorithmically-generated flight paths. Onboard sensing is limited to lower-end (lightweight and low-cost) inertial measurement units (IMUs) and global positioning systems (GPSes) providing limited accuracy, with low- to medium-resolution cameras. This is aggravated by infrastructure-less recovery methods that are often little more than controlled crashes, the impacts of which disturb calibrated relative sensor mountings. Finally, limited processing power is available onboard, requiring that primarily compressed raw sensor data (commonly camera video) rather than smaller processed data (such as image pixel locations of detected targets) be sent over limited-bandwidth downlinks. Common designs therefore include relatively narrow-angle lensing to provide the highest-resolution imagery for a given area of ground, which, combined with low-altitude flight, produces a small sensor footprint. The result is a very distinct reconnaissance problem from that encountered with larger, higher-altitude UAVs: rather than needing to detect and identify objects of interest from within a wide sensor footprint (primarily a signal processing task), motion of the UAV itself to

direct a smaller sensor footprint toward likely target locations or vantage points (an estimation and planning task) is required.

(a) ScanEagle
3m wingspan
13kg empty

(b) Joker 2
1.8m rotor diam.
8kg loaded

(c) RQ-16 T-Hawk
35cm diam.
8.4kg typical

(d) RQ-11 Raven
130cm wingspan
1.9kg typical

Figure 1.3: Examples of small field-reconnaissance UAVs considered primarily in this thesis, such as the Insitu ScanEagle catapult-launched and hook-retrieved tactical surveillance UAV, (a) the Maxi-Joker 2 camera-transport helicopter, (b) the Honeywell RQ-16 T-Hawk vertical takeoff and landing ducted-fan UAV, (c) and the hand-launched AeroVironment Raven local reconnaissance UAV. (d)

At the same time, the choice of focus on small UAVs should not be taken to greatly limit the applicability of ideas presented in this thesis. Such vehicles are chosen as a lowest common level of capability and as a typical case fulfilling an important role. Arbitrarily larger UAVs will inevitably face many of the same difficulties when operating in sufficiently large environments, in which their speed, field of view, or turning radius are limited relative to the scale of or pace of activity within an environment. As a particular example, large UAVs providing wide-area surveillance may often do so by either viewing the area through a high-resolution imager of which only a small sub-window is sent over a downlink (due to bandwidth and processing

constraints) or by viewing the area through a sensor mounted on a gimbal that must be aimed at the actual sub-area of interest. In both cases, the effective field of view is likely to be small relative to the environment, and the act of slewing a sub-window or gimbal is analogous to moving a small aircraft, in that the location of focus must be chosen and that some delay is present in its execution.

1.2 General Problem Statement

Using one or more UAVs to locate a target and provide persistent surveillance that generates an estimate of the target's location[1] is both an extremely desirable task in practice as well as the implicit objective of a considerable amount of literature on UAV search and tracking. Nevertheless, it appears to only have been first formally stated by Frew and Elston as the continuous cooperative search, acquisition, and tracking (CSAT) problem for active robot sensor networks. [43] In that framework, the three components are (1) an area search or coverage control task to traverse a region with a finite sensor footprint while optimizing a performance objective to detect one or more targets, (2) solving a task assignment or resource allocation task to assign each robot to a location somehow best covering all targets, and (3) performing cooperative tracking and estimation to both guide the motion of robots around detected targets and estimate target locations. Though phrased generically, it is most applicable to UAVs in that it considers robots with motion constraints, finite sensor footprints coupled to a robot's location, and inaccurate observations used to produce an estimate of target location.

As the focus of this thesis are the elements of search and tracking, the CSAT components defined by Frew of acquisition (target identification) and observer assignment will not be considered explicitly and instead treated as part of the transition between search and tracking. The problem may then be modeled as depicted in Figure 1.4, in which a target is first sought while in the *search* state, then, once detected, actively tracked via continuous *surveillance and geolocation* while observations are received. If tracking is lost (for instance, due to too much time elapsing since the last observation), the search state is re-entered to *recapture* the target. Owing to important differences between a target having completely unknown location that must be found and one that has been recently lost, recapture is treated as an independent mission state in the more formal problem decomposition of Section 2.2, however in

[1]The task of processing sensor observations (in sensor-local or vehicle-local coordinates) and converting them into a best estimate of a target's location in world (geographic) coordinates is commonly called *geolocation* and will be henceforth referred to as such.

its simplest form the problem may be viewed as having two high-level states: one in which the target is being observed, and another in which it is not and must be first located.

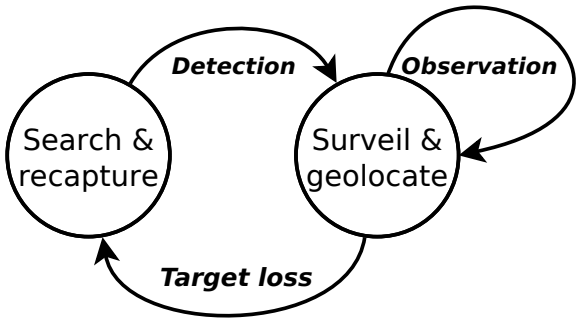


Figure 1.4: State diagram of the cooperative search, acquisition, and tracking problem, with the acquisition step folded into the transition between search and tracking.

Problem Subtasks

The high-level subtask loops of these two states and their interpretations in the context of this thesis, repeated continuously while in either state, may be stated as follows:

Search

1. Propagate prior knowledge of possible target position to present time.

   A prior map of possible target locations—perhaps stored as a map of relative probability or unsearched regions—may be available, and if so, it must be updated with the passage of time to reflect possible target motion into previously cleared or unlikely regions.

2. Move to optimize detection likelihood, prevent escape, or both.

   Flight paths are chosen to steer UAV sensor footprints to optimize one or more performance metrics, such as maximizing the expected number of targets detected in a given interval, minimizing the expected time until a target is detected, minimizing the probability of escape, or possibly guaranteeing that any targets in an area will be detected.

3. Process sensor readings to determine whether detected.

   As areas are traversed by a sensor footprint, a target detection or recognition

mechanism must be executed to report whether a target is in view, initiating a transition to the tracking state.

4. Record searched areas and update possible target locations.

   Knowledge that a target is not present in a given region may affect future flight paths, for instance by marking the area as cleared or having reduced likelihood of containing a target, and a prior target location map indicating this may be provided to the next iteration of the search.

Tracking

1. Process sensor readings for continuous tracking.

   Sensor readings must be processed to determine whether the tracked target is currently in view, and if so, generate an observation.

2. Build and refine sensor detection or tracking model.

   Given repeated observations of a target, it may be possible to improve detection accuracy or speed by automatically optimizing detection parameters for the currently tracked target, effectively learning its appearance, fingerprint, or signature that may change over time with the observer's perspective.

3. Filter observations to estimate target location.

   Every successful observation of a target may be used to improve or update its geolocation estimate, producing an estimate more accurate than any single observation.

4. Anticipate target motion between observations.

   The target's geolocation history may be used to predict its future location at the time of the next anticipated observation, for instance by computing and apply a velocity estimate or determining that the target is likely to be stationary.

5. Move to improve estimate certainty and maintain view.

   To maintain tracking, a path for each sensor footprint must be generated so as to produce good observations in the future, for instance by maintaining its placement over the target's predicted location.

Scope of Study

As the search and tracking problem as stated is incredibly broad and any practical implementation extremely complex, a number of assumptions and simplifications

must be made for the purpose of this thesis. Except as noted, the following facets will be neglected for the remainder of this document and are assumed to be separately provided or resolved by existing or independent modules. Many have also received considerable attention in related literature. Key facets that will not be considered beyond references to related work in Section 3.2 include:

- Self-localization and environment mapping
- Automatic target recognition or data association
- Aerodynamics and low-level vehicle control
- Sensor aim or gimbal state planning
- Obstacle avoidance and agent deconfliction
- Target occlusion by terrain
- Distributed or decentralized estimation and control

Related Problems

The problem of continuous search and tracking for surveillance and geolocation is quite similar to but sharply distinct from a number of existing well-studied problems. Some of the most related include planar area coverage, (polygonal) region clearing, generalized pursuit-evasion, and so-called moving target indicator (MTI) tracking using radar. A brief statement of these problems and their differences is provided in Section 2.3.

1.3 Summary of Contributions

The general theme of this thesis is to simplify the search and pursuit problem by exploiting natural terrain constraints on a target that may be present, thereby reducing resource requirements and improving observable performance, in terms such as search time, geolocation error, frequency and duration of target loss, and recapture time. The form of such constraints considered is primarily that of roads within environments, though generalizations are also considered.

To accomplish this, it is proposed that results from several related fields be leveraged as toolboxes, with aspects and instances of this problem mapped to abstract problem instances for which theoretical results or tractable algorithms are known.

These include the use of constrained and multi-modal estimation for geolocation from existing large-scale tracking and guidance problems (Section 4.3), search for constrained adversarial targets from the field of graph search (Chapter 6), and potential avenues for discrete recapture and local search strategies from pursuit-evasion (Section 6.4)

High-level contributions of this thesis include:

- A detailed study of the search and tracking problem including a decomposition into more specific problems as parameters are varied (Section 2.2)
- A study of and development of strategies for recapture as its own special case distinct from search or pursuit (Section 3.5)
- Improved geolocation representations and filters for UAVs providing highly uncertain observations (Section 4.2)
- A demonstration of improved target pursuit by exploitation of road-network graph structure applying constrained estimation (Section 4.3.1)
- A practical open area search strategy optimized for high-level autopilots and slowly reacting target detectors (Section 5.1)
- An approximation of a UAV camera's field of view of the ground (its so-called sensor footprint) that provides a large computational improvement for action-tree based trajectory planners for probabilistic search (Section 5.2.2)
- A novel coverage search method for road networks formulated as one of several location visitation sequencing problems (Section 5.3)
- A search strategy built upon abstract tree search providing guaranteed capture of adversarial targets in road networks having infinite or unmodeled motion capabilities (Section 6.2)
- Two extensions of this strategy for targets having a known bound on their speed, providing guaranteed recapture or local search (Section 6.3)
- Real-world implementation and field validation of many of the proposed ideas and realistic simulation of real-time implementations of others in a form readily applied to existing fielded vehicles (Section 4.1)

These ideas may be summarized in the statement of this thesis:

> *By exploiting natural terrain constraints on a target's location and motion, the tractability and performance of UAV search and pursuit may be greatly improved.*

1.4 Document Outline

The remainder of this document is structured as follows:

Chapter 2 presents subcases of the general search and tracking problem arising as parameters are varied, defines the task optimization problem in terms of observation inputs and control outputs, and highlights urban-like environments in particular as worthy of study.

Chapter 3 summarizes existing work in target search, geolocation, and pursuit and considers challenges both due to weaknesses in these approaches and fundamental to the tasks.

Chapter 4 details target geolocation and pursuit elements including experiments providing a practical need for resulting improvements in search and pursuit representations, methods for improved target geolocation with high-uncertainty observations, and the geolocation accuracy and task performance benefits derived from explicitly considering road constraints.

Chapter 5 considers the case of search for non-adversarial targets (stationary or whose motion may be stochastically modeled), evaluating classical discrete Bayesian estimation as an efficient road network search mechanism and proposing a novel coverage strategy cast as a location visitation sequencing problem.

Chapter 6 addresses the case of search for evasive targets having either unlimited (or otherwise unmodeled) speed or some known bound on speed and provides strategies derived from mapping to simpler search abstractions that are also suitable for recapture of lost or fleeing targets.

Chapter 7 summarizes completed work and suggests possible future work that both may represent feasible thesis extensions and interesting ideas not within its scope.

CHAPTER 2

Problem Study

This chapter examines and more precisely defines the broad search and tracking problem studied thus far. Section 2.1 presents such a definition and phrases it as an abstract optimization task with varying objectives depending on the task phase and mission requirements. It is then treated as a general continuous sensor positioning problem that reduces to one of several simpler cases as properties of the target are varied in Section 2.2. Next, Section 2.3 lists several related problems and explains their distinctions. Section 2.4 raises the importance of using environmental knowledge to guide the choice of approach given its effect on possible target position and motion. The special case of road networks as an instance of this is motivated in Section 2.5, with suggestions for broader generalization to environments containing more open areas considered in Section 2.6. Finally, Section 2.7 chooses simple but largely realistic models for UAV sensing and motion to permit later formalization, algorithmic development, and results comparison.

2.1 Problem Definition

The search and tracking task considered may be concisely stated as:

> Locate a potentially evasive target in a bounded environment using one or more UAVs, provide persistent sensor coverage and continuous geolocation estimates once it is detected, and reacquire tracking when lost.

The first phase is a *search* task in which a trajectory for each UAV is sought that best (most quickly or with the greatest assurance) detects a target. The second is primarily a *pursuit* task in which trajectories are sought that maintain a sensor footprint over the target that provides good observations for geolocation. Reacquisition attempts to *recapture* a target for which tracking was lost by performing a local search before uncertainty in the target's position resulting from possible motion grows too large. Not considered as an independent mission state in the original CSAT formulation, a recapture phase is considered explicitly throughout this thesis given important distinctions in target location knowledge from full-fledged search.

The general problem may be phrased as a UAV control optimization problem in which a flight path is chosen for each UAV with one of the following objective functions chosen based on phase and varying mission requirements.

Locating or re-acquiring:

- Maximize probability of detection over given search duration
- Minimize expected time until detection
- Minimize search time providing at least a given probability of detection (Important special case: p=1, or guaranteed search or recapture)

Tracking:

- Maximize expected time in view over a given tracking duration
- Minimize expected average or terminal geolocation uncertainty over a given tracking duration
- Minimize expected tracking time until geolocation uncertainty drops below a given threshold

Optimizing these objectives may be formally, albeit for now quite abstractly, stated. Let $\mathbf{x}_{\text{UAV}}$ be the full state of the UAV and $\mathbf{x}_{\text{tgt}}$ represent all information known about possible target position. Let $\mathcal{U}$ be the space of UAV control inputs at a given instant for all UAVs, elements of which depend on the precise motion model considered but which may for example be of the form of a bank angle to hold, a waypoint to head towards, a thrust vector to apply, or a multi-dimensional vector of subsystem commands (e.g., a rudder command and sensor pointing angle). Each implicitly contains a duration for which to apply the control command. Likewise, let

$J(\mathbf{U}, \mathbf{x}_{\text{UAV}}, \mathbf{x}_{\text{tgt}})$ be the cost function evaluating a discrete-time control plan $\mathbf{U} \in \mathcal{U}^k$ of length k under a given objective. Then,

$$\mathbf{U}^* = \underset{\mathbf{U}}{\operatorname{argmin}}\, J(\mathbf{U}, \mathbf{x}_{\text{UAV}}, \mathbf{x}_{\text{tgt}})$$

is the optimal sequence of control inputs to apply. Depending on the objective, the sequence length k may be a fixed or free variable.

The UAV motion model defining $\mathcal{U}$ considered for the purpose of this thesis is given in Section 2.7 and specific cost functions for tracking and search are compared in Sections 3.3.3 and 3.4.2 respectively.

2.2 Problem Decomposition

An alternative view of the problem is as a continuous sensor positioning task with the immediate goal of providing useful observations of a target, implying the need to locate and reach it if not presently in view.

The three phases of the preceding section are then subcases of the same problem with differing certainty in target location. If a target's location is known—for instance because it is presently being tracked within the sensor footprint or had been very recently—the immediate objective of an observing UAV is *pursuit* of the target, moving so as to place the sensor footprint over the predicted future location of the target to maintain or quickly regain tracking and improve geolocation certainty. If a target's location is only roughly known—as lying within some small region or among a set of hypotheses—because it has not been observed recently, then *recapture* of the target is desired to prevent further growth in uncertainty. Lastly, if no estimate of a target's location is available or if it is at best contained within a large region, a wider-scale *search* is required to first locate the target to (re)commence pursuit. Characteristics of these cases and typical examples are summarized in Table 2.1, with intuitive graphical depictions given in Figure 2.1.

Another important aspect is knowledge of target motion, or simply target predictability. At one extreme, if a target is known to be *stationary* (or either has known velocity that may be directly subtracted or is cooperatively transmitting its location), the area simply need be covered until it is located, at which point the observer may choose flight paths around it providing the most desired observations. At the other, a completely *adversarial* or evasive target will presumably avoid detection and persistent surveillance, possibly requiring many UAVs executing carefully chosen trajectories to corner the target and block escape paths. In between lie *stochastically*

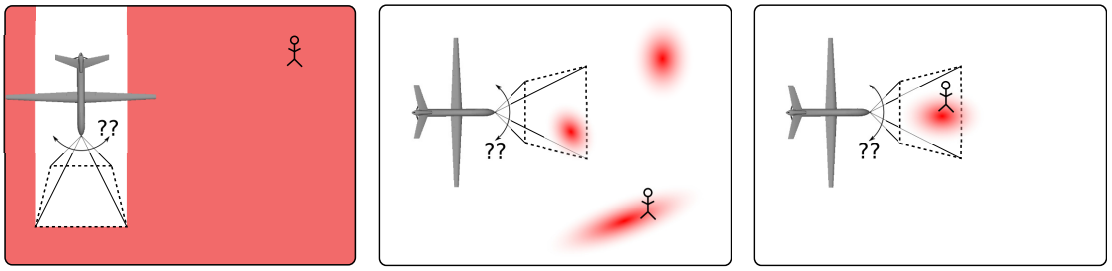
	Search	**Recapture**	**Pursuit**
Target location	No prior or weak regional prior	High-uncertainty estimate	Known (actively estimating)
Target observation	Possibly never	None recently	Recently observed
Examples	• Clearing an area containing an unknown number of targets • Identifying locations of targets known to exist within an area • Lost target not captured before uncertainty grew to entire area	• Pursued target was not in expected location after a brief loss of view • More targets than observers and must give up pursuit on one to check on another • Momentarily known location from beacon or remote intelligence and need to get back in view before loss	• Want to keep a target in view despite observer's or its motion • Target in view but location uncertainty is large or asymmetric • Target just left view and want to return to view quickly

Table 2.1: Characteristics of the three problem cases as target location certainty varies.

moving targets, the motion of which is not precisely known but is modeled by an uncertainty distribution. This latter category accounts for non-cooperative targets that might otherwise act adversarially but are ignorant of or indifferent to their pursuers' presence.

Figure 2.2 depicts a further decomposition of the problem along this additional axis of target predictability. This exposes substantially more strictly defined and, in some cases, well-studied problems to which the general task reduces for a given level of certainty in target location and its predictability.

For an effectively stationary target, the problem is fundamentally an active sensor placement task with an unchanging world. For an actively tracked target, trajectories are chosen that are most likely to provide the best observations, for instance which provide lowest (possibly a-posteriori) geolocation uncertainty. Existing strategies for accomplishing this as the simplest case of pursuit are considered in Section 3.3.3. Meanwhile, a target in need of recapture in this context is one that view of which has been lost (perhaps due to wind-induced trajectory tracking error), and trajectories are required that quickly move the sensor footprint back into positions over the target providing similarly good observations. Typically, the same existing strategies



(a) Search (b) Recapture (c) Pursuit

Figure 2.1: Intuitive depictions of pursuit, recapture, and search, along with the need to steer the path of the observer. Red shading indicates the relative probability of target presence, and the trapezoidal region leading the UAV indicates the region visible to it (formalized in Section 2.7).

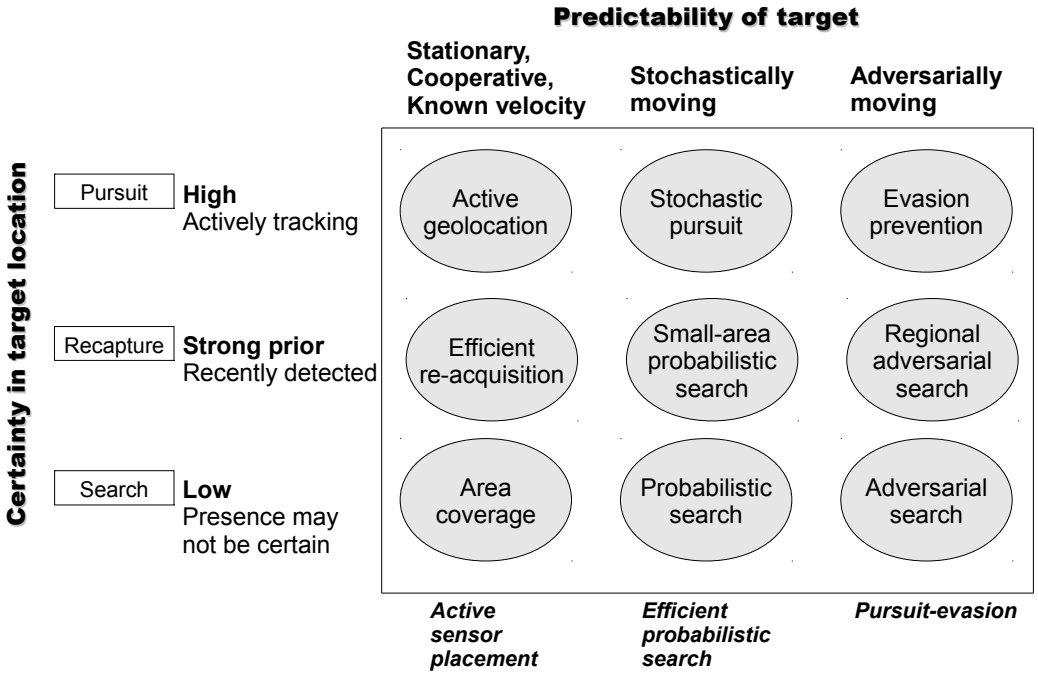


Figure 2.2: Decomposition of the unified search and tracking problem along the two axes of knowledge of target position and target predictability.

apply to this case in that observations of most likely target locations are desired, and no concern is present for the target escaping. A target of unknown location, however, must first be located. To do this for a stationary target requires covering all locations the target might be, possibly prioritized by a relative likelihood, but

again without concern for possible motion during the execution of the search that might recontaminate already-searched areas. Existing strategies for regular, uniform environments are reviewed in Section 3.4.1 and for irregularly-shaped or prioritized environments in Section 3.4.2, with an additional version contributed in Section 5.1 that is particularly practical for small UAVs of limited maneuverability.

The primary difference for a stochastically moving target is the need to continuously predict this motion and record resulting growth in uncertainty. For pursuit, this corresponds to predicting and diffusing likely future target locations when planning where to place sensor footprints for best observation. This corresponds to the more difficult general case of observation-predicting pursuit that existing work described in Section 3.3.3 has considered. Extensions for improved geolocation accuracy and planning tractability for simple road-like environments are presented in Section 4.3 and suggested for more general road networks in Section 4.3.3. Skipping to unknown-location targets requiring search, mere coverage is now insufficient because target motion may now, with varying likelihood, enter previously cleared areas. Existing strategies reviewed in Section 3.4.2 therefore consider probabilistic recontamination producing a changing irregular prior that is searched as for a stationary target with non-uniform location likelihood. Recapture of such targets has not previously been well considered, though fundamentally either a local-area probabilistic search or pursuit with a wide-area likelihood or multiple hypothesis may perform well. Intuitive possibilities are considered in Section 3.5, with viable recapture strategies for road network environments stemming from local instances of guaranteed search presented in Chapter 6.

Adversarial targets are of course the most challenging and are typically assumed to be able to move in any direction at any time, with either infinite or bounded speed, and surveilling them is fundamentally a pursuit-evasion task. Though a well-studied problem as a general abstract geometric problem, with several possibly-applicable abstractions considered in Section 3.1, this case has with few exceptions [102] not been considered for UAV pursuit and recapture. Given the applied nature of the task, this is likely due to the unrealism of omniscient high-speed targets, when in practice a real target might move a small fraction of the width of a sensor's field of view between observations. Search for fleeing or otherwise evasive targets by UAVs has received study and is often relevant simply when no confidence in any particular probabilistic motion model for a target is present. Existing methods are summarized in Section 3.4.1 for regularly-shaped environments and noted to be few for irregular environments in Section 3.4.2, while guaranteed search and recapture within road networks is presented in Chapter 6.

2.3 Related Problems

A number of existing problems that have received much previous attention address variations of aspects of the problem considered here. Some of the most related problems and a brief statement of differences include:

- Area coverage

  Generic algorithms to cover an area in the least time, such as producing a minimal carpeting through Boustrophedon or spanning-tree decomposition, may suffice to search for a non-adversarial target. However, a path these generate is intended as the path for the sensor footprint and must be translated to a UAV flight path, which may not be directly possible or require sharp turns a UAV cannot execute. Further, an open-loop coverage pattern must take into account both the finite width of the sensor footprint and the need to incorporate some overlap to address slight trajectory tracking errors due to wind. Several existing area coverage algorithms designed for UAV target search are considered in Section 3.4.1.

- Region clearing

  Generic formulations of clearing a polygon [56] or a discrete state space represented as a graph [92] have been studied as abstract search mechanisms and are discussed further in Section 3.1. Clearing may be useful as a search strategy when an unknown (and possibly zero) number of targets are present in an environment and the goal is to simply detect them all or verify none are actually present. A primary difference with these is that UAVs are not constrained to the same environment as the target—changing the problem rather completely. These formulations assume either a conical flashlight-like or (possibly bounded) omni-directional sensor model for the searcher that reaches outward into the environment containing a target, which is quite different from the sensing model here used for UAVs equipped with cameras or similar sensors defined in Section 2.7. In practice, area clearing is never guaranteed due to uncertainty in sensor pointing and target detection, requiring some accommodation for redundant observations to achieve a desired certainty threshold.

- Pursuit-evasion

  In general pursuit-evasion scenarios, also discussed further in Section 3.1, one or more pursuers attempts to reach or enclose a fleeing target, possibly well-modeling the case of a recapture search for a known target. A key difference from common formulations is that the position of a UAV's target is known with

only potentially large uncertainty until viewed and that it is unlikely that a real target may be truly adversarial in its knowledge of pursuer position and intent. Further, pursuit-evasion typically concludes at capture and would not necessarily place pursuers in positions conducive to persistent tracking.

- GMTI tracking

  Ground moving target indicators (GMTI) are extremely critical but little-publicized components of real-world large-scale targeting operations. Typically built on aerial Doppler radar sensors, they form the tracking portion of so-called look-down/shoot-down systems that modern defense forces worldwide have come to rely upon, with common implementations including the highly recognizable airborne early warning and control (AWACS) aircraft and radar nosecones of fighter jets shown in Figure 2.3. These systems typically operate at high-altitude (15km), can observe a wide-area of up to hundreds of square kilometers, and can report relative target locations with error in the tens of meters. The task of tracking with such sensors is largely one of target discrimination and data association, and the sheer scale, cost, and launch complexity is otherwise simply well beyond that considered for field-reconnaissance. General approaches to filtering geolocation observations using such sensors, however, remain largely applicable and are described further in Section 3.3.2.

(a) Boeing E-3 Sentry AWACS (b) MiG-31 Nosecone Radar

Figure 2.3: Examples of systems including GMTI sensing.

2.4 Effect of Environment on Problem Approach

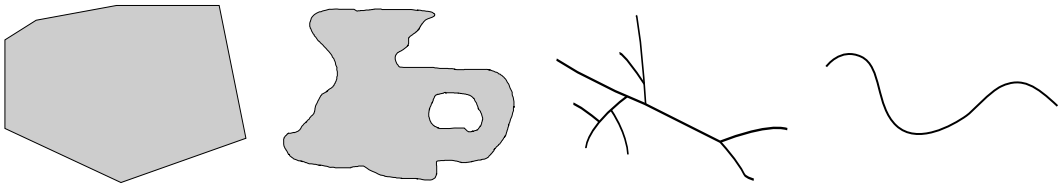
Thus far, the problem has been stated independently of the environment in which the task is to be performed. Indeed, for a given point in the decomposed problem space presented, the basic approach strategy–be it covering an entire area by any

efficient means to locate a stationary target or calculated sweeping to contain an adversarially moving target in a progressively shrinking region–remains the same regardless of the environment. However, the form and structure of an environment may critically affect both the difficulty of the task and the appropriateness of a given approach.

Possibly the two most important properties of an environment in this regard are its *sparsity* and *connectivity*, measuring the fraction taken up of the bounding convex hull in the continuous space in which it is embedded and the average number of locations to which a target may directly move in a short period of time from any given point, respectively. Although not particularly useful to quantify directly given that a numeric value of connectivity would be uncountably infinite for continuous environments, it is valuable to consider a spectrum of environment classes and their influence on choice of approach. General classes considered here are open and regularly shaped areas, target-obstacle ridden or irregularly shaped areas, collections of intersecting paths (such as road networks), and single paths. These are summarized in Table 2.2 with appropriate choices of representation of target location likelihood, which is most immediately affected by both factors given a need to define (at least implicitly) a possible target location set and a neighboring location set for each valid position. An accompanying intuitive graphical depiction of the class spectrum is shown in Figure 2.4.

	Open & regularly shaped	**Many obstacles or irregularly shaped**	**Intersecting paths**	**Single path**
Sparsity	Dense	Moderate	Very sparse	Arbitrary
Possible target motions	Continuous	Continuous	Discrete few	2 directions
Example likelihood store	• Continuous 2-D distribution • Dense grid • Unconstrained particles	• Densely connected cells • Constrained particles	• Multiple continuous 1-D distributions • Cells or particles lying on a graph	• Continuous 1-D distribution • Cell array • 1-D particles

Table 2.2: Characteristics of and appropriate target location uncertainty representations for general classes along the combined sparsity-connectivity spectrum.



(a) Open & regular (b) Irregular (c) Intersecting paths (d) Single path

Figure 2.4: Intuitive graphical depictions of significant general classes along the combined sparsity-connectivity spectrum. Shaded regions or solid line segments indicate possible target locations.

The significant practical impact of these properties appears when evaluating the quality of placing sensor footprints at given future locations. This has two components. The first is, roughly speaking, the number of possible target locations that must be considered to determine whether each will be in view and the quality of observation that would be received if seen there, which depends on properties such as the dimensionality of this location space, the number of samples needed to cover it (if using a discretized representation), and whether these evaluations may be coalesced or somehow collectively approximated (for instance by computing the result of a single cumulative probability distribution function). The second component is the number of places the target can move in an instant, which arises when computing predicted target motion or area recontamination. This too is a similar measure of the size of a space, which in this case is the space of directions it may move or possible short-term destinations. Intuitively, smaller is better, suggesting wise use of efficient representations when sparsity and low connectivity permit. This arises in formal terms in Section 3.3.3 and is the motivation of the proposed use of structure that enables low-dimensional representations for sparse, low-connectivity environments.

Finally, a number of other properties of the environment may affect the choice of approach, though not as severely, and are not a focus of this work. Beyond strict connectivity or binary location validity, relative traversability of regions of an area by a target, measured perhaps both as desirability and mobility of traversal, affecting the likelihood of a target moving into an area or a model of its motion within it. This is considered to some extent by existing methods for probabilistic search reviewed in Section 3.4.2 and is proposed in future work as a means to improve performance in road networks. Presence of occlusions to target observation such as tall buildings or trees might suggest the use of a higher-dimensional environment map representing directional visibility and is presently considered as an extension. Boundedness of the environment is another, as open areas from which the target might move infinitely

far require additional reasoning for containment or require treatment as search and tracking in a progressively enlarging area. Finally, environment scale can have a substantial impact on the approach strategy. In the extremes, a very small area on the order of the scale of the sensor footprint can be trivially observed nearly continuously with little effort, while an extremely large environment may preclude the use of long-term trajectory optimization and permit only very locally optimal solutions. One major goal of the proposed exploitation of structure is to represent sparsely-connected subspaces of an otherwise large environment to avoid this very problem, but the scale spectrum and appropriate thresholds for choosing certain methods over others is not otherwise considered in this thesis.

2.5 Motivation for Road-network Constraints

If an area of interest contains primarily roads, such as in an urban area or as a series of roads through otherwise untraversable terrain, the structure or topology of these roads is particularly informative for a search or pursuit task. The three key elements of such information include the restriction of a target's possible location to a small subset of the environment (assuming a target must lie on a road), the imposition of strong continuity (and hence short term predictability) on its motion during the typical case of motion along a road segment, and limitation of the branching factor in target paths to splitting at junctions (or, less probably, U-turns). These together designate road networks as a useful instance of a high-sparsity, low-connectivity environment.

Such constraints are immensely useful to UAV observers by directly easing many of the task challenges enumerated above. Sharp constraints as provided by roads greatly mitigate large uncertainty in individual observations, improved predictiveness as provided by limited target paths better survives frequent observation dropouts and slow returns to view, and fewer target location hypotheses and lower-dimensional uncertainty distributions reduces both the set of possible observations to evaluate for predictive planning and the computational and memory burden of running or branching filters based on hypothetical observations.

At the same time, while superficially representing a very small segment of the space of possible environments, road networks constitute a realistic and worthwhile case to consider. Primarily, as the developed proportion of land area worldwide continues to expand, any rescue, surveillance, perimeter protection, or suspect pursuit mission is likely to take place in or near a population center. Indeed, already by 2007 half the world's population lived in urban areas, and this is expected to reach 70% by 2050. [131] This presents an immediate practical demand. Additionally, the

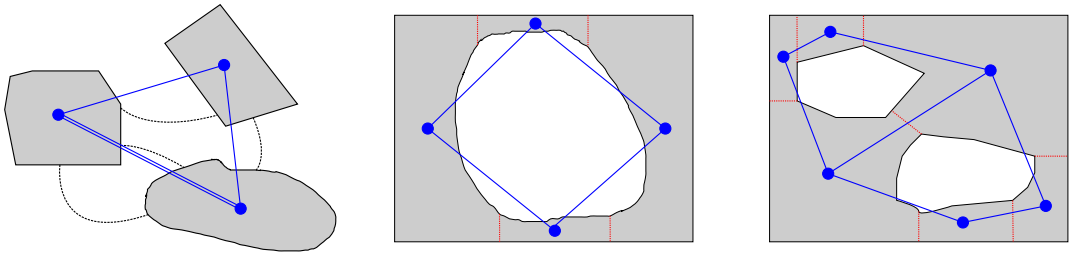
tracking and guidance community, noted in Section 4.3.3 as having extensively studied constrained and multi-modal geolocation on road networks using high-altitude radar, has long recognized the value of incorporating such terrain knowledge in heavily fielded systems and thus provides precedence. Finally, it is certainly the case that for many scenarios, operation over an equivalently-sized open area may simply be infeasible with a given number of vehicles and a given level of computation, so a practical solution to these strictly stretches the operating envelope of such systems.

2.6 Generalization to Mixed Environments

Though the clear focus in this thesis is environments containing road networks whose connectivity is representable as a graph, generalization to a broader class of environments is possible with some ease. One such means is to apply a hierarchical strategy in which pinch points or short boundaries between otherwise open regions are first identified, producing a set of disjoint regions with inter-connectivity defined by shared boundaries that may be blocked to prevent target passage between the two regions. This may at one level be treated as a graph, in which nodes are the open regions and edges are their boundaries. Then, in for instance the case of a search task, each region may be seen at a lower level as a small open area reasonably efficiently covered by any existing open-area search strategy (e.g. coverage or guaranteed sweeping). Several examples of such decompositions of increasing complexity are shown in Figure 2.5.

Using such a decomposition, the outer overlaid graph may, conceptually, be searched using any applicable means, with several alterations. These entail mapping the blocking of an edge to blocking an entire regional boundary, while clearing or guarding a node constitutes executing an appropriate open-area sweep within the region (if not already performed since the last opportunity for recontamination) and blocking all its entrances. Given sufficiently many UAVs, environments of considerable size and complexity may now be considered. Similarly, the case of recapture may be likewise extended, operating at the level of the virtual graph, though once a target has possibly reached a large open area, it must be searched, quickly growing in required resources to the number of vehicles required for a full-fledged search. The case of pursuit is more challenging, as the improvements presented and proposed in Section 4.3 rely on the presence of narrow corridors (e.g. between regions), and options for generalization remain an open point of consideration.

The proposed decomposition reduces a large class of environments that otherwise occupy a large area and which present an infeasible challenge down to a distinct set of simpler environments joined as a graph, permitting formal reasoning using



(a) (b) (c)

Figure 2.5: Three example environments to which the proposed decomposition is applied, as open areas connected by boundaries that may then be treated as nodes and edges, respectively, depicted as a superimposed graph in blue. As a further example, the applicability of a guaranteed search such as that described in Chapter 6 is considered, in which nodes are guarded and edges swept. The first environment (a) presents an ideal example in which several areas each requiring few searchers have only a few narrow (easily-guarded) interconnections, while the total area is large and would require many searchers if treated as an open area. The second (b) provides an alternative case of an otherwise large area containing an obstacle occupying most of the area. Here too, if treated as an open area, many searchers would be required, whereas it may instead be decomposed into several regions each requiring few searchers and connected by short boundaries. The last (c) points out that though decompositions of complex environments may be treated as a graph, it may not be advantageous to do so. In this case, the number of UAVs required to guard boundaries and perform regional searches exceeds that required to simply sweep the enclosing rectangle, as may be seen from the presence of two large open areas and many boundaries, the length of even several of which is approximately the width of the entire enclosing rectangle.

their connectivity. Clearly, for sufficiently densely connected environments such as those containing only sparsely distributed obstacles, this decomposition provides little benefit and may produce a graph requiring an unnecessarily large number of UAVs to search it. It is argued that by and large, such environments reach levels of dense internal connectivity that they are better considered as nearly completely open areas that simply remain a difficult problem.

2.7 Observer Modeling

UAVs differ from traditional ground-based or abstractly planar vehicles in a number of important ways. These are now enumerated, their models stated specifically for use in the remainder of this document, and their implications considered.

Sensing constraints

Whereas common models for a planar vehicle's sensor are generally of the form of a ray emitter with bounded range or bearing (if not omnidirectional), a UAV's field of view will generally form a sensor footprint on the ground that is a finite spot offset from the vehicle that is the result of intersecting similar ray emissions with the ground. The position and form of this spot may further be maneuver-dependent as UAV orientation changes, unless constant level flight is assumed for simplicity, and moving the spot requires potentially complex vehicle motions. This is further formalized in the following subsections and Section 3.3.1.

Additionally, given the long range to targets (50 to several hundred meters), limited onboard computation to process sensor readings, and the lossy nature of wireless transmission links, received raw sensor data will inevitably contain noise or be missing portions that cause likelihood of target detection given an observation substantially lower than for a ground vehicle that can get possibly arbitrarily close to a target, have large amounts of onboard computation, and need not wirelessly transmit sensor data for processing.

Motion constraints

All UAVs must maintain lift and are generally designed to operate within a region of their state space (appropriately called a flight envelope) within which an onboard autopilot is programmed to maintain this lift. For fixed-wing aircraft, one resulting constraint is that of a minimum speed that must be maintained, though with the positive effect that average speeds for fix-wing aircraft are much higher than those of a typical autonomous ground vehicle. For most aircraft, the flight envelope dictates a maximum turn rate corresponding to a minimum turning radius. Typically, there also exist dynamic constraints that couple acceleration or turn rate to vehicle attitude, including for instance the common need to bank (roll) to turn. This too is further formalized in a later subsection.

Terrain obstacles

While ground vehicles are generally incredibly vulnerable to obstacles–unable to see over those on the order of the vehicle height and unable to traverse obstacles above a small fraction of the vehicle height–UAVs, by virtue of their altitude, have in practice minor vulnerability to them. UAVs may pass over obstacles altitude-permitting, but it is much more critical that collisions be avoided than for ground vehicles, both because lightweight vehicle design implies great fragility and because the dynamic disturbance of a collision will

result in a fatal excursion from the flight envelope. Likewise, again by virtue of altitude, UAVs can see over obstacles as the geometry allows (taller obstacles casting a longer shadow) but not beneath them as a ground vehicle might otherwise be able.

Vehicle design

Field-reconnaissance UAVs considered here are typically limited by fuel or battery capacity to relatively short flight durations of one half to several hours, whereas a ground vehicle need not continuously oppose gravity to maintain this weight at altitude. This has led to models of dynamically sized teams or explicit refueling subtasks to account for the limited longevity of a single UAV, but this aspect is not considered here. More importantly, as previously introduced, weight, cost, and size limitations that are much more stringent than those applicable to ground vehicles, internal state sensing or proprioception is substantially worse than that available on ground vehicles, with resulting detriment to observations relying on this state. Finally, such UAVs cannot meaningfully interact with targets given little available payload and thrust, so missions by necessity must be indefinite, as a target may simply continue to move once initially detected.

Taken together, UAVs possess substantial differences from ground vehicles, but it may be noted that these produce largely complementary properties. This motivates the possible use of both types for demanding missions, an idea that was the very focus of initial experimentation described in Section 4.1 providing motivation for practical representational improvements. Much room for further exploration in this area remains, however.

UAV Sensor Model

A great variety of external sensors have been either used or suggested for use aboard UAVs. Many, including laser scanners and active radar arrays are beyond the payload capacity of field-reconnaissance UAVs, but many more including especially passive varieties remain feasible. Visible-spectrum cameras (so-called electro-optical sensors) are a staple given that the use of autonomy on such UAVs is in its infancy, and human operators are still nearly always present. Passive infrared cameras, direction-finding radios (optionally with a wide-area transmitter component to detect emergency rescue reflectors), and detectors with specialized spectral sensitivity (e.g. radiological emission sensing) have likewise seen use or proposal. All of these have the property that they can be modeled as reporting a detection as a direction vector or spherical

coordinate in vehicle-local coordinates, or as more commonly termed, a bearing-only sensor.

Here, visible-spectrum cameras (or any sensors having a sensing plane and an analogy for focal length) are assumed, but any sensor providing such a direction vector is equally relevant. A general observation at time t is defined as

$$\mathbf{z}_t = \begin{cases} \emptyset, & \text{no target detected} \\ \mathbf{x}_{\text{tgt}}^{\text{img}}, & \text{target detected} \end{cases}, \tag{2.1}$$

where $\mathbf{x}_{\text{tgt}}^{\text{img}}$ is the image pixel coordinate of (the center of) a detected target. According to the standard pinhole camera model, [129] this 2D image projection of a target is related to the target's 3D location in the reference frame of the camera via the 3x3 camera calibration matrix $\mathbf{M}$ by

$$\begin{bmatrix} \mathbf{x}_{\text{tgt}}^{\text{img}} \\ 1 \end{bmatrix} \equiv \mathbf{M}\mathbf{x}_{\text{tgt}}^{\text{cam}} \tag{2.2}$$

when expressed in homogeneous coordinates.

Inverting this relationship provides a ray parallel to the vector $\hat{\mathbf{v}}_{\text{tgt}}^{\text{cam}}$ in the camera reference frame given by

$$\hat{\mathbf{v}}_{\text{tgt}}^{\text{cam}} \equiv \mathbf{M}^{-1} \begin{bmatrix} \mathbf{x}_{\text{tgt}}^{\text{img}} \\ 1 \end{bmatrix}, \tag{2.3}$$

again expressed in homogeneous coordinates.

Finally, if $\mathbf{R}_{\text{UAV}}^{\text{world}}$ and $\mathbf{R}_{\text{cam}}^{\text{UAV}}$ are rotation matrices taking vectors from UAV to world and camera to UAV reference frames, respectively, then the target must lie along the ray in space given in world coordinates by

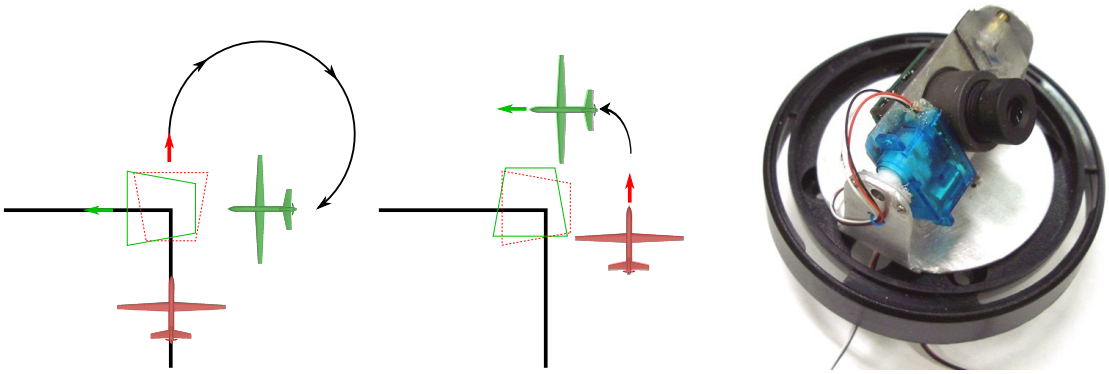
$$\hat{\mathbf{v}}_{\text{tgt}}^{\text{world}} = \mathbf{R}_{\text{UAV}}^{\text{world}} \mathbf{R}_{\text{cam}}^{\text{UAV}} \hat{\mathbf{v}}_{\text{tgt}}^{\text{cam}}. \tag{2.4}$$

Methods for transforming this ray into an estimate of target location–for instance by intersecting rays from multiple detections or intersecting a ray with a known terrain model on which a target must lie–are considered in Section 3.3.1.

UAV Sensor Pointing

Sensor mounting and pointing, generating the value of $\mathbf{R}_{\text{cam}}^{\text{UAV}}$ in the preceding equations, greatly influences the details of any search or tracking strategy. One of

three variants is commonly assumed throughout the literature: a fixed downward-angled/forward-pointing or directly downward-pointing camera, a fixed downward-angled/side-pointing camera, or an actively steered camera mounted on a powered gimbal. Together with the choice of motion model (specified next), this dictates the form of maneuvers such as loiter and sweep and has particular benefits as well as disadvantages.



(a) Forward-pointing (b) Side-pointing (c) Gimbal from Quigley et al. [97]

Figure 2.6: Three examples of common sensor orientations and consequences. A UAV equipped with a forward-pointing (a) sensor (whose footprint is shown as a trapezoid) and having a minimum turning radius attempting to follow a road must overfly it and re-align if the road makes too sharp a turn. The initial UAV pose is shown in red (and its footprint a dotted red trapezoid), and its final pose is shown in green. In contrast, a UAV equipped with a side-pointing (b) sensor can orbit in place to make such a turn, maintaining view of the road, regardless of the angle of the turn. A gimbaled sensor (c) allows re-pointing during maneuvers, but this requires an additional control input and has some delay. For simplicity and greatest generality, a fixed-orientation is typically assumed in this thesis, and given its relative advantages, the orientation is chosen as a side-pointing one.

Fixed forward- and downward-pointing cameras are most intuitive, as the pose and motion of the aircraft map directly to the path of the sensor footprint. To search an area, the UAV is moved as desired, and the footprint moves beneath it. Pursuit of a moving target implies (in its basic form) moving the UAV with the target. Loitering above a slow-moving or stationary target or guarding a location simply requires remaining above it. For rotorcraft, these tasks are straightforward. However, for fixed-wing aircraft having minimum speed and turning radius, these are more difficult. If a pursued target makes a sharp turn or drops its speed below the UAV's minimum, the UAV has no choice but to overfly the target (most likely losing view in the interim) while repositioning to resume pursuit. Loitering may be

implemented as a tight orbit or as some form of figure-8 or clover-leaf pattern if the sensor footprint is sufficient. Multiple vehicles spaced along the pattern may be required for total coverage. A major advantage of such a mounting is that as long as a target is visible from overhead, occlusion by nearby terrain obstacles is avoided. This is particularly useful in urban canyons which may have tall buildings on either side but possibly long stretches of open road between them.

The common alternative specifically intended to avoid difficulty in loitering is a side-pointing camera. Without adding any particular disadvantage for rotorcraft, this allows a fixed-wing design to provide persistent surveillance of a target by merely orbiting it. Pursuit is implemented by placing the UAV at a lateral standoff to a target. Conveniently, at sufficiently slow speed, arbitrarily-shaped sensor footprint trajectories may be followed: typically the vehicle may simply move along the required path with this standoff, and where too tight a turn is required, an orbit around the current footprint location may be followed until again aligned with the desired trajectory. For slow and stationary targets, this provides adequate pursuit. However, for fast-moving targets making sharp turns, loss of view is still inevitable. An additional downside is increased vulnerability to environmental occlusion. In the case of urban areas, tall buildings may prevent viewing a target from certain directions.

Finally, gimbal-mounted cameras may be aimed arbitrarily, though via an additional axis of control that may itself have constraints and dynamics (e.g. angle limits, delay, and inertia). This allows, for instance, the vehicle to have the benefits of either a forward- or side-pointing camera at any time and switch between them as desired. Clearly, this increases the effective field of view and can help to reduce target loss during pursuit. Chief disadvantages are fragility, weight (reducing mission duration), and the additional control complexity of having to produce a planned trajectory for it in addition to the vehicle. Examples of existing work in gimbal control are provided in Section 3.2.

Though gimbals are now very common on even small field reconnaissance UAVs, this thesis primarily assumes the presence of a fixed side-pointing camera. This is done for several reasons. The presence of a gimbal introduces additional complexities such as positioning limits and dynamical constraints of its own, an attempt is made to be applicable to a wider range of vehicles including those without one, an emphasis on required vehicle motions is desired instead, and because in many cases integrating a gimbal into a motion planner is straightforward. Meanwhile, a fixed side-pointing camera is generally preferable to other fixed pointing orientations and is available on all AeroVironment Raven UAVs, the choice of typical widely-fielded vehicle targeted by this thesis.

The choice to assume the presence of a side-pointing camera should not be taken

to represent a limitation in the applicability of the ideas in this thesis, however. At a minimum, it represents a lowest common level of capability implementable on either fixed-wing aircraft or rotorcraft, whether or not a gimbal is actually present (as it may always be steered to a constant angle). The addition of a gimbal will only improve the performance of any strategies presented herein, by simply adding flexibility in operation, reducing vulnerability to occlusion, reducing target loss frequency (increasing time in view), and improving target geolocation accuracy. The primary weakness of this work in this area is that, as explicitly stated in Section 3.2, no particular effort to perform any reasoned actions with a gimbal beyond attempting to maintain view are made. However, several avenues for gimbal exploitation are suggested in Section 7.1, among which is that gimbal control inputs may be treated as an additional control axis for motion planning without any substantial modification to an existing planner.

UAV Motion Model

For simplicity, details of aircraft dynamics are not considered here. It is assumed, instead, that there exists a function computing a forward model of each UAV's dynamics so that if $\mathbf{x}_{\text{UAV}}(t)$ represents the complete state of the UAV at time t, then the predicted future state $\mathbf{x}_{\text{UAV}}(t')$ may be computed as

$$\mathbf{x}_{\text{UAV}}(t') = f_{\text{motion}}(\mathbf{x}_{\text{UAV}}(t), \mathbf{u}, t' - t), \tag{2.5}$$

where $\mathbf{u}$ represents a control input vector (such as a turn or thrust command) to be held for the duration $t' - t$.

Where it is necessary to consider an explicit model and for algorithmic comparison, a simple form of a discrete time kinematic model of a so-called Dubins car [34] is used as f_{motion}. The discrete equations of motion are given by

$$\begin{aligned} x_{k+1} &= x_k + V\Delta t \cos\theta_k \\ y_{k+1} &= y_k + V\Delta t \sin\theta_k \\ \theta_{k+1} &= \theta_k + \omega_k \Delta t \\ & |\omega_k| \leq \omega_{max}, \end{aligned} \tag{2.6}$$

where $\mathbf{x}_{\text{UAV}} = [x_k, y_k, \theta_k]^T$ is the position and heading of a UAV in its motion plane (assuming constant altitude). A constant (or slowly varying) forward speed V is assumed, and the single control input is a steering or turn rate ω_k having bounded magnitude ω_{max} that is held for the entire duration of a timestep Δt. This is a common model in the literature on UAV motion planning [124, 45, 86] that reasonably approximates fixed-wing aircraft and helicopters flying at fixed altitude.

As an additional realism improvement, a rough model of the need to bank to perform a turn is incorporated by coupling turning rate to roll angle so that turning at a given rate produces a roll angle resulting in a so-called coordinated turn, defined by

$$\phi = \text{atan}\left(\frac{V\omega}{g}\right), \tag{2.7}$$

where g is the gravitational constant (approximately 9.81 m/s^2).

Implications for Ground Vehicle Strategies

These distinctions from ground-based or abstract planar vehicles imply a need to treat UAVs somewhat differently, preventing the simple invocation of a search or tracking algorithm not designed explicitly for them for several reasons. First, the differing motion constraints from most ground vehicles renders infeasible trajectories that may call for reverse motion, rapid turns, or (for fixed-wing aircraft) low speeds. Further, trivial mappings of abstract trajectories to feasible UAV motions–such as substituting a slow orbital turn in place of a zero-radius turn–may result in differing-cost paths under a given objective function than expected and will produce sensor footprint trajectories with excursions from the intended path, undermining for instance a continuous sweep. Additionally, an algorithm assuming that an observer must lie in the same plane as a target will fail to take advantage of the ability of UAVs to fly above and see beyond obstacles, therefore providing paths that are necessarily sub-optimal. Lastly, the impracticality of reliably conveying large amounts of data over UAVs' lossy long-range links and hence need for efficient or distributed representations is easy overlooked in abstractions, though neither is this issue a focus of this thesis.

Most existing work in UAV search and tracking, summarized in the following chapter, has therefore focused on specialized or novel control strategies, approaching from the problem from an aerospace or estimation rather than abstract search or pursuit-evasion perspective. This thesis proposes an instance of that alternative perspective to attain superior performance where environment structure permits.

CHAPTER 3

Related Strategies for Search and Tracking

As a robust topics of study, the disparate areas of UAV search and tracking have accumulated a large body of literature independently considering techniques for geolocation, control for pursuit, and search. This chapter considers this work in the context of the general CSAT task and presents a summary of existing approaches to each component. Section 3.1 first enumerates common abstractions of which the problem might be considered an instance and thereby have known properties or possibly available (albeit not necessarily immediately applicable) solutions. Related aspects of the problem that must be addressed in any practical implementation but not considered further here such as data association and aerodynamics are then briefly summarized in Section 3.2. Returning to relevant issues, classical and existing approaches to producing geolocation observations, filtering these to best produce combined estimates in the face of potentially large error, and then using these estimates or raw observations to pursue targets with heuristic or optimized trajectories are then summarized in Section 3.3. These and more case-specific trajectory generation schemes for reaching and observing targets whose position is not well known—the search case—is next presented within a taxonomy of scenarios in Section 3.4. Finally, independent study of the recapture case for targets having large but clustered location uncertainty is advocated in Section 3.5, with several intuitive and potentially applicable existing strategies considered.

3.1 Common Abstractions for Search and Pursuit

Search and pursuit has been formally studied for many decades in a number of contexts, from the completely abstract to those with immediate mobile robotics or field applications. Traditionally, the problems of *search* and *pursuit-evasion*[1] have been treated quite independently. Search is commonly defined as the case in which a team of searchers wishes to clear an area of any targets that may be present, without necessarily knowing whether one is even present (thus also earning the name "one-sided search"). As a result, solutions often take the form of an offline precomputation. Pursuit-evasion typically takes the form of attempting to reach, encircle, or maintain coverage of a known target that may be trying to avoid capture and whose position is usually assumed to be known. As a pursuer's motion depends on a target's choice of motions, reactive online computations are generally required.

Separately but often tightly coupled to a given problem formulation, two types of targets are considered: adversarial and probabilistic. Adversarial targets actively attempt to evade detection or capture and may be granted infinite speed or complete omniscience of pursuer location, strategy, and intent. A strategy that is guaranteed to detect or capture an infinitely powerful target offers the additional desirable properties that no other motion model of the target is needed and that the temporal dimension is effectively removed (eliminating concern for physical scale of the environment or distances between points), however it may be extremely conservative both in resource requirements (such as number of agents) and the number of action steps or amount of time required to guarantee success. Probabilistic targets, on the other hand, are assumed to move with some stochastic model usually independent of searcher or pursuer motion, corresponding to a target that is oblivious of or indifferent to the presence of its pursuers. Strategies for detecting or capturing probabilistic targets, then, seek to perform optimally in expectation with respect to some metric such as minimum expected time to detection or capture, maximum probability of either within a fixed time interval, or maximum expected number of observations within an interval.

The search problem may be well divided between probabilistic and adversarial targets. Search for probabilistic targets formed the basis of the so-called theory of search in the operations research community, which has a long history dating back to naval warfare strategies of World War II and later formalized in the seminal works of Koopman. [74] Roughly, such a search seeks to maximize the probability of target

[1]The terms *search* and *pursuit-evasion* are used nearly interchangeably in some literature or with varying definitions between fields, to much confusion. Here, an attempt is made to simply provide a common, reasonable, and internally consistent definition.

detection given a prior distribution of possible target location using a fixed amount of available search effort and target detection likelihood per effort, producing a best search density schedule indicating where to apply given amounts of effort. This has over time grown to include more advanced models of target detection, variations in target motion, and more computationally complex solutions given advances in computing. Well-regarded surveys include those by Stone [119] and later by Benkoski. [15]

Search for adversarial targets, meanwhile, has been considered both for bounded continuous spaces and discrete environments whose connectivity between regions is represented as a graph. Adversarial search in continuous polygonal environments was most famously studied by Guibas et al., [56] providing an algorithm to clear a planar polygon possibly containing an arbitrarily fast target using a minimal team of searchers having omnidirectional unlimited-range sensors that cannot see through walls and showing the NP-hardness of the problem. Among other extensions, this has been generalized to searchers having limited-angle field of view [48] and bounded-speed targets [128] for improved realism and practicality in mobile robotics. For environments better represented as a graph of discrete locations, the game first defined by Parsons [92] and independently by Petrov [94] is that of locating a missing explorer in tunnels of a cave system by a minimal team of searchers, regardless of the motions of the explorer. This is the classical formulation of the problem known as *graph search* that typically assumes a completely omniscient adversary that is sometimes equivalently treated as a diffusing poisonous gas that must be corralled. The minimal searcher team size required for a given graph is called its *search number*, the determination of which (and therefore an optimal search schedule as well) has been shown to be NP-complete. [78] Though many problem variations exist, the class dichotomy considered here is that of a target that may lie only on an edge (requiring *edge search*)–the original and more commonly studied version–versus that of a target occupying only nodes (requiring *node search*). The case of node search is generally more applicable to practical robotics problems and has received attention in this context from Hollinger and Kehagias et al. [61, 70] for indoor search and from Kolling and Carpin [73] in a hybrid framework called GRAPH-CLEAR in which nodes represent areas to be cleared and edges are weighted by the number of searchers required to guard them. Edge search, however, may be more applicable to outdoor scenarios and forms the basis of guaranteed road network search presented in Chapter 6. Finally, traditional formulations of graph search typically permit searchers to instantaneously move between any two points in the graph (producing so-called teleporting or jump search), which is quite unrealistic for any physical application and is eliminated by constraining a search schedule to be an *internal search* in which searchers are themselves constrained to the graph topology as well. A stronger notion

introduced by Barrière is that of *connected search* in which searchers must lie within a growing connected cleared subgraph, conceptually maintaining a safe region for supply lines from a start point. [13] A thorough survey of existing work in canonical forms of guaranteed graph search was performed by Fomin and Thilikos. [41]

Pursuit-evasion has likewise been considered in several separate continuous and discrete-space formulations. In continuous space, a pursuit-evasion problem is often treated as an instance of one of several simple games. One is the so-called *lion-and-man* game in which a lion (pursuer) attempts to reach the location of an evader having the same maximum speed within a closed geometric environment such as a circle or polygon. More recent research has considered unbounded or higher-dimensional areas and sensing limitations, and many results have over time been shown for variations of the problem. Several interesting ones include the facts that a single lion can eventually capture an evader in any simply-connected polygon, [66] that three can do so in any polygon, [16] and that $d + 1$ lions can capture an evader in the open space $\mathbb{R}^d$ if and only if its initial location lies within their convex hull. [75] Somewhat of a generalization of the lion-and-man game is the so-called *princess-and-monster* game in which (in its original formulation [65]) a monster (pursuer) attempts to reach an evader of arbitrary speed in a closed environment in which neither player can see the other, providing extension to local-only sensing and fast targets. This is an instance of a more general class of differential games, of which the *homicidal chauffeur* problem [64] is more famous. In that problem, a pedestrian (evader) that can move arbitrarily at slow speed seeks to avoid an automobile (pursuer) that is much faster but is less maneuverable due to motion constraints (such as automobile or aircraft dynamics). An excellent summary of its history and known properties was compiled by Patsko and Turova. [93]

In discrete space, pursuit-evasion problems are also frequently represented as graphs. Given further discretization in time such that pursuer and evader move in alternating discrete rounds, the problem is now a discrete game commonly called *cops and robbers*. In typical formulations, pursuers attempt to reach the location of (the node containing) the evader, and both players receive total information as to the location of the other. In this game, the analogue to the search number is the *cop number*, denoting the number of pursuers required to guarantee capture in finite time for a given graph. The set of graphs requiring only one pursuer was characterized by Quilliot, [98] but it is EXPTIME-complete to determine whether k cops are sufficient for a given graph and initial conditions. [53] Full knowledge of target location may be realistic if its position is provided by a sensor network or by an aerial observer directing ground agents, but otherwise is impractical for most robotics applications. Adler et al. [2] consider the variant termed *hunter-and-rabbit* in which neither player

knows the other's location or motion decisions (which is thus essentially the discrete form of the princess-and-monster game), for which only randomized pursuit strategies are shown to be useful and have tight asymptotic bounds on expected capture time.

For each of these problem formulations, many variations exist, relaxing or tightening assumptions on or capabilities of both the pursuers and evaders. Small changes of any form often change the problem structure substantially, and known results vary considerably. Though tantalizing similarities to practical physical tasks are present, remarkably little use of formal search and pursuit-evasion theory has been made in applied robotics literature. Chung et al. [25] present a review of several fundamental results and recent applications in mobile robotics. The immediate applicability of any given abstraction to the aerial search and tracking task is particularly unclear, especially because most abstractions model pursuers as restricted to the same space as the target and (for visibility-based search) as having a point sensor aimed outwardly within the planar environment. Two existing examples demonstrating some applicability include the use of discrete probabilistic search for a randomly moving evader by an air-ground team providing guaranteed finite capture time [133] and the use of continuous differential game theory to provide multi-UAV interception of a target whose current position but not future motions are known. [103] The prior provides no guarantees for adversarial targets or performance bounds for purposefully-moving targets, and the latter relies on the unrealistic assumption of precisely known target location while further lacking physical implementation. Several other instances of probabilistic search for non-adversarial targets without reference to formal search or pursuit-evasion and providing only locally optimal behavior are described in succeeding sections. A major motivating goal of this thesis is the development of mappings to graph-based abstractions permitting formal reasoning for all three cases of CSAT.

3.2 Related Aspects not Considered

Many aspects of a practical implementation of or topics relevant to the search and tracking problem have been well studied previously and are neglected in this thesis, as much of the research into the design, control, and applications of UAVs comprised the bulk of original forays in the area as robotics or aerospace problems. Here, they are instead assumed to have been addressed as needed or remain fundamental challenges. Several of the most prominent include:

Self-localization and environment mapping

Accuracy of available vehicle state may be quite poor given the low-end sensors onboard field-reconnaissance UAVs, and there exists a sizeable body of work on attempting to reduce this error by augmenting these sensors with visual localization using an onboard camera likely already present for surveillance. Full simultaneous localization and mapping (SLAM) systems using Extended Kalman Filters (EKFs) were exhibited by Bryson and Sukkarieh [20] and Caballero et al. [21], in which the prior formulate their filter with bearing-only observations of tracked landmarks and the latter with 3D observations formed by projecting tracked landmarks and intersecting with a known ground plane. Madison et al. [82] demonstrate tracking multiple ground features over several tens of seconds to constrain the standard Structure-from-Motion 8-point algorithm, which solves for all the feature locations and vehicle poses. As an alternative to explicit landmark tracking, Andersen and Taylor [7] show that with a planar ground assumption, a homography-based visual odometry algorithm can be combined with GPS/INS sensors using an Unscented Kalman Filter (UKF) to improve the vehicle pose estimate. A major downside to most of these strategies is a need for stable visual tracking of landmarks over hundreds of video frames, which is difficult given low-resolution video of potentially featureless terrain from a fast-moving, wind-buffeted platform. Matching of live video to existing satellite imagery has been proposed by Conte and Doherty [28] and Sim et al. [114], but at significant computational cost and vulnerability to changes in terrain appearance. As any such strategy may leave substantial residual error, target geolocation with uncertain vehicle state is a related topic discussed in Section 3.3.2 and improved upon in Section 4.2.

Automatic target recognition or data association

Detection of a target, tracking it efficiently while in view, and distinguishing it from any other observable targets are themselves challenging problems, particularly using noisy sensor data. When using a camera, this is largely a computer vision problem, and has received much attention given its applicability to a huge range of applications. Automatic detection from a stored model or target classifier, automatic detection of moving targets, and operator-initiated or manual tracking are all means that have been previously proposed. Section 4.1 discusses several and describes the operator-initiated automatic visual tracking system used to motivate and evaluate proposed algorithmic improvements in field experiments. It is otherwise assumed that by some means, observations are received when a target is detected or tracked, and a negative observation received to indicate the lack of such. Section 2.7 defines

this more precisely.

Aerodynamics and low-level vehicle control

The development and evaluation of small, low-cost control systems for SUAVs and MAVs has been a popular topic of study given the many constraints the design of such UAVs impose. As this is a complex and largely well-understood problem of its own, with many such UAVs now incorporating sophisticated controllers, the realistic assumption is made that an autopilot is present that provides high-level action primitives such as waypoints, orbit or hover points, or steering-like commands and which further deduces and cancels out effects of wind to the extent possible. This too is defined more precisely in Section 2.7.

Sensor aim or gimbal state planning

An onboard target-observing sensor such as a camera may have a means to alter its relative pointing orientation so that the sensor footprint on the ground may move without requiring motion of the UAV, for instance by mounting on an actuated gimbal. Though many field-reconnaissance UAVs may be equipped with a gimbal, it is explained in Section 2.7 that a fixed, side-pointing sensor will typically be assumed without loss of generality of the ideas presented. It is of course acknowledged, however, that increasingly many UAVs possess a gimbal given ever-improving reliability and implementation cost and that this remains an important issue in practice. As a starting point, existing work in online continuous gimbal control may be built upon, including for instance workload-reducing camera-centric teleoperation by Quigley et al. [97] and work by Skoglar that is both foundational [116] and treats gimbal control as a sensor management task. [115]

Obstacle avoidance and agent deconfliction

Though UAVs may fly over obstacles given sufficient altitude, it is typically critical that they avoid them, as collisions are unlikely to be as survivable as they may be for ground vehicles given greater airframe fragility and the need to maintain lift. An interesting problem that has received some study is the safe navigation of urban or mountainous terrain using low-altitude UAVs, however for simplicity the assumption will be made that any terrain obstacles will lie strictly below a fixed flight altitude. To some extent, this reduces the problem to a two-dimensional problem, though constraints such as the need to bank to turn and the altitude-dependent geometry

of observations are maintained, covered further in Sections 2.7 and 3.3.1. Further, to simplify planning, it is assumed that even coincident flight paths will never result in collision, which may be enforced by the common practice of specifying a different altitude for each aircraft with a safe buffer distance between each.

Target occlusion

In any real-world mission, there may exist terrain obstacles that prevent a UAV from viewing a target. These may take the form of overhead cover (under which a target may simply hide) or tall nearby structures that prevent viewing from certain directions. The issue of occlusion is not directly addressed in this thesis, though opportunities for its incorporation are highlighted at many points and as an important area for future work. Given this importance, occlusion, particularly in urban environments as is the focus here, has received substantial prior study by others in various contexts. Semsch et al. [113] evaluate two algorithms for continuous coverage of urban areas taking into account occlusion to minimize visitation intervals. Geyer [49] integrates an occlusion-aware visibility model into an efficient probabilistic search strategy to minimize trajectory segments providing no useful observations. Kim and Kim [71] propose optimized spacing of UAVs around an orbit of a pursued target within an urban environment to minimize occlusion effects. Within road networks specifically, Kirubarajan et al. [72] and Ulmke and Koch [130] (among several others) build occlusion modeling into their multi-hypothesis target geolocation estimators further described in Section 4.3.3.

Distributed or decentralized estimation and control

Given possibly large communication distances, limited bandwidth, feeble onboard computation and storage, and the nontrivial likelihood of losing contact with any one vehicle, it is desirable that practical implementations use some means to minimize the amount of data transferred and avoid single points of failure. This has been addressed by a large body of work on distributed estimation and decentralized control with much success. While such issues are not a focus of this work, concise representations of possible target location are sought that require little data to be transmitted or which are easily factorized for agent assignment *to enable* such strategies to be more easily applied. This also serves the vital purpose of greatly improving computational tractability, needed both in physical implementation and the example results considered. Levels of coordination and decentralization are touched on further in Section 3.3.3.

3.3 Tracking and Pursuit with UAVs

Given a current or recent detection of a target, the task of pursuing it is comprised of two components: processing the observations to produce the best geolocation estimate possible and choosing a near-term trajectory that aims to maintain or improve this estimate, typically implying the further need to keep the target in view. Here, this is split into a pipeline of three stages consisting of generating geolocation observations from target detections, filtering these observations and predicting target motion in their absence to produce a best target location estimate, and choosing flight paths most likely to maintain a useful stream of future observations or paths that specifically optimize expected observations.

3.3.1 General approaches to geolocation

Once a target is detected using an onboard bearing-only sensor such as a camera modeled in Section 2.7, two broad approaches for mapping a pixel coordinate to a ground location are found in the literature. One is to register live UAV video frames to previously geo-referenced satellite or aerial imagery of the area, [29] which immediately provides the world location for any pixel of the image without any dependence on potentially-erroneous UAV pose reported by telemetry. However, this approach requires such prior imagery, accurate image registration may not be possible if the terrain is highly repetitive or homogeneous, registration may fail entirely if significant changes have occurred in the area since the prior imagery was acquired (as could likely be the case with smoke and rubble when responding to a disaster or conflict), and the computational and memory burden necessary to store and manipulate large maps may be infeasible. More common, therefore, is a second approach that uses an estimate of the UAV's pose derived from on-board sensors and the identified pixel coordinate to generate a ray in world coordinates as in Equation 2.4 that is then intersected with previous observations or a prior model of the terrain to produce an observation of target location. Using this approach, a target's position in world coordinates $x_{\text{tgt}}^{\text{world}}$ must satisfy

$$\mathbf{x}_{\text{tgt}}^{\text{world}} = \mathbf{T}_{\text{UAV}}^{\text{world}} \mathbf{p}_{\text{cam}}^{\text{UAV}} + d\hat{\mathbf{v}}_{\text{tgt}}^{\text{world}}, d \in \mathbb{R}, \tag{3.1}$$

where, corresponding to Equation 2.4, $\mathbf{T}_{\text{UAV}}^{\text{world}}$ is the pose transformation taking coordinates from UAV to world frames, $\mathbf{p}_{\text{cam}}^{\text{UAV}}$ is the calibrated location of the camera with respect to the vehicle body, and d is the distance to the target along the ray $\hat{\mathbf{v}}_{\text{tgt}}^{\text{world}}$ emanating from the camera's center.

Examples intersecting rays from multiple observations to produce geolocation estimates include those by Madison et al. [82] and Quigley et al., [97] in which a best near-intersection (as errors in pixel localization and vehicle state will prevent exact intersection) of all rays is taken to be the target's location. This has the advantage that no dependence on any prior terrain imagery or model is required (and, indeed, targets need not be restricted to lying on the ground), but many observations may be needed before a meaningful target location is available, and applicability to moving targets is not completely straightforward. In contrast, the ray corresponding to a single target detection may be intersected with a prior model of terrain in the form of a 3D height-map or simpler Digital Terrain Elevation Database (DTED) available for much of the planet from satellite surveys. [50] As with matching to geo-referenced imagery, this provides an estimate of target location from a single detection but can operate with rapidly-changing or partially-obscured terrain, at the cost of a very strong dependence on accurate UAV state. For the case of locally flat terrain, the distance d to a target needed to compute its location using Equation 3.1 is simply

$$d = \frac{z_{\text{terrain}} - z_{\mathbf{p}_{\text{cam}}^{\text{world}}}}{z_{\hat{\mathbf{v}}_{\text{tgt}}^{\text{world}}}}, \tag{3.2}$$

where z_{terrain} is the local terrain height. This may be extended to account for jagged terrain and man-made obstacles through proper geometric intersection with a model.

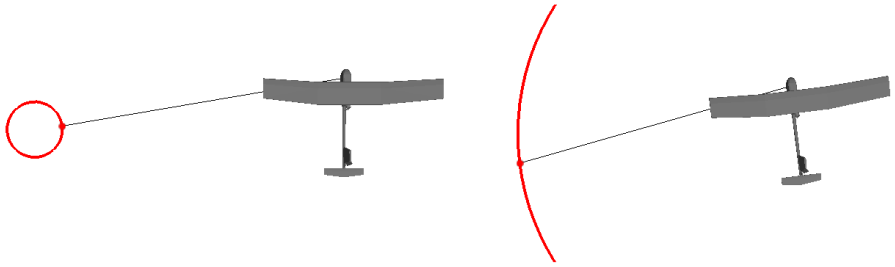
Whatever the means used to produce an individual target observation, it is convenient to define a function in its place

$$\mathbf{x}_{\text{tgt}}^{\text{world}} = f_{\text{obs}}(\mathbf{s}_{\text{UAV}}, \mathbf{x}_{\text{tgt}}^{\text{img}}), \tag{3.3}$$

where $\mathbf{x}_{\text{tgt}}^{\text{img}}$ is the pixel coordinate of the center of the visible detected target and $\mathbf{s}_{\text{UAV}}$ represents a state vector containing all UAV state such as position, orientation, camera calibration parameters, etc. that may be necessary to produce the individual geolocation observation. This permits abstraction of any underlying details. In the remainder of this document, the output of f_{obs} is referred to as a *pseudo-observation* since observations are actually received as image pixel locations.

Different axes of target state contribute differently to error in geolocation observations, two of which are depicted in Figure 3.1. Under the ray intersection model, error in UAV (usually GPS-provided) or the camera's relative position contribute additively to geolocation error and typically produce errors at the level of up to several meters, while UAV and relative camera orientation has a trigonometric impact that may be very large as distances to targets may be on the order of several hundred meters. Worse, as IMUs onboard field reconnaissance UAVs were previously noted to have relatively low-bandwidth and low-accuracy, these are the most

poorly sensed of UAV state dimensions. Consider the following example. Suppose a UAV lies 100m above and parallel to flat horizontal terrain and has a camera pitched downward by 30° centered on a target. The range to the target will then be 200m and the distance along the ground approximately 173m. If uncertainty in vehicle heading is modeled as a Gaussian distribution with $\sigma = 5°$, then with a rough heading error bound of $3\sigma = 15°$, uncertainty in resulting observations is bounded roughly by $173 \sin(15°) \approx 27$m, possibly encompassing quite a large area. For this reason, substantial effort in UAV geolocation literature has focused on filtering single observations to produce more accurate target location estimates.



(a) UAV Position Error (b) UAV Heading Error

Figure 3.1: Graphical depictions of the impact of error in differing UAV state dimensions on an individual target geolocation observation using ray intersection with terrain. Here, a UAV using a side-mounted camera is geolocating a target on level terrain. In (a), if error in UAV position is introduced so as to trace a circle in the plane, an identically-scaled circle appears in geolocation estimates, indicating additive effect on error. If, however, error in UAV heading is introduced as in (b), geolocation error is greatly amplified and scales linearly with distance to the target, tracing out an arc and indicating large non-linear impact.

3.3.2 Common geolocation filtering strategies

For stationary targets, direct averaging of observations produced by f_{obs} is an obvious first means of filtering, as it eliminates zero-mean noise in observations and provides some robustness to small fixed biases in UAV state estimates (such as in heading) that produce symmetric error that can be eliminated by averaging, for instance, over observations taken around an orbit of a target. Few error sources are symmetric in this way, let alone fixed, however. Redding et al. [101] perform essentially this, using linearization of the image projection function and recursive least-squares to efficiently compute a best target location while treating image projection as an uninvertible operation (conceptually permitting geolocation without terrain knowledge). Barber et

al. [12] improve upon this by wrapping least-squares filtering with online optimization to solve for fixed biases in sensor yaw and pitch that minimize variance in (or equivalently maximize clustering of) individual observations. Overcoming large or varying offsets, however, remains challenging and was addressed in Section 4.2 using an evidence-grid filter that greatly improves estimates in the presence of large, time-varying error.

Kalman filters provide an alternative strategy for geolocation filtering that permits explicit probabilistic modeling of uncertainty in particular UAV state dimensions and target motion. For these reasons and their instrinsic simplicity, they have received frequent mention (usually as EKFs) in the geolocation literature. [67, 96, 104, 135] Formulations typically belong to one of two varieties. One form treats observations as (correctly) lying in the space of pixel coordinates and explicitly includes the projection of target world to image location in the filter's observation function, while the other operates completely in world coordinates and receives pseudo-observations produced by f_{obs}. The latter is preferred here where possible as it permits the use of very simple filter models (typically, using the identity observation function) and encapsulates all aspects of observation generation within a single module. Either form requires an observation error propagation model to transform uncertainty in UAV state to uncertainty in observations, and therein lies the weakness of these filters. For the second model using pseudo-observations, standard Jacobian linearization of f_{obs} may be used to approximately convert Gaussian uncertainty in UAV state $\mathbf{s}_{\text{UAV}}$ having covariance $\mathbf{\Sigma}_{\text{UAV}}$ to Gaussian uncertainty in observed target location having covariance

$$\mathbf{\Sigma}_{\text{obs}} = \mathbf{J}_{f_{\text{obs}}} \mathbf{\Sigma}_{\text{UAV}} \mathbf{J}_{f_{\text{obs}}}^{T}, \tag{3.4}$$

where

$$\mathbf{J} = \frac{\partial f_{\text{obs}}}{\partial \mathbf{s}_{\text{UAV}}}. \tag{3.5}$$

Meanwhile, EKFs operating in the space of pixel coordinates implicitly perform identical linearization on the opposite, forward-projection, function. However, because the function representing either direction is highly nonlinear in a number of its parameters, particularly those pertaining to UAV orientation, Jacobian linearization may present a very poor approximation. For this reason, variations of UKFs or sigma-point Kalman filters have also been applied, with improved results. [22, 67] However, several problems remain. First, these too impose Gaussian uncertainty distributions on state dimension (often out of convenience), yet uniform or bounded uncertainty may be more appropriate for large errors or biases that may vary. Worse,

covariances are often chosen entirely arbitrarily (and highly optimistically), producing filters that are extremely likely to enter an inconsistent state with overconfident, incorrect estimates. True uncertainty distributions resulting from typical state uncertainties are further explored in Section 4.2 along with filters using uncertainty having parametrically-representable bounded uncertainty that are suggested as one possible means of improvement.

Sampling, Monte-Carlo, or particle filter approaches permit approximate representation of arbitrary posterior target uncertainty distributions, and for this reason have also been considered in aerial target geolocation. [105] Here too, two forms may be considered. The first is the classical particle filter implementation, [9] in which given an observation $\mathbf{z}_t$, each particle $\mathbf{x}_i$ representing a possible target location may be projected to image coordinates using some means such as Jacobian linearization or the Unscented Transform to produce an uncertainty distribution in image coordinates and receives a weighting depending upon $p(\mathbf{z}_t|\mathbf{x}_i)$ under this distribution. Unfortunately, this requires a projection and uncertainty propagation per particle, which may be infeasible on low-end embedded hardware or where hypothetical future observations must be repetitively considered. A slightly differing approximation making use of pseudo-observations instead weights each particle under $p(\mathbf{x}_i|\mathbf{z}_{\text{pseudo}_t})$, requiring only a single projection and error propagation per observation update, greatly improving computational performance. Finally, discretized environments represented as grids may conceptually be treated as collections of particles as well, in which particles do not themselves move during process model execution but rather gain or lose probability, but otherwise receive identical Bayesian observation updates. [126]

Where a target is known to lie in an environment having discrete terrain types with differing characteristics, such as when a target is present on or near a road, constrained or multi-model geolocation provides a potential means to improve accuracy by making use of a most-specific target model offering the most accurate prediction at all times. Extensive use of such filters has been made in geolocation from high-altitude radar as mentioned in Section 2.3 on large vehicles such as those shown in Figure 2.3 to both automatically classify targets as on-road or off-road and improve geolocation accuracy when they are on-road by reducing the set of possible target locations to those lying on the road. Examples include the use of multiple-model Kalman filters [72] and variations of constrained particle filters. [37] Both have shown marked increases in estimate accuracy in this context, and for this reason and their representational compactness that they are proposed for use in road-constrained pursuit by field-reconnaissance UAVs. Present uses, however, differ substantially in that the purpose is to better identify road-traversing targets and improve geolocation estimates using already very accurate sensors. In such larger vehicle contexts, online

pursuit is also nearly never considered, as given the omnidirectional nature of such radars and the typically huge distances to targets, even substantial motions on the part of the sensing aircraft can have negligible effect on the geometry or availability of observations. Further, missions for these aircraft are often pre-planned with little infrastructure in place to direct their paths based on the information needs of remote teams. This is in immense contrast to the smaller UAVs considered here that *must* continuously move to maintain view given small, finite sensor footprints providing low-accuracy observations and whose primary mission is active reconnaissance.

3.3.3 Control for pursuit

To maintain view of a target that was recently observed and to continue to reduce uncertainty in its location, flight paths that accomplish this must be chosen. This is of particular necessity for fixed-wing aircraft in constant motion for which turn decisions must be continuously made and when pursuing moving targets. Often closely coupled with geolocation schemes, a number of strategies for accomplishing this have been previously proposed and may be roughly separated in to two categories.

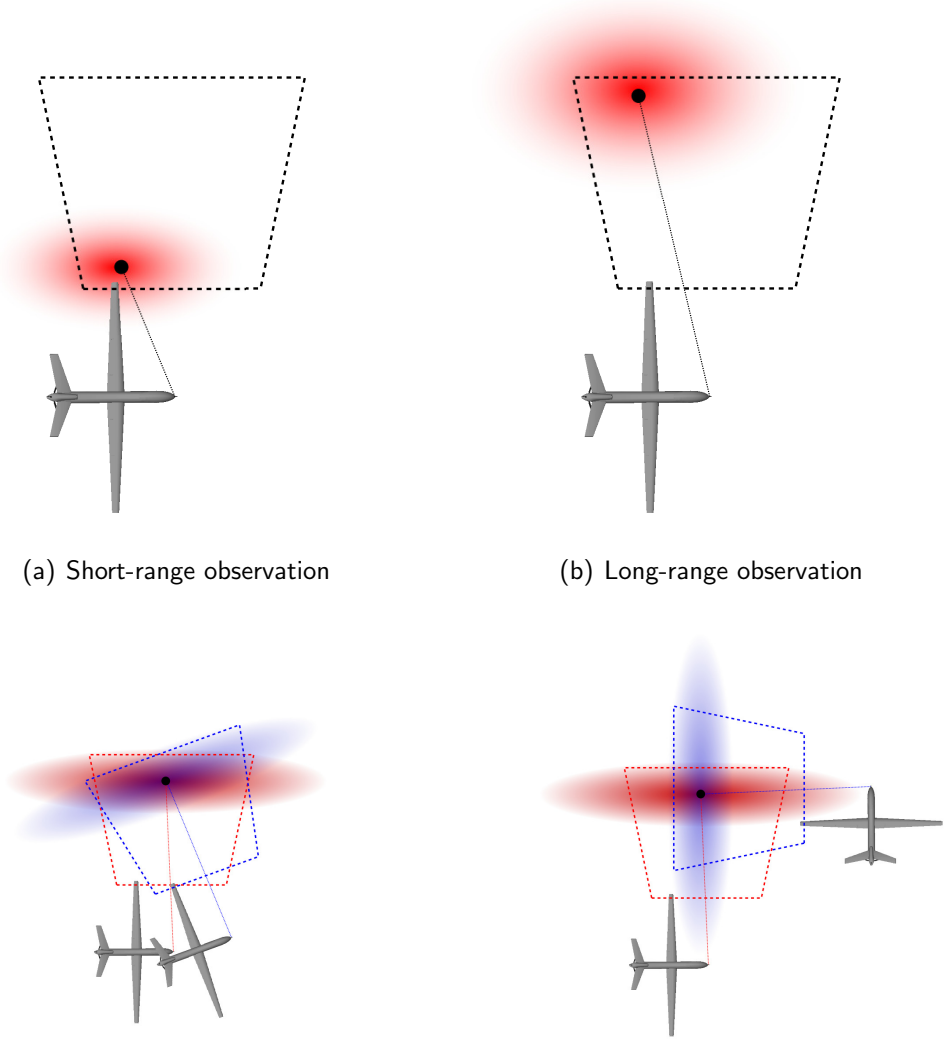
The first class of approaches are heuristic methods to maintain tracking using geometric or reactive guidance rules. Most rely on the intuition that keeping a pursued target as close to the center of the sensor footprint as possible will maintain view in the short-term and provide maximum robustness to disturbances or estimation errors that may place the center of view away from the target's true location. The simplest such means is a control law that directs the UAV to loiter near a target's estimated location and drives the sensor footprint's center to this estimate. For aircraft having a side-looking or gimbaled camera, a common strategy is to enter an orbit around the estimate of radius necessary to maintain that as the desired footprint center. [79, 97, 100] Alternative derivations without the explicit goal of an orbit but instead maintaining a constant relative viewing angle of a target are presented by Rysdyk [107] and Theodorakopoulos and Lacroix. [122] This has been extended to teams of multiple orbiting aircraft, in which by maintaining equal spacing between vehicles, maximally diverse viewpoints are provided. [44, 121, 136] This provides an informal solution to the problem of maximizing observation information diversity described next. For aircraft instead having downward or forward-pointing cameras, sinusoidal, square-wave, and flower-petal trajectories parameterized by a current target location estimate have been proposed to stay on top of targets of varying speeds. [81, 117] For targets having uncertain location, ellipsoidal orbits tracing level curves of a Gaussian uncertainty distribution providing reasonable coverage of an ellipsoidal region around a mean estimate have been suggested. [42, 71] For cases

in which target geolocation estimates may be unavailable or particularly uncertain, visual-servoing has been used to direct UAV motion using control laws tied to the pixel coordinate location of a visible target, usually with the goal of driving it to the image center. [108, 109] When automatic detection is not available, similar control laws may be used that map user joystick input indicating desired displacement of the image center relative to the target to UAV bank commands. [97]

Conceptually, as long as the UAV can react sufficiently quickly to resulting control outputs, pursuit of moving targets is possible in addition to persistent coverage of a static target by simply adjusting control law set-points as new locations are observed. However, heuristic approaches provide no guarantee of successful persistent pursuit or even necessarily applicability of a particular form of trajectory to a given target scenario. By their nature, they also provide no intrinsic means for recovery if a target is not observed where it was expected to be present (or, with the few noted exceptions, handling of growth in target location uncertainty), since they typically consider only most-likely point locations of targets. Finally, they do not take into account the time required for a UAV to react to changes in target location, even if that is easily predicted from currently visible motion, instead just hoping that the settling time of a control law is small relative to the time required for a target to move a substantial fraction of the sensor footprint radius so as to maintain view.

A second class that addresses many of these weaknesses and provides formal reasoning about both uncertainty in target location and limited UAV maneuverability is that which may be generally described as predictive trajectory planning seeking to maximize expected observation quality. The essential principles underlying this are that uncertainty in a target observation is dependent on the pose of a UAV relative to the target (appearing, for instance, in Equation 3.4 as $\mathbf{J}$ varies with this relative pose), while a-posteriori geolocation filter uncertainty is necessarily dependent on the uncertainty of observations received (and thereby the relative UAV poses from which the observations were made). Intuitive graphical examples of these concepts are provided in Figure 3.2.

The trajectory optimization problem may first be defined formally, albeit generally. Suppose a present target location uncertainty distribution $p(\mathbf{x}_t|\mathbf{Z}_t)$, target motion model $p(\mathbf{x}_{t+1}|\mathbf{X}_t)$, and UAV sensor observation model $p(\mathbf{z}_t|\mathbf{x}_t)$ are known, where $\mathbf{Z}_t = \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$ is the observation sequence received so far and $\mathbf{X}_t = \{\mathbf{x}_1 \ldots, \mathbf{x}_t\}$ is a hypothetical target path. Further, let $C(\mathbf{x}_{\mathrm{UAV},t}, \mathbf{z}_t|\mathbf{X}_{\mathrm{UAV},t-1}, \mathbf{Z}_{t-1})$ be a scalar cost function indicating the value of receiving an observation $\mathbf{z}_t$ while having pose $\mathbf{x}_{\mathrm{UAV},t}$ and having previously followed the path $\mathbf{X}_{\mathrm{UAV},t-1} = \{\mathbf{x}_{\mathrm{UAV},1}, \ldots, \mathbf{x}_{\mathrm{UAV},t-1}\}$ and received the observation sequence $\mathbf{Z}_{t-1}$. Then, the optimal action to choose at time t having followed path $\mathbf{X}_{\mathrm{UAV},t}$ and having received observations $\mathbf{Z}_t$ is



(a) Short-range observation

(b) Long-range observation

(c) Similar-pose observations

(d) Orthogonal-pose observations

Figure 3.2: Intuitive examples of varying UAV pose affecting the uncertainty of a single observation (here, that observations of closer targets (a) provide lower uncertainty in general than observations of more distant targets (b)) and the a-posteriori uncertainty of a filter receiving two observations from differing relative poses (here, that two observations having elongated uncertainty ellipses provide nearly overlapping uncertainty when taken nearby, (c) while the same observations may have much smaller overlap—admitting a much smaller set of possible target locations likely under both—when taken from orthogonal orientations (d)).

$$\mathbf{u}_t^\star = \underset{\mathbf{u_t}}{\operatorname{argmax}} \int_{\mathbf{z}_{t+1}} \int_{\mathbf{x}_{t+1}} \int_{\mathbf{x}_t} \begin{array}{l} C\left(f_{\text{motion}}(\mathbf{x}_{\text{UAV},t}, \mathbf{u}_t, \Delta t), \mathbf{z}_{t+1} | \mathbf{X}_{\text{UAV},t}, \mathbf{Z}_t\right) \\ \cdot p(\mathbf{z}_{t+1}|\mathbf{x}_{t+1}) \\ \cdot p(\mathbf{x}_{t+1}|\mathbf{X}_t) \\ \cdot p(\mathbf{x}_t|\mathbf{Z}_t) \end{array} \tag{3.6}$$

and may iterated repeatedly to evaluate the value of applying a control sequence $\mathbf{U}_t = \{\mathbf{u}_1, \ldots, \mathbf{u}_t\}$ and receiving observations $\mathbf{Z}_t$ (of which all but the first are hypothetical) if the target were to follow a hypothetical path $\mathbf{X}_t$ from initial UAV and target state estimates $\mathbf{x}_{\text{UAV},1}$ and $p(\mathbf{x}_t)$ respectively. Note that choosing an optimal $\mathbf{U}_t^\star$ from an initial $\mathbf{x}_{\text{UAV},1}$ and $p(\mathbf{x}_t)$ thus constitutes an additional maximization over all possible $\mathbf{X}_t$ and $\mathbf{Z}_t$ that may occur during its execution.

The cost metric $C(\cdot)$ may be defined a number of ways, so as to either maximize the probability of target visibility (intuitively keeping the target in view), maximize the certainty of or information provided by observations (intuitively seeking "good" observations), or maximize the so-called *information gain* of successive observations to minimize a-posteriori filter uncertainty (intuitively seeking "varied" observations). Scalar representations of observation or a-posteriori Gaussian uncertainty are commonly defined using varied matrix properties of either the so-called Fisher Information Matrix (FIM) or the relevant covariance matrix. Definitions and relative strengths of each matrix metric in the context of UAV tracking are thoroughly discussed by, for instance, Ponda [96] and Skoglar. [116] In practice, both anecdotally and as reported by some authors, [23] many of these produce very similar results. Finally, cost metrics may be defined in several ways over a trajectory, the two most common being the average value over expected observations (so as, for instance, to expect to view a target as frequently as possible or achieve consistently low observation error) and the terminal value (for instance, minimizing final a-posteriori uncertainty at the possible cost of potentially high interim uncertainty).

Quite clearly, directly optimizing Equation 3.6 is completely impractical, as it implicitly invokes hypothetical geolocation filter updates and each of the dimensions over which it integrates and optimizes is a continuous space. Thus, practical implementations make one or more substantial approximations. One of the simplest is to discretize any of the relevant spaces, such as considering only a small discrete set of possible UAV actions or perhaps a single most likely target motion given its current estimated state. Others include ignoring filter uncertainty so that $p(\mathbf{x}_t)$ is a delta function centered at the filter mean, ignoring vehicle pose uncertainty or assuming that only the most likely observation will occur (reducing $p(\mathbf{z}_t|\mathbf{x}_t)$ to a delta function), not running filter observation updates using predicted observations (elim-

inating the $p(\mathbf{x}_{t+1}|\mathbf{x}_t)$ term and effectively assuming a stationary target), planning assuming negative observations (which are usually faster as no camera-ray geometry need be computed) when maximizing the probability of seeing the target at least once during the trajectory, and metric-specific approximations such as roughly emulating information-gain by simply summing expected information matrices and applying the desired matrix metric at the terminal point of the trajectory.

Existing strategies applying such optimization for aerial pursuit fall into one of several categories. A common choice in the literature is to simply and greedily move along the gradient of the selected information metric, corresponding to a single-step optimization that may typically be computed quite efficiently. [23, 59, 60, 96] As expected, a major failing with these is a frequent rush to a local minimum, which may be particularly fatal for sensors such as cameras with finite field of view: since typically the lowest-uncertainty measurements are achieved when the UAV is as close as possible to the target, iterated optimization can easily cause the UAV to simply overfly the target and lose sight. Additionally, once a target is out of view, re-acquiring it is not necessarily guaranteed, as oscillatory behavior skirting the target or even flying directly away may result should a non-minimum phase trajectory have been required to reach it again. Slow direct search or cost-function optimization up to some horizon has also been used, [46, 47, 59, 106] but unsurprisingly, gross approximations, sparse discretizations, and very short horizons are required to maintain tractability. To improve upon this, pre-computation of optimal short-term trajectories given an estimated relative target pose using function optimization or dynamic programming have been proposed, [30, 99] greatly aiding tractability albeit still requiring substantial approximations.

A strongly related issue for multi-UAV teams is the level of collaboration between vehicles. [2] On one end of the spectrum, lies completely independent behavior, in which vehicles geolocate and plan independently, possibly reporting their independent estimates to one another or a central location for comparison or averaging. Substantially overlapping behavior and redundant observations may result, but all computation and communication may be completely decentralized and parallelized. Above this lies forms of ad-hoc cooperation, in which vehicles may asynchronously broadcast their observations, estimates, and intended motions so that each agent has the best possible information with which to choose its motions and may avoid areas or perspectives that are presently well covered by others, but without explicit group decision-making. A stronger level of collaboration still is that simply termed

[2]The terms *cooperation* and *coordination* have received varying definitions in the literature. Here, *cooperation* refers to loose levels of collaboration in which agents may assist one another, while *coordination* implies some form of strongly-coupled joint behaviors.

here cooperative behavior, also known as sequential allocation, in which agents synchronously report their intended motions so that each successive vehicle chooses its actions knowing what preceding ones will do in the near-future. This permits elimination of redundant coverage, but it may still produce arbitrarily poor behavior if the action cost metric is not chosen appropriately. A classical example is that of attempting to pursue two targets using two UAVs: if the metric chosen is to minimize the average per-vehicle distance between the sensor footprint center and each target's estimated location, both will choose to fly towards the midpoint of the target locations and fail to make any observations; if instead, the metric is maximizing average a-posteriori geolocation certainty, they will appropriately assign themselves to a different target. Finally, fully coordinated or joint motion planning, typically implemented as a central planner, treats the optimization problem as a higher-dimensional one in which an action is in fact a vector of actions, one for each vehicle. This permits proper consideration of arbitrarily complex joint behaviors, in which one UAV may be behaving in a highly suboptimally manner locally but yet contributes critically to reaching a global optimum. Clearly, progression along this spectrum is accompanied by greatly increased computational complexity, as action spaces merge and opportunities for parallelism disappear. Two ways to mitigate this are to use sparser or lower-dimensional representations wherever possible to simply reduce problem complexity or to use separable or factorizable representations such as discrete hypotheses to which separate vehicle-to-target assignment may be applied.

Overall, pursuit represents a particularly challenging task, for several reasons. First, the finite size of a UAV's sensor footprint and limited maneuverability results in slow returns to view as the UAV must circle back to re-acquire an overflown target, producing so-called bursty observation dropouts. Uncertain target motion prediction increases the space of likely target locations with the passage of time, making long-horizon planning difficult and reducing the duration of dropouts that may be handled, while large uncertainty in vehicle state produces many possible observation outcomes for a given target location. Taken together, and despite many existing efforts, planning to produce best expected observations remains largely fundamentally intractable. Particularly agile (fast-moving or high-maneuverability) targets or those assumed to be taking evasive action only worsen this, as the space of likely target locations increases greatly between observations. A sure means to improve upon this is to reduce the space of possible target location or actions however possible, and one such way using a sparse and constraining environment representation is proposed in Section 4.3.

3.4 Search with UAVs

If instead a target's location is not known–either because it has never yet been observed or it successfully evaded tracking and has not been seen for some time–a search is required to locate it before pursuit. As small UAVs, particular early ones, have limited payload capacity (thereby restricting possible sensing to lightweight cameras) and maneuverability (limiting pursuit performance), search for targets or objects of interest is a natural application and has likewise received considerable attention. Roughly speaking, existing strategies may be classified based on the level of uniformity with which search attention is directed across the area of interest.

3.4.1 Search in regular areas with uniform prior

If an area being searched is in some sense regularly-shaped (for instance, tightly bounded by a convex simply-connected polygon) and uncertainty in target location is uniformly distributed throughout the area, the search task may be simply phrased–though it may not necessarily be easily accomplished–as covering the entire area using uniformly-directed search attention so as to have observed every point in the environment before a target possibly located in adjacent un-searched space would have had a chance to reach that point.

For stationary targets, this represents a relatively straightforward uniform coverage task. Ablavsky and Snorrason [1] provide commonly cited geometric strategies for UAVs modeled as Dubins cars using straight-line sweeps to form lawnmower, zamboni, and box-spiral patterns. In a similar vein, coverage using autopilot-provided orbit motion primitives has been considered in in Section 5.1. Extending this to multiple vehicles requires some form of region assignment to avoid redundant coverage or the possibility of vehicle collision. Enns et al. [38] suggest extending single-UAV sweeping to interleaved sweeps by multiple vehicles, addressing turning constraints. Beard and McLain [14] consider similar forms of sweeping while avoiding hazards and maintaining a minimum communication distance between vehicles should this be required. For larger areas that may not necessarily form a simple polygon, Maza and Ollero [84] propose a polygonal partitioning of an area, such that one partition is assigned to each UAV and covered using an existing simpler sweeping strategy. An alternative decomposition in which an area is decomposed into slightly-overlapping circular regions that are then covered by spiral coverage patterns is instead put forth by Girard et al. [52] Intriguingly, Jones [68] demonstrates aerial coverage by generating a planar coverage pattern using well studied methods for multiple ground vehicles and maps these to executable UAV waypoints. Overall, the case of sta-

tionary targets may be considered both well-studied and straightforward, however long-duration searches or many vehicles may be required for task completion in large areas.

Targets that may be moving adversarially in an otherwise regular environment having initially uniform location uncertainty must be treated somewhat differently, as target motion will recontaminate already-searched regions. Given a velocity bound on a target sufficiently below that of a UAV and modeling the UAV sensor footprint as a circle lying within the true footprint, McGee and Hedrick [85] show adversarial target search using containment by progressively shrinking irregular orbits. For larger areas and faster targets, Vincent and Rubin [134] suggest parallel sweeping modeling sensor footprints as small rectangles and state a relationship between the required number of UAVs and their minimum necessary speed to assure detection of a target having a particular maximum speed in an area of given size. Overall, the case of adversarial targets may be considered well-studied but is a difficult one as many fast UAVs may be required to assure detection of a target.

3.4.2 Search in irregular areas or with non-uniform prior

If the search area instead has highly irregular shape or substantial interior regions in which a target could not lie (such as obstacles or otherwise untraversable terrain), an outer bounding polygon or decomposition into few convex polygons may present a poor approximation of the area and result in needless coverage of areas in which a target could not lie. Similarly, if the prior uncertainty distribution on possible target location is highly non-uniform, it may be wasteful to cover low-likelihood areas at all and at least logical to search higher-likelihood areas first to achieve shorter expected detection time. Finally, for the case of a target that is moving non-adversarially and modeled stochastically, uniform coverage search as for a stationary target is inapplicable due to recontamination, yet adversarial search may be needlessly conservative, requiring excessive capture time or vehicle resources.

For stationary or stochastically-modeled targets, cellular observation-predicting trajectory planning is commonly applied, usually so as to choose a path for the searchers that maximizes the probability of detection or expected detection time along the anticipated coverage pattern. A first example that still results in uniform coverage of a region is that shown by Ahmadzadeh et al., [5] which plans paths subject to motion constraints and arbitrary initial conditions to maximize the average probability of observing any point in the environment during the path. Polycarpou et al. [95] present a framework later built upon by Flint [40] that plans paths for a team of UAVs searching for a stationary target in an environment with areas hav-

ing varying likelihood of target presence as well as possibly moving threats, which is shown to provide substantially improved detection time over a uniform coverage search. For moving targets, it is common to discretize the environment by either decomposing it into cells containing a probability of target presence lying only within feasible regions or by distributing particles according to an initial prior distribution, running Bayesian probability updates as the search progresses as though for a cellular or particle geolocation filter from Section 3.3.2, and then optimizing paths that maximize probability of detection given expected coverage of cells or particles and their respective probability. Bourgault et al. [17] present a famous early example, in which one or more UAVs searches for a randomly diffusing target represented in a probability grid by moving along the local probability gradient. Despite this local behavior, by sharing observations, well-coordinated behaviors are demonstrated. Tisdale et al. [124, 125] implement and field-test a multi-UAV sequential allocation scheme using horizon planning to maximize the short-term probability of target detection using a similar grid or particle representation receiving Bayesian updates that fully take into account sensor uncertainty, detection likelihoods, and false-detection likelihoods. Yet another example includes that which Geyer [49] describes, using a similarly-updated particle representation for single-vehicle search for a target taking into account occlusion using a pre-computed visibility map and which contains an optimization to greatly speed up hypothetical particle filter updates given expected coverage for a given trajectory. Of particular note is his observation that memory-less planning (that does not perform hypothetical filter updates in response to expected observations at all but simply independently attempts to maximize the probability of target detection at each step) performs nearly as well as full planning. Conceptually, these forms of planning that optimize detection time or short-term probability of detection are fundamentally identical to the aforementioned trajectory planning for pursuit of Section 3.3.3, [60, 106] but can be perceived as having more target hypotheses and much weaker target location certainty, greatly worsening planning tractability and in practice requiring stronger approximations or suboptimal planning. Simulations presented in Section 5.2 suggest that in some environments, these methods may still prove quite practical if the environment and target uncertainty can be represented compactly.

Occasionally, further target modeling permits simplified trajectory generation, even for moving targets. One unique case is that of a missing person in a wilderness rescue scenario. Goodrich et al. [54] argue that an appropriate distribution for target location in such an instance is a peak at the point last seen with radially symmetric falloff outward from this point. For this case, the authors suggest a uniform coverage pattern that spirals outward from the peak, thereby tracing probability contours and

minimizing expected time until detection.

Meanwhile, search for adversarially moving targets having non-uniform initial location probability or in irregularly-shaped environments (particularly roads as is the focus here) has not been well considered in the context of aerial search. For some environments, the aforementioned decomposition into regions that are then uniformly swept for adversarial targets may be feasible, but many details such as possible target motion between regions remain unclear. As reviewed in Section 3.1, adversarial search in arbitrary polygons has been well-studied for ground vehicles and serves as inspiration for mapping UAV search tasks to such abstractions.

3.5 Recapture

If observations are not received during pursuit for a long duration, tracking may be considered to have failed. At this point, the predicted location of the target is likely to have uncertainty covering a wide area, providing little use to short-term pursuit-level observer motion planning, as any trajectory covering a portion of the uncertain area will have nearly the same poor likelihood of detection. It may also happen that a target sighting is briefly reported, but no UAVs are near the reported location to begin pursuit. By the time any can reach the area, uncertainty in the target's location may have grown well beyond the point at which a UAV could expect to see the target at its reported location; rather, some form of local search is required. This case of recapture does not appear to have received attention in the aerial search and tracking literature, yet in practice it may arise frequently when attempting to track agile or evasive targets with poorly maneuverable, low-accuracy sensors. Resorting to initiation of a search of the surrounding area may be heavy-handed, possibly requiring the requisition of additional UAVs, substantial setup time, or the coverage of an unnecessarily large area by its conclusion. A rare instance of work on a related case is that by Krishnamoorthy et al. [76] in which a patrolling UAV receives notifications of recent target presence by fixed unmanned ground sensors, and an optimal sensor visitation pattern for the UAV and bounded capture time are derived by dynamic programming for the special case of Manhattan grid road networks (extended to general road graphs in ongoing work).

Though the recapture scenario appears to have received little consideration in the existing literature on UAV search and tracking, several intuitive and some existing pursuit-evasion strategies may be applicable. For instance, if a target is lost during pursuit, varying possible target motions can be predicted to generate discrete hypotheses of increasing uncertainty. An optimal trajectory can be planned as for pursuit in expectation over all these possible motions, though likely requiring con-

siderably more UAVs than for typical pursuit to achieve meaningful results. One example of this is the iteration of a process model along roads, splitting at branches to produce discrete hypotheses, [57] a concept built upon further as a growing contaminated subset of the underlying road network graph in Section 6.3 in realization that the tractability of producing recapture plans depends critically on having few, moderately-accurate hypotheses of target location implying a strong need for great target predictability. This is of course difficult in open environments in which, subject to possible motion constraints, the target may move in nearly any direction. Thus, where having few hypotheses is impractical, it may be possible to simply diffuse possible target location using a speed estimate or bound, creating a progressively larger area that may be searched using any of the preceding methods for areas with non-uniform prior. Given a sufficient number of UAVs, it may also be feasible to optimally cover the target uncertainty distribution using sensor placement techniques [58] so that the target is most likely to be detected both instantaneously and once it moves. Finally, there may be cases in which a target's location is moderately actually known yet it cannot be continuously observed because its speed is comparable to that of pursuing UAVs, so that the target is in some sense constantly in need of recapture. This may be treated as more of a pursuit-evasion problem, having, as previously mentioned, limited mention in the aerial tracking literature.

CHAPTER 4

Geolocation and Pursuit

Initial effort broadly focused on understanding the fundamental difficulties underlying the tasks of search and tracking through field experimentation. Section 4.1 describes in detail various aspects of these experiments, basic challenges encountered, and the essential conclusions that motivate improved representations for better performance. Section 4.2 explains the need and several strategies for accurately representing and combining high-uncertainty observations in the face of a highly nonlinear observation function to produce geolocation estimates with otherwise impossible accuracy. As a first step towards exploitation of environmental structure to improve task performance, the use of a road constraint during pursuit is considered in Section 4.3 and is shown to have a number of obvious but inspirational benefits. Extension of this idea to road networks is motivated by the impracticality of a naive cellular representation in the style of existing work. Finally, conclusions drawn from this work are summarized in Section 4.4, strongly motivating the proposed use of environmental structure in the succeeding chapter to simplify the problem and permit the use of theoretically analyzable abstractions.

4.1 Experimental Motivation and Methodology

Experiments have considered the use of stock or minimally-modified commercially available small robotic vehicles so as to build upon already highly refined low-level controllers and autopilots to permit high-level algorithmic study and demonstrate

capability extension through field-installable software. Taking inspiration from the complementary nature of differences betwen air and ground vehicles noted in Section 2.7–higher fidelity sensing and larger computing payloads available from ground vehicles, but faster wide-area sensor coverage with little concern for terrain obstacles from aircraft–a heterogeneous experimental collaboration platform comprised of one or more UGVs and one or more UAVs was used.

The base system is comprised of a single UAV, a single UGV, and a common high-level control interface. The UAV platform selected for this system is the RQ-11B Raven [3] produced by AeroVironment, Inc and pictured in Figure 4.1(b). Relatively inexpensive and easily transported by a single person, the Raven can be hand-launched from a small clearing, providing an excellent example of the class of UAVs considered. It is also possibly the currently most widely deployed UAV system in the world–with over 10,000 units delivered–so that applicable software extensions may have immediate worldwide impact. As the UGV platform, a stock iRobot Explosive Ordinance Disposal (EOD) Packbot with a development payload and additional sensing is used. Based upon a very widely used small ground robot that is easily transported in a Humvee and deployable by a single operator, it is pictured in Figure 4.1(a). Finally, the common control interface appears as software running on a ruggedized laptop computer (such as that shown in Figure 4.1(c)) for overall system portability.

(a) (b) (c)

Figure 4.1: The UGV, UAV (previously pictured in Figure 1.3), and computing hardware that comprises this heterogeneous collaborative surveillance system. All components are man-portable and battery operated.

Primarily only aspects of UAV search and tracking are considered here, though such hetereogeneous collaboration has shown itself [32] to be of great benefit to the completion of practical reconnaissance missions and, while beyond the immediate

scope of this thesis, is considered a clearly fruitful area of future study.

4.1.1 Unified control infrastructure for fielded vehicles

As nearly all such vehicles, including those selected as the experimental platform, are in practice either manually controlled or directed using manually specified waypoints individually provided to each vehicle, a first step towards autonomous collaborative field-reconnaissance is a unified control interface. This was implemented in the form of an overhead satellite map window on which overall mission state is overlaid and a function-specific control window for each active vehicle, examples of which appear in Figure 4.2. Using this, an operator might for instance select an area to search using the map window, designate an object of interest for tracking in live video presented in a vehicle's control window (perhaps an initial UAV detection), observe its geolocated position back in the map window, and direct another vehicle to the same location with a single high-level command (perhaps to direct a UGV for closer inspection). Described in further detail in a system-specific publication, [32] such interfaces are not a focus of this thesis, though they are a necessary component of an effective collaboration system using multiple vehicles performing coupled search and pursuit maneuvers. Additionally, with few existing examples, [62, 118] this is believed to be one of the first demonstrations of such unified air-ground mission execution.

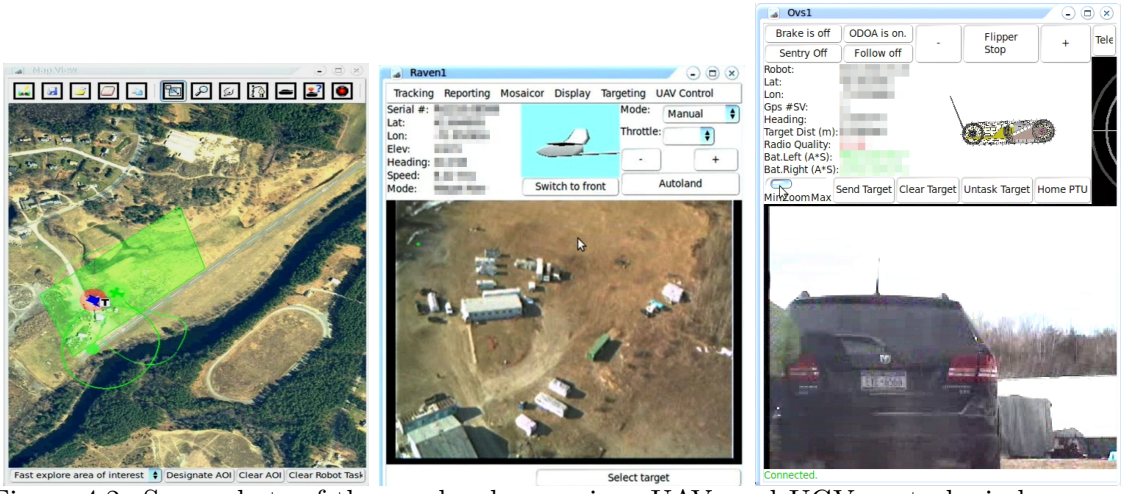


Figure 4.2: Screenshots of the overhead map view, UAV, and UGV control window components of the common operator interface for unified collaborative control.

4.1.2 Geolocation observations from visual tracking

In order to produce observations in the form of Equation 2.1 with which to generate geolocation observations, some means of continuous target detection or tracking is required. For the relatively wide angle camera and lossy video transmission of the UAV considered, coupled with the desire to track novel targets in rescue or surveillance scenarios, an operator-initiated tracking scheme is chosen over any form of automatic visual target recognition, though one such form is briefly considered. Under this scheme, as a UAV searches an area or attempts to recapture a target, targets visible in the video feed are designated by an operator click, these designated targets are then tracked through succeeding video frames without further operator intervention by one of several algorithms, and a recovery procedure is executed to attempt to re-acquire targets that have become occluded by terrain or that have left the view of the camera. The center of this designated or tracked position is used as the image coordinate observation and is then used to generate a ray that is intersected with an estimate of the local groundplane derived from an assumed terrain database as described in Section 3.3.1 to produce a single geolocation observation in the space of target location.

The system developed handles high levels of camera motion caused by wind, performs out-of-view tracking recovery that is frequently required in practice, and handles tracking and recovery through video corruption or blackouts caused by communications glitches. As previously noted, wind is a particular problem for aircraft of such size as it causes rapid accelerations in the vehicle'ss attitude and hence considerable unwanted camera motion. This means an objects location in one video frame might be significantly different in subsequent frames, causing failures in many traditional tracking schemes and also making it difficult for a human user to initially designate a target in the video stream.

For the purposes of this work, it is assumed that the camera does not have variable zoom capability or a mechanical gimbal for pointing. The Raven has both electronic zoom and software panning abilities, however these are not used both for greater applicability to other UAVs and because these were not designed for automatic software control in the case of the Raven. The algorithms described here will operate perfectly on a vehicle with either and may be extended to take advantage of them.

The visual tracking pipeline implemented is described more completely in Appendix A. Briefly, it is divided into several segments. The first is image stabilization to assist a human operator's designation or an automatic tracking algorithm by eliminating effects of rapid unwanted camera motion caused by wind-induced attitude disturbances. Lucas-Kanade alignment [10] is one of the formative approaches. How-

ever, with large amounts of image motion, modification of this traditional approach to avoid local minima in its internal alignment search process was required. The next segment is the process of designation itself. Adopting the methodology of operator-assisted designation rather than highlighting a target in previously captured imagery or by using a joystick to continuously aim a gimbal or cursor, [97] the operator either draws a box around the target that is then automatically sized to best fit the identified target or clicks on the center of the target, causing an estimated box to be drawn around it that is then likewise resized. The last segment of the tracking process are the algorithms themselves used to track a designated target. For this purpose, both a mean-shift color template tracker (using a direct implementation of this concept [27] as well as a variation of an alternative [26] that continuously re-weights a set of features derived from functions of color values) and spatial appearance tracking using a modified form of the well-known Kanade-Lucas-Tomasi (KLT) algorithm. [127] Tracker selection remains a challenge given differing performance in varying scenarios highlighted in Figure 4.3. On average, the spatial template tracker performs best and is selected by default, though the operator may choose an alternate tracker at any time if appropriate.

(a) Changing perspective (b) Similar objects (c) Background in template

Figure 4.3: Examples showing two cases, (a) and (c), for which non-spatial tracking dominates, and another, (b), for which a spatial template is more appropriate.

Inevitably, a tracked target will often leave the view of a UAV's camera, especially when lacking a gimbal or if zoomed in, reducing the field of view. The described system attempts to recover from out-of-view tracking losses by estimating the camera motion by iterative image stabilization to predict the new location of the target, and when the predicted location re-enters the view, searches for the target inside a sizeable search window surrounding this expected location. In addition to out-of-view events, other factors may induce tracker failure. For instance, analog and digital radio communications loss causes corruption in the video stream. Failures in the tracker are heuristically indicated as occasions when the template no longer matches well with the search window, and then recovery is attempted once the period

of corruption is over. An example of this procedure applied to both video corruption and an out-of-frame event is shown in Figure 4.4.

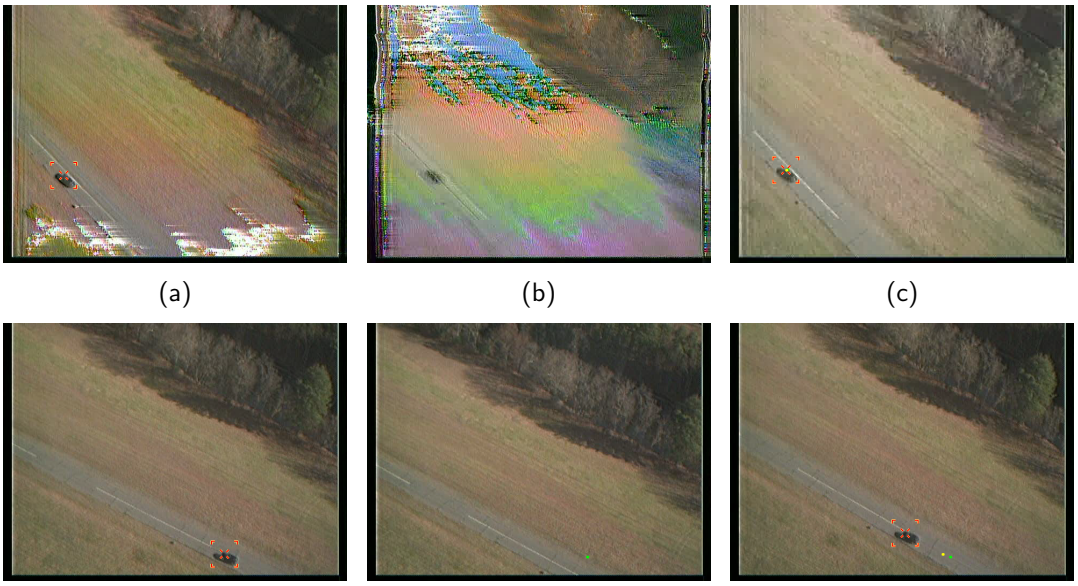
4.1.3 Initial experimental conclusions

The described tracking pipeline, built on practical modifications of two popular template tracking methods, has proven incredibly reliable in practice, with tracking durations on the order of a minute without operator intervention not uncommon for unambiguous targets in still weather. Successful tracking has been demonstrated in open areas and of targets moving on roads and within clutter, examples of which are shown in Figure 4.5. Formal testing against standard tracking datasets remains to be performed, however.

Overall experimentation has accumulated several tens of hours of testing and data collection in a variety of environments, in varying wind conditions, of stationary and varying-speed targets including several pursuit trials. Two main conclusions based on field experimentation might be made.

The first is that individual target geolocation observations may be extremely inaccurate, primarily due to errors in vehicle orientation (particularly heading) caused by latency and jitter in the state data stream, wind-induced angular velocities exceeding the signal bandwidth of both onboard state estimation and the state data stream, and the lack of a reliable heading reference beyond Earth's weak magnetic field used by an onboard compass. In fact, common existing geolocation filtering methods chronicled in Section 3.3.2 provide only moderate improvement due to the magnitude of these errors and slowly varying biases. Section 4.2 presents several ways that have been considered to improve upon this.

Second is the observation that common autopilot implementations providing slow response to desired control inputs–as both latency in acting upon these and a low frequency cut-off in system dynamics–coupled with the typical prioritization of vehicle stability and safety over responsiveness results in large minimum turning radii (50 to one hundred meters) and high latency for a control input to take effect (up to several seconds). The practical impact is that if a target is overflown and the vehicle must turn around, the process may take on the order of half a minute, producing long dropouts in observations of a target, during which motion predictions coupled with a prior geolocation estimate are relied upon. This motivates the use of multiple UAVs to maintain persistent tracking during individual dropouts as well as the importance of accurate geolocation filtering and careful modeling of target motion to maximize the accuracy of predictions. Section 4.3 introduces the use of structure as a promising means to accomplish this.



(a) (b) (c)

(d) (e) (f)

Figure 4.4: Different types of tracking recovery. The top row shows the tracking recovery after a period of communications corruption. The bottom row shows the tracking recovery after the target leaves the field of view.

(a) Open area (b) On-road (c) Cluttered area

Figure 4.5: Examples of tracking a target in the open, moving on a road, and in a cluttered area.

4.2 Geolocation Under High Uncertainty

Once a target has been located and is being actively tracked, geolocation filtering such as using the existing methods reviewed in Section 3.3.2 is needed to provide an accurate estimate given a series of inaccurate individual observations. However, the nature of errors in observations produced by the described experimental platform presented particular challenges to these methods.

A typical plot of geolocation observations over a single orbit of a stationary target as shown in Figure 4.6 indicates the presence of a large, time-varying bias in addition to considerable per-observation error. In large part this is due to error in measurements of vehicle heading with which geolocation observations are made. Figure 4.7 depicts typical heading error over the course of several orbits, exhibiting error that is large, time-varying, and non-Gaussian in distribution. This violates typical existing filter assumptions of zero-mean or fixed-bias, Gaussian-distributed error. The immediate impact on uncertainty modeling, shown in Figure 4.8, is that Jacobian linearization of the highly nonlinear observation function in Equation 3.3 as used by most strategies in Section 3.3.2 can produce arbitrarily poor approximations of the actual crescent-shaped uncertainty distribution. Meanwhile, as also shown in the figure, imposing a Gaussian approximation of the true uncertainty distribution can at best provide an uninformative, over-conservative of little use to information-theoretic trajectory planning. Therefore, an effective filter representation must have two characteristics: it must handle the transformation of large vehicle state error through the highly nonlinear observation function and must represent target location uncertainty using a distribution that is non-Gaussian in this space.

While the magnitude of the errors described may represent a somewhat extreme example, the challenge of geolocation under high observer state uncertainty is a fundamental one and remains relevant with continued movement towards ever-smaller and inexpensive UAVs with little payload budget in every sense. Further, object localization with large, asymmetric error using a highly nonlinear observation function represents a challenging nonlinear estimation problem in its own right.

To address this, three geolocation filters were developed that better handle this form and magnitude of uncertainty. The first uses an evidence-grid to accumulate sampled uncertainty over many observations of a stationary target. The latter two take inspiration from the highly polar nature of the uncertainty, one being a particle filter that receives observations in the space of range and bearing, and the second being a bounded recursive filter that operates in an over-parameterized space to parametrically represent uncertainty distributions having this crescent form. All have shown themselves to typically outperform existing methods on the geolocation

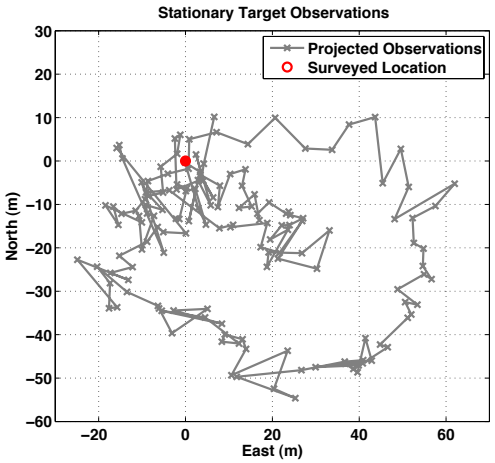


Figure 4.6: Observations of a stationary target over one orbit, exhibiting large, time-varying bias.

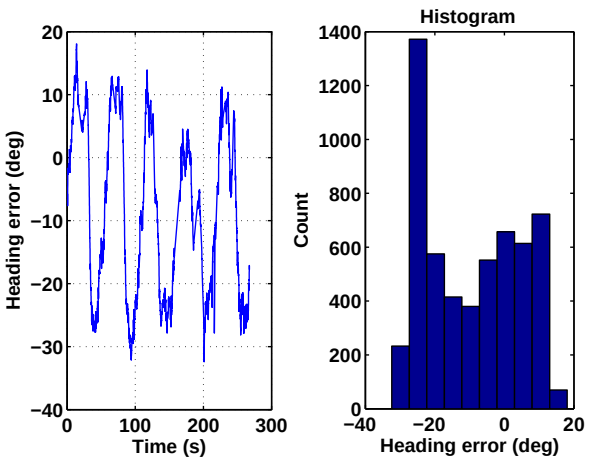


Figure 4.7: UAV heading error over several orbits, exhibiting large, time-varying, non-Gaussian distributed error.

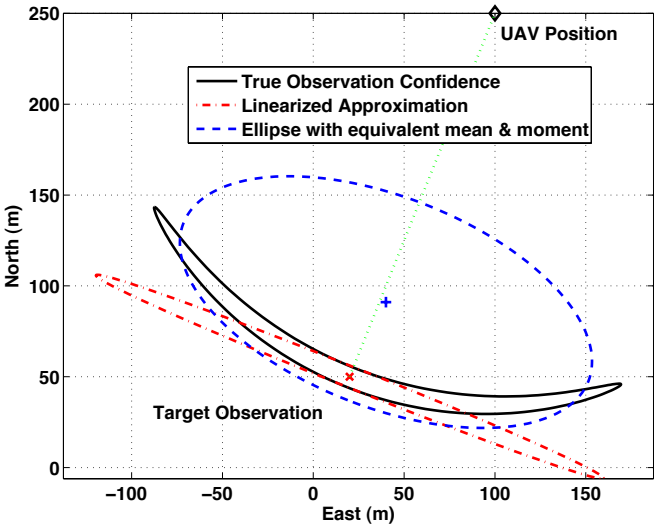


Figure 4.8: Comparison of the true observation uncertainty distribution with a Gaussian approximation generated by Jacobian linearization and a Gaussian approximation having the same mean and covariance as the true distribution.

data collected during experimentation, however thorough comparison between the filters and against existing methods remains to be performed.

4.2.1 Evidence-grid filter

To avoid modeling the complex crescent-shaped uncertainty distributions of each geolocation measurement parametrically, a discretized representation is considered that captures the shape of this distribution to produce a geolocation filter appropriate for stationary targets. [88]

The solution space – the location of the target in world coordinates – is first represented as a discrete grid, in which each cell contains the relative likelihood of the target being in that cell, much as a cellular probability grid receiving Bayesian likelihood updates during a search, however in this case cells do not contain explicit probabilities. Then, the essential idea is to sample the error distribution for each of the components of the vehicle pose and derive, after performing this and accumulating these distributions over a number of observations, the most likely target location. Formally, define

$$\mathbf{x}^{\text{world}}_{\text{tgt,sample}} = f_{\text{sample}}(\mathbf{s}_{\text{UAV}}, \mathbf{x}^{\text{img}}_{\text{tgt}}, \mathbf{e}) = f_{\text{obs}}(\mathbf{s}_{\text{UAV}} + \mathbf{e}, \mathbf{x}^{\text{img}}_{\text{tgt}}), \tag{4.1}$$

where f_{obs} is the pseudo-observation generation function defined in Equation 3.3 and $\mathbf{e}$ is a specific offset in the UAV state. The filter approach is to use $f_{\text{sample}}(\cdot)$ to provide discrete geolocation hypotheses from a distribution of UAV state errors, $\mathbf{E}$, each comprised of many discrete state offsets, $\mathbf{e}$. The offsets that form $\mathbf{E}$ are sampled from the modeled error distribution of the UAV state. The error model comprises all dimensions of the vehicle pose, with the heading error being the dominant dimension in our case.

Next, let the pair of functions

$$(i, j) = \text{world2grid}(\mathbf{x}^{\text{world}}) \tag{4.2}$$

$$\overline{\mathbf{x}}^{\text{world}} = \text{grid2world}(i', j') \tag{4.3}$$

define the mapping from any point in world frame to the integer index (i, j) in the grid, and from any integer index (i', j') in the grid to world coordinates, respectively.

Given this, the algorithm may be defined formally as shown in Algorithm 4.1. Starting with a blank grid representing the discretized world space comprised of $M \times N$ cells, all states of the vehicle $\mathbf{S}_{UAV} = \{\mathbf{s}_{UAV}\}$ and corresponding image pixel detections $\mathbf{X}^{\text{img}}_{\text{tgt}} = \{\mathbf{x}^{\text{img}}_{\text{tgt}}\}$ at which the target was observed during a flight

trajectory. For each of these, the algorithm loops through a discrete sampling of the space of possible error values $\mathbf{e}$ along the sensor or state dimensions constituting substantial error sources. For each of these, the observation function $f_{\text{sample}}(\cdot)$ is applied to compute the world position of the target given this $\mathbf{e}$. To produce a best continuous-space estimate of the target's location, a kernel density estimate (KDE) [112] is computed by imposing a likelihood kernel function $K(\cdot)$ around each sampled world position. A two dimensional Gaussian kernel with standard deviation of one grid cell (representing the kernel bandwidth, here designated h) was selected in the implementation. The updates to the grid can be computed incrementally for each frame, making the filter efficient. The final estimate from the filter is computed as a weighted mean, $\mathbf{x}_{\text{est}}$, over the grid.

Input: N, M, $\mathbf{S}_{\text{UAV}} = \{\mathbf{s}_{UAV}\}$, $\mathbf{X}_{\text{tgt}}^{\text{img}} = \{\mathbf{x}_{\text{tgt}}^{\text{img}}\}$, $\mathbf{E} = \{\mathbf{e}\}$, h
Output: Best geolocation estimate $\mathbf{x}_{\text{est}}$

$\text{grid}[1..M][1..N] = 0$
foreach $\{s_{\text{UAV}}, \mathbf{x}_{\text{tgt}}^{\text{img}}\} \in \{S_{\text{UAV}}, \mathbf{X}_{\text{tgt}}^{\text{img}}\}$ **do**
  foreach $\mathbf{e} \in \mathbf{E}$ **do**
    $\mathbf{x}' = f_{\text{sample}}(\mathbf{s}_{\text{UAV}}, \mathbf{x}_{\text{tgt}}^{\text{img}}, \mathbf{e})$
    for $i = 1$ *to* M **do**
      for $j = 1$ *to* N **do**
        $\mathbf{x} = \text{grid2world}(i, j)$
        $\text{grid}[i][j] = \text{grid}[i][j] + K_h(\mathbf{x}^{(k)} - \mathbf{x}'^{(k)})$
$\xi = \sum_{i=1}^{N} \sum_{j=1}^{M} (\text{grid}[i][j])$ `// Compute normalization factor`
$i_{\text{est}} = \sum_{i=1}^{N} \sum_{j=1}^{M} (i \times \text{grid}[i][j]/\xi)$
$j_{\text{est}} = \sum_{i=1}^{N} \sum_{j=1}^{M} (j \times \text{grid}[i][j]/\xi)$
$\mathbf{x}_{\text{est}} = \text{grid2world}(i_{\text{est}}, j_{\text{est}})$

Algorithm 4.1: Formal description of the evidence grid geolocation filter.

Specification of the samples in the error distribution $\mathbf{E}$ depends on the vehicle uncertainty model. In the experiments performed here, a conservative heading uncertainty distribution is chosen as a uniform distribution over ± 45 degrees. The errors of the other pose dimensions (roll, pitch, and position) are less profound and are noted empirically to be closer to Gaussian distributions. The specific values chosen are zero mean Gaussians with 3σ bounds of 5 degrees for roll and pitch, and 7 meters for position.

For the results depicted here, 2000 samples for each observation are chosen, with a grid size of 500m by 500m and each cell measuring 5m by 5m. An example of a

sampling from a single observation from a flight test can be seen in Figure 4.9. There is an obvious trade-off between increasing the number of samples or the number of cells to improve accuracy and maintaining a tractable computational burden.

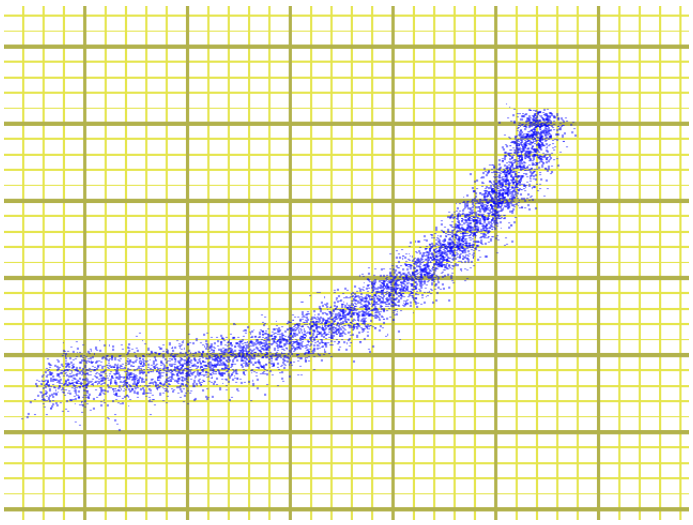


Figure 4.9: Illustration of a single observation, sampling the UAV uncertainty distribution and projecting geolocation estimates with each UAV state offset applied onto the ground plane. These are then accumulated over a series of observations. Scale: large grid cells are 50×50 meters.

Figure 4.10 provides an example of the filter in operation though a series of snapshots from an actual geolocation experiment in which a single UAV orbited a stationary target. Table 4.1 provides a comparison between this filter, a reasonably tuned EKF using an appropriate constant-position process model and matching uncertainty covariances, and a naive Cartesian mean of all observation coordinates. Lastly, Figure 4.11 provide two illustrative examples from separate geolocation experiments (one in still winds and the other in extremely stiff winds) showing the relative superiority of this strategy.

Both empirically and as the provided data suggest, this filter works very well in practice. Key advantages are that no assumptions need be made about the form of observation uncertainty, and that if the distribution is known, arbitrarily complex ones may be approximately represented. One disadvantage is that no clear quantitative confidence measure is available, and any chosen will necessarily be conservative given the additive nature of observation accumulation. Additionally, while its computational burden has not presented a problem in practice, greatly increasing the

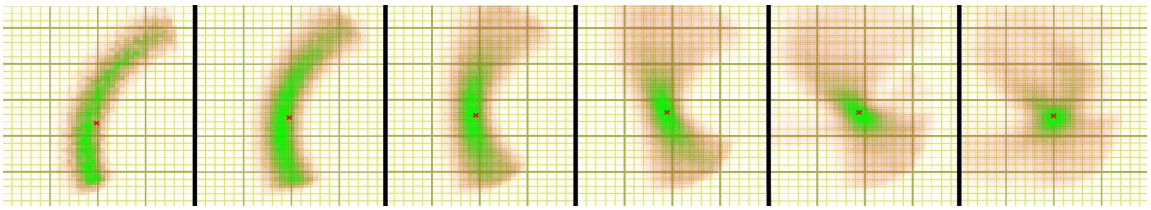
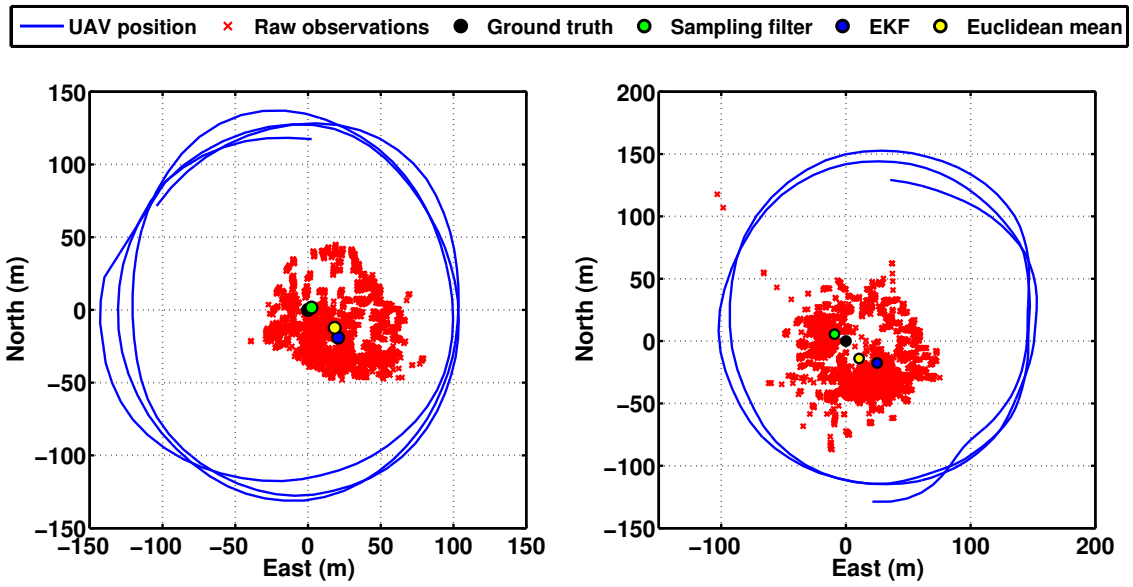


Figure 4.10: Visualization of the filter over a UAV's orbit of a target, in chronological order. The 2D evidence grid is represented as a grid of relative cell probabilities (the two axes are the North and East location of the object). Cells vary from bright green (high target probability) to red (low probability) to blank (none). Scale: large grid cells are 50×50 meters.

Flight #	Wind	Head Err	Alt Err	Mean	Kalman	Evidence-grid
1	High	Moderate	Moderate	17.5m	30.6m	10.6m
2	Moderate	High	Moderate	22.0m	28.1m	2.9m
3	Moderate	Very high	High	8.7m	15.3m	4.5m
4	Low	Moderate	Moderate	13.4m	19.7m	5.4m
5	High	Moderate	Moderate	29.6m	35.2m	8.0m
6	High	High	High	7.3m	15.4m	7.7m
7	Low	High	High	15.0m	22.2m	5.5m
Mean				16.2m	23.8m	6.4m

Table 4.1: Comparison of evidence grid filter against a reasonably tuned EKF and a naive Cartesian mean estimate over several flight tests



(a) (b)

Figure 4.11: Plot of geolocation results from two test flights, under both (a) low to moderate wind and (b) high-wind conditions. The evidence-grid error-sampling filter outperforms an EKF and a naive Cartesian mean in both cases.

number of samples per observations or using a system having an extremely complicated f_{obs} function may become intractable. Finally, it is not easily extended to moving targets. Two obvious potential methods for such an extension include applying a diffusion kernel between observations and adding an additional dimension over discretized velocity vectors; both increase computational requirements substantially and are unlikely to perform well.

4.2.2 Range-bearing particle filter

Taking inspiration from such sampled representations of location uncertainty and the polar nature of a given observation's uncertainty similar to that present in range-only localization problems, [33] a particle filter taking observations parameterized as a polar range and bearing was also considered.

First, a particle filter essentially identical to that described in Section 3.3.2 using pseudo-observations is constructed. Then, given an observation $(s_{\text{UAV}}, \mathbf{x}_{\text{tgt}}^{\text{img}})$ and Gaussian uncertainties on UAV state having covariance Σ_{UAV}, $f_{\text{obs}}(s_{\text{UAV}}, \mathbf{x}_{\text{tgt}}^{\text{img}})$ is applied as before to produce a location in world coordinates $\mathbf{x}_{\text{tgt}}^{\text{world}}$ (here, explicitly projected in 2D to a local ground-plane). However, rather than simply applying linearized Jacobian transformation of Σ_{UAV} to produce uncertainties Σ_{x} in the Cartesian observation location, it is first transformed into polar coordinates as

$$\begin{bmatrix} r \\ \beta \end{bmatrix} = f_{r\beta\ \text{obs}}(s_{\text{UAV}}, \mathbf{x}_{\text{tgt}}^{\text{img}}) = \begin{bmatrix} \sqrt{(x_{\text{tgt}} - x_{\text{UAV}})^2 + (y_{\text{tgt}} - y_{\text{UAV}})^2} \\ \text{atan2}(y_{\text{tgt}} - y_{\text{UAV}}, x_{\text{tgt}} - x_{\text{UAV}}) \end{bmatrix}, \tag{4.4}$$

where $[x_{\text{tgt}}, y_{\text{tgt}}]^T = \mathbf{x}_{\text{tgt}}^{\text{world}}$ and $[x_{\text{UAV}}, y_{\text{UAV}}]^T$ is the projection of $\mathbf{x}_{\text{UAV}}^{\text{world}}$ onto the same local ground-plane. Then, a new linearized Jacobian transformation of UAV state uncertainty may be defined as

$$\Sigma_{r\beta} = \frac{\partial f_{r\beta\ \text{obs}}}{\partial \mathbf{s}_{\text{UAV}}} \Sigma_{\text{UAV}} \frac{\partial f_{r\beta\ \text{obs}}}{\partial \mathbf{s}_{\text{UAV}}}^T. \tag{4.5}$$

Despite still being a linearization of the observation function, the range-bearing Gaussian approximation using this covariance can better model the shape of the true observation uncertainty than either Cartesian Jacobian linearization or the Unscented Transform, as Figure 4.12 indicates.

Completing the implementation of the particle filter, as pseudo-observations are received as a polar position and uncertainty relative to the current UAV position, each particle receives a likelihood update using its likelihood under the observation Gaussian distribution as before except that now each particle is momentarily

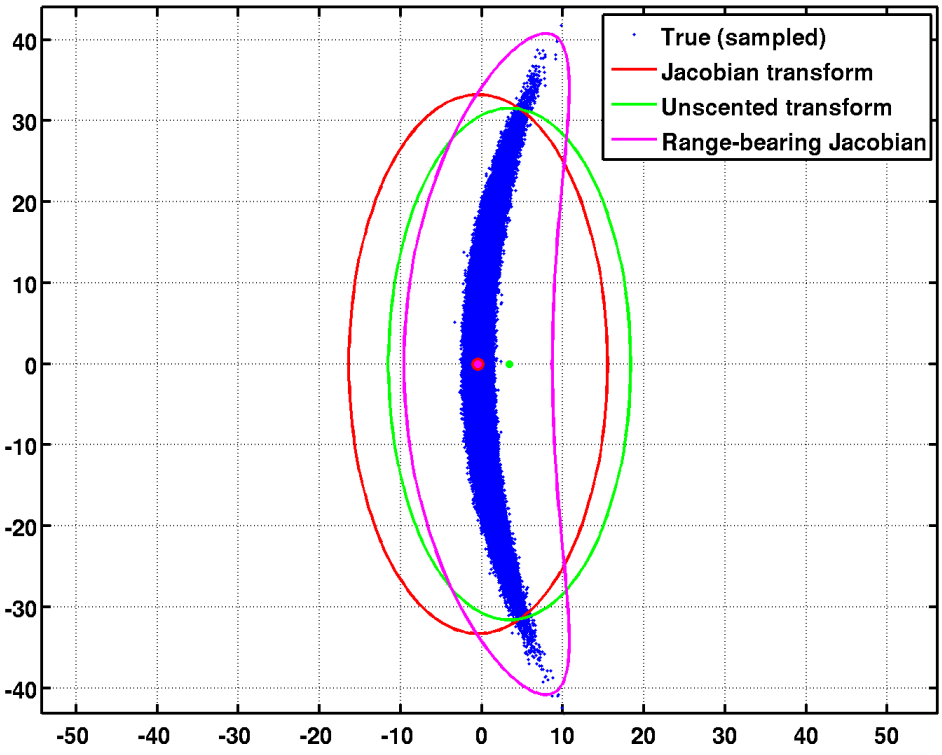


Figure 4.12: Comparison of uncertainty representation approximations compared with a monte-carlo sampling of the true distribution for a single observation with typical parameters (axis scale is meters), similar to Figure 4.8. Of these, a Gaussian uncertainty distribution represented in range-bearing space best matches the shape of the true distribution. The Unscented Transform in Cartesian space better captures the sample mean but remains constrained to a Gaussian.

re-parameterized in polar coordinates using the latter half of Equation 4.4 for the evaluation.

A series of snapshots of this filter during pursuit of a target moving along a road is shown in Figure 4.13

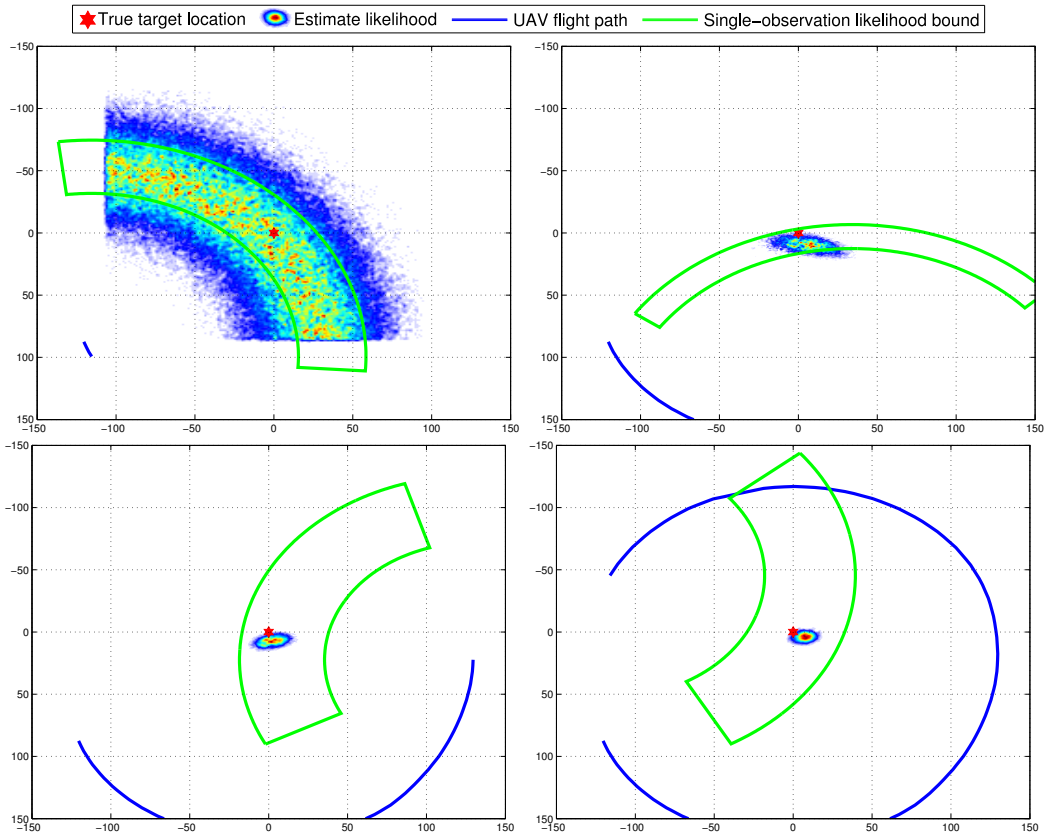


Figure 4.13: Snapshots during the execution of the range-bearing space particle filter during geolocation of a target. In this instance, observations are provided as Gaussian in range and uniformly distributed in bearing around the point observation. The estimate likelihood is here represented by a relative particle density. After approximately half an orbit, the filter converges to an estimate with approximately 12m error.

In practice, this filter performed approximately as well as the traditional form of a particle filter of Section 3.3.2 (in which f_{obs}^{-1} is linearized around each particle and the particle projected into image coordinates to compute a likelihood for each particle), but with much less computation, instead requiring only a single evaluation and linearization of f_{obs}. Thus, realtime execution with a typical particle set size of 5000 presented no difficulty.

While formal comparison against existing and the other developed filters remains to be performed, several conclusions may be made. First, though accuracy and validity of the confidence measure is better for moving targets than an EKF, this filter is still not nearly as accurate as the evidence-grid filter applied to stationary targets. Additionally, both severe sensitivity to the accuracy of the target process model (as slow but discontinuous drift in observation error may result in overconfident clustering at an incorrect location) and a strong need for delicate outlier rejection (as inlier observations may themselves be highly erroneous) were observed. Together, these imply a need for large numbers of particles (though still a tractable number) and careful tuning varying between target types. Finally, though again tractable on portable ground-station computing hardware, the substantial level of computation required exceeds what is practical to embed aboard aircraft in the foreseeable future. Likewise, as a representation comprised of a large set of hypotheses, the same difficulties performing observation-predictive planning as noted in Section 3.3.3 apply in full force.

4.2.3 Over-parameterized bounded filter

Given these ongoing weaknesses, yet another form of geolocation filter is motivated, with several properties standing out. First, while thus far the general crescent shape of observation uncertainty distributions can be approximately modeled parametrically as a range and bearing, the a-posteriori distribution does not retain this shape and has only been sampled, maintaining a desire for a filter with an exclusively analytical representation. Additionally, given the intended use in cooperating teams and as a continuous estimate to seed observation-predictive trajectory planning, a representation that is compact to transmit and rapid both to apply observation updates and to evaluate observation likelihoods is needed and would conveniently be provided by a small parametric representation. Separately, given that large, time-varying error is not well modeled by any particular probability distribution (as a bias may dwell near consistent values), a representation instead considering an error *bound* within which offsets must lie may prove more appropriate.

Therefore, a filter similar to one proposed for range-only SLAM [120] was applied, [55] combining two ideas to produce an analytical representation of observation and a-posteriori target location uncertainty in a recursive filter framework with only three state parameters.

First, a bounded recursive filter proposed by Schweppe [111] is selected, in which uncertain target location, target process model noise, and observation error are all represented by an individual bounded ellipsoidal set

$$\tilde{\mathbf{\Sigma}} = \left\{ \mathbf{x} : [\mathbf{x} - \hat{\mathbf{x}}]^T \mathbf{\Sigma}^{-1} [\mathbf{x} - \hat{\mathbf{x}}] \leq 1 \right\}, \tag{4.6}$$

where $\mathbf{x}$ is a true state, $\hat{\mathbf{x}}$ is an estimate, and $\mathbf{\Sigma}^{-1}$ is a positive semi-definite matrix analogous to the covariance of a Gaussian distribution.

Then, for a stochastic linear system using such bounds, such as

$$\begin{aligned} \mathbf{x}_{k+1} &= \Phi \mathbf{x}_k + \mathbf{G}\mathbf{u} && (4.7) \\ \mathbf{z}_k &= \mathbf{H}\mathbf{x}_k + \mathbf{v}, && (4.8) \end{aligned}$$

where $\mathbf{\Phi}$ and $\mathbf{H}$ are transition and observation models and u and v represent process and observation noise having ellipsoidal bounds $\mathbf{Q}^{-1}$ and $\mathbf{R}^{-1}$ respectively, a recursive filter similar to the Kalman filter is defined as follows.

Predict:

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{\Phi}\hat{\mathbf{x}}_k \tag{4.9}$$

$$\mathbf{\Sigma}_{k+1}^- = \alpha_k [\mathbf{\Phi}\mathbf{\Sigma}_k \mathbf{\Phi}^T + \mathbf{Q}] \tag{4.10}$$

where $\alpha_k \in [1, 2]$.

Update:

$$\begin{aligned} \hat{\mathbf{x}}_k &= \hat{\mathbf{x}}_k^- + \rho_k [(\mathbf{\Sigma}_k^-)^{-1} && (4.11) \\ &\quad + \rho_k \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H}]^{-1} \mathbf{H}^T \mathbf{R}^{-1} [\mathbf{z}_k \\ &\quad - \mathbf{H}\hat{\mathbf{x}}_k^-] \end{aligned}$$

$$\mathbf{\Sigma}_k = \eta_k [(\mathbf{\Sigma}_k^-)^{-1} + \rho_k \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H}]^{-1} \tag{4.12}$$

$$\begin{aligned} \eta_k &= 1 + \rho_k - [\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-]^T [\rho_k^{-1} \mathbf{R} && (4.13) \\ &\quad + \mathbf{H}\mathbf{\Sigma}_k^- \mathbf{H}^T]^{-1} [\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-] \end{aligned}$$

for non-negative ρ_k.

Conceptually these equations use set summation and intersection to perform estimation. The key algebraic difference to the Kalman filter is two scalar parameters α_k and ρ_k that adjust the set volume produced by sum and intersection operations. Known guaranteed constant values ($\alpha_k = 2$ and $\rho_k = 0.5$) prove too conservative in practice, so in implementation a scalar convex optimization is performed at each time step to find the bounded estimate with minimum set volume, requiring only several

optimization iterations. Observations that do not intersect the predicted estimate are detected and rejected as outliers.

Next, the use of a higher-dimensional state space embedding is used as demonstrated by Hanebeck, [19] in which the 2D target geolocation estimate is mapped as

$$\mathbf{x}_{\text{tgt}}^{\text{world}} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \rightarrow \quad \mathbf{x}^{\star} = \begin{bmatrix} x \\ y \\ x^2 + y^2 \end{bmatrix}. \tag{4.14}$$

Interestingly, a wide range of non-linear sensing problems resulting in non-linear, non-convex, and non-simply-connected estimate sets may be expressed in this space. Specifically relevant for this problem is the ease of natively representing crescent-shaped polar uncertainties.

Applying this to the geolocation estimation problem, a range-bearing observation is taken as in Equation 4.4, and $\Sigma_{r\beta}$ from Equation 4.5 is decomposed for the following equations as

$$\mathbf{\Sigma}_{r\beta} = \begin{bmatrix} \sigma_r^2 & \sigma_r\sigma_\beta \\ \sigma_r\sigma_\beta & \sigma_\beta^2 \end{bmatrix}. \tag{4.15}$$

Then, for each range-bearing observation, an approximate observation in the over-parameterized space is constructed as in [120] by combining annular and cylindrical sets that capture range and bearing bounds (respectively) as follows and illustrated in Figure 4.14.

Range Bound:

$$\begin{aligned} \mathbf{z}_{range}^{\star} &= r^2 - (x_{\text{uav}}^2 + y_{\text{uav}}^2 - \sigma_r^2) \\ \mathbf{H}_{range}^{\star} &= [-2x_{\text{uav}} \ -2x_{\text{uav}} \ 1] \\ \mathbf{R}_{range}^{\star} &= (2r\sigma_r)^2 \end{aligned} \tag{4.16}$$

Bearing Bound:

$$\begin{aligned} \mathbf{z}_{circle} &= \begin{bmatrix} x_{\text{uav}} \\ y_{\text{uav}} \end{bmatrix} + r \begin{bmatrix} \cos\beta \\ \sin\beta \end{bmatrix} e^{-\sigma_\beta^2/2} \\ \mathbf{R}_{circle} &= (2r\sin(\sigma_\beta/2))^2 \mathbf{I}_{2x2} \end{aligned} \tag{4.17}$$

Example snapshots from a typical execution of this filter are provided in Figure 4.15, showing the rapid convergence to an accurate estimate with small error

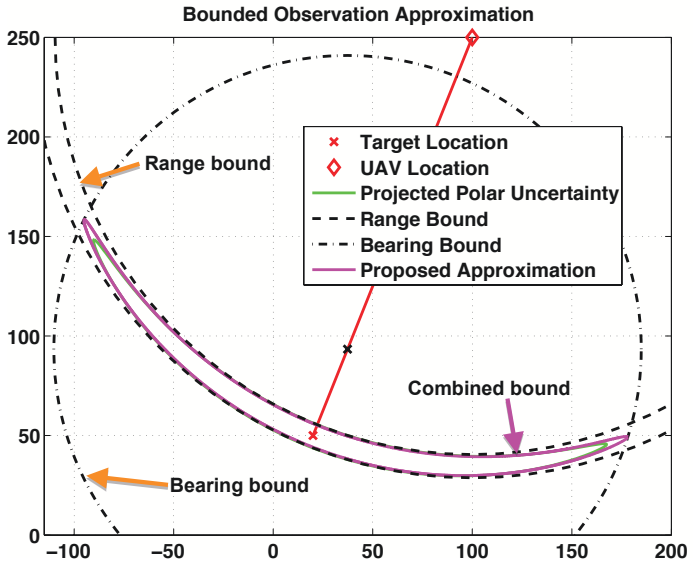


Figure 4.14: Construction of an approximate range-bearing observation uncertainty bound in the over-parameterized state space. Note the close match between Gaussian polar uncertainty and the combined approximate bound.

bound. While here too thorough comparison against these other developed filters and existing methods remains to be performed, a number of preliminary conclusions may be made.

Advantages this filter provides include rapid estimate convergence from even extremely high-uncertainty observations given its set-intersect nature, no need to assume any particular error distribution in vehicle state, and a compact representation with only three state parameters and associated ellipsoid bound matrix need be stored or broadcast to convey estimate state. The small state space and linear recursive filter form result in rapid prediction and observation updates, while estimate set membership testing (analogous to target location likelihood evaluation) is also extremely fast. This greatly simplifies and improves the tractability of decentralized information-sharing and observation-predictive planning in which hypothetical filter branching and target likelihood evaluations are frequently performed.

Downsides include the absolute criticality of the validity of uncertainty bounds on observations given the use of an intersection operation, as the integration of an invalid measurement will result in rapid, if not immediate, filter failure. Careful outlier detection and rejection is therefore vital. The use of highly conservative error bounds for each observation improves robustness but provides a tradeoff against convergence

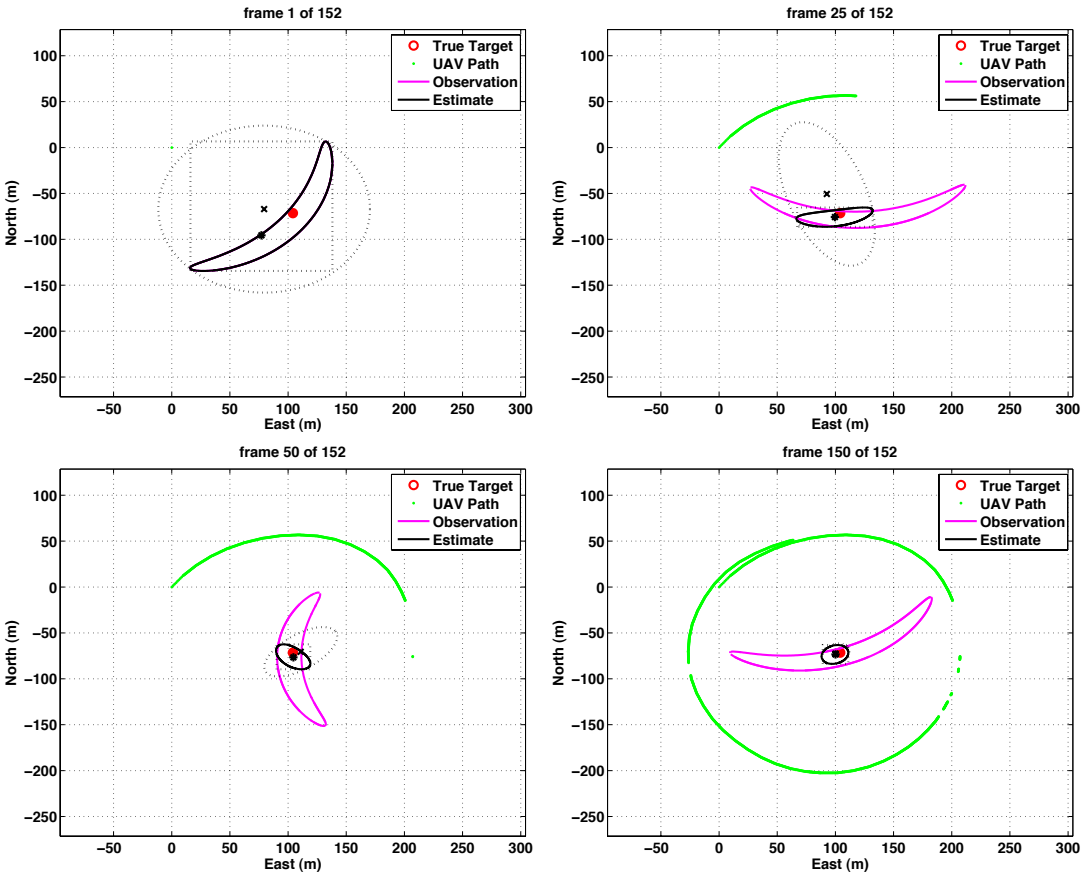


Figure 4.15: Sequence of snapshots during geolocation of a stationary target using the over-parameterized bounded filter. The dotted ellipse is a fast, conservative approximation of filter confidence; the dotted box is a more computationally intensive but tighter approximate confidence measure. The black crescent is a level-set of the true filter confidence estimate.

rate and steady-state confidence. Additionally, where needed, representing the set of 2D world locations lying within the higher-dimensional ellipsoidal bound is not completely straightforward. Membership testing (Equation 4.6) is fast to compute, and implicit level curves of the ellipsoidal projection suitable for visualization are conveniently a quartic polynomial having an analytic solution. For parametric representation, two options include simple geometric projection of the ellipsoid to two dimensions, providing a very fast but (possibly uselessly) over-conservative bound, and computing a smallest axis-aligned bounding box containing all components of the estimate set, which though substantially slower to compute provides a much tighter approximation. Examples of these options are depicted in Figure 4.15. Finally, a potentially serious limitation is that in order to retain the linearity of the model, only stationary or constant-velocity targets are representable. It remains an unanswered question as to whether EKF-style linearization that would enable arbitrary target process models retains numerical stability and acceptable levels of performance.

4.3 Pursuit of Constrained Targets

The proposed methods for improved geolocation reduce estimate error and well model the nature of observation uncertainty found in small-UAV geolocation, but as they may still require a large number of observations to produce such an estimate, inadequacy for pursuit purposes remains. Further, given inevitable long-duration dropouts contemplated in Section 3.3.3 as a UAV circles back to an overflown target and limited onboard computation with which to consider best trajectories for possible target locations over a wide area, higher-certainty target prediction from a previously estimated location is needed. Together, these motivate a desire to limit both the space of locations in which target presence need be considered and the set of locations within this space to which a target can move at a given time. The means proposed in this thesis for accomplishing this is the use of pre-existing knowledge of environmental constraints producing such simplification.

4.3.1 Single-road pursuit

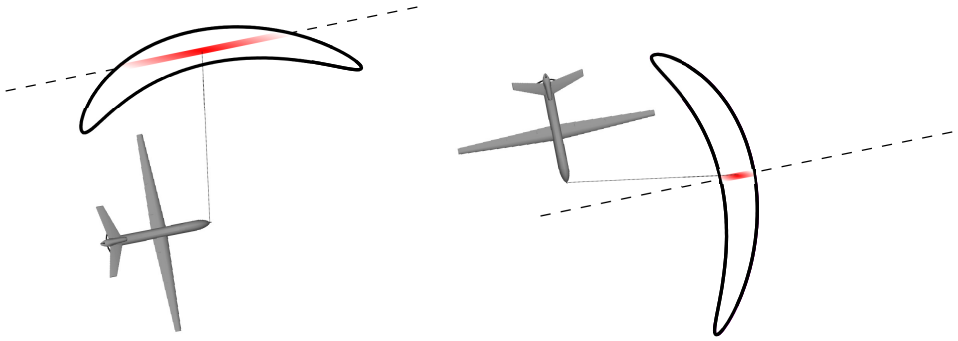
A simple but instructive and still highly relevant case to first consider is that of a target known to be moving along a single road segment. Given an existing map indicating the path of the road, the geolocation problem is now reduced to one dimension, regardless of the (possibly arbitrarily winding) actual path of the road. For the purpose of this exploration, a road represented as a contiguous sequence of linear line segments is assumed, as well as some means to choose the correct

line segment given an observation (such as choosing the nearest one to the raw observation). Then, the problem is further reduced to the case of a series of raw observations and a single line segment.

As noted in Section 3.3.2, road-constrained geolocation has been previously considered extensively in tracking with high-altitude radar, and an enumeration of general benefits and distinctions from this context is warranted. The primary known benefit is simply greater geolocation accuracy, stemming from two sources: better accuracy from filtering of observations and improved prediction in the absence of observations. The first is due largely to the fact that relative to the surrounding open area, the set of possible target locations is much smaller (being one rather than two dimensional, the number of samples required for a uniform discrete sampling grows linearly rather than quadratically with the size of the region), such that even a completely uninformed uncertainty distribution occupies a physically small area. Better filter accuracy is also due to more accurate raw observations, which if also constrained to one dimension, likewise eliminate an entire uncertainty dimension. Meanwhile, the second source of overall geolocation accuracy–improved prediction–results from a similar dimensionality reduction in the space of target actions or destinations, in discrete form again having linear rather than quadratic growth with region size. The practical impact of such improved prediction is that reduced observation frequency (and, for moving targets, fewer additional observations more than stationary targets) is required.

In the context of target pursuit with field-reconnaissance UAVs, however, road-constrained geolocation provides additional practical benefits that have not been considered in the aforementioned constrained estimation literature, primarily due to the mobile nature and limited field of view of sensors in this context. Clearly, overall improvement in geolocation accuracy fulfills an immediate CSAT mission goal. Given freedom in sensor placement not typically found in radar sensing applications, specific advantage of the target state space reduction may be taken so that the larger dimension of asymmetric observation uncertainty (as from large heading uncertainty) is aligned with the perpendicular axis of the road as frequently as possible and thereby eliminated, drastically improving raw observation quality. This is illustrated in Figure 4.16. Relatedly, the combined effect of improved estimate accuracy and improved predictiveness implies better robustness to observation dropouts during reacquisition of overflown targets, since the uncertainty region for reacquisition will be small. Taken together, all of these benefits improve task performance, but for a key reason specific to mobile sensing: the characteristic of having essentially a positive feedback loop on target location certainty, as a high-confidence location provides confident future sensor placement, whereas an uncertain location risks poor

future placement. In terms of observable impact, such performance improvement translates to more accurate geolocation estimates and reduced frequency of target loss.



(a) Road-parallel observation (b) Road-perpendicular observation

Figure 4.16: Illustration of relative value of placing a sensor footprint with asymmetric observation uncertainty (here, due to large heading uncertainty) parallel (a) or perpendicular (b) to the road vector. A likelihood bound (Gaussian level curve) for each observation is depicted as a black crescent, and one-dimensional projected observation uncertainty is shown in red.

In addition to performance, constrained geolocation also provides a drastic improvement in scalability. As both the spaces of target location and target actions are reduced, a radically simpler observation-predicting trajectory planning problem results through a reduced integration volume for Equation 3.6. Practically, when evaluating the value of placing the sensor footprint at a given position in the future (a single term of this integral), a much smaller set of possible observations need be tested or branched on. Indeed, if sampling a discrete set of possible observations, once again a number linear rather than quadratic in the size of the target region is needed. Additionally, reduced filter state dimension necessarily results in reduced memory and computational requirements when cloning current filter state and applying hypothetical target predictions and observation updates during UAV action search for planning. Altogether, such scalability improvement permits longer-horizon trajectory planning and more tractable increases in UAV team size.

To demonstrate and evaluate these benefits, a simulation study was performed [31] comparing the behavior of several filters and trajectory planning. First, given a road

segment characterized by a line segment $\mathbf{p}_{\text{road}} + \hat{\mathbf{v}}_{\text{road}} t$ (for a 2D point $\mathbf{p}_{\text{road}}$ and unit vector $\hat{\mathbf{v}}_{\text{road}}$), a raw pseudo-observation $\mathbf{x}_{\text{tgt}}^{\text{world}}$ having Gaussian uncertainty with covariance $\Sigma_{\text{tgt}}^{\text{world}}$ is projected to the road as a new scalar pseudo-observation

$$x = (\mathbf{x}_{\text{tgt}}^{\text{world}} - p_{\text{road}}) \cdot \hat{\mathbf{v}}_{\text{road}} \tag{4.18}$$

having Gaussian uncertainty with scalar variance

$$\sigma^2 = \hat{\mathbf{v}}'_{\text{road}} \Sigma_{\text{tgt}}^{\text{world}} \hat{\mathbf{v}}_{\text{road}}. \tag{4.19}$$

Generation of the scalar observation is merely the geometric projection of a point onto a line, while variance is derived from Jacobian covariance transformation of the projection function and is exact because the projection is a linear function. Filter modification to incorporate the road constraint depends on its internal representation: for Kalman filters maintaining a Gaussian estimate, only a 1D Gaussian is used; grid representations are modified to only provide cells along the length of the road; and particle filters sample points only along the road's line segment. Trajectory planning for pursuit may be performed by any existing means, except that where possible observations need be enumerated or sampled, far fewer are needed. In experimentation here, sampling 1D Gaussians with 7 points proved adequate.

In the simulation scenario considered, the task is defined as pursuing a vehicle moving along a 2.5km long roadway using a single UAV. The target's nominal speed is 4.5m/s (10mph), and to roughly simulate evasiveness or confusion, it accelerates at up to 5.4m/s^2 (equivalent to accelerating from rest to 60mph in 5s), within the range 3.4 to 6.8m/s (5-15mph). This is roughly as fast of a target that a UAV traveling at 15m/s (33.6mph) can pursue while retaining any substantial maneuverability. Coordinated turns are enforced, and control commands received by the UAV are low-pass filtered with a cutoff frequency of 2Hz to somewhat simulate an autopilot optimizing for smooth flight. Uncertainties are modeled as uncoupled Gaussian uncertainties with standard deviations of 3m in lateral position, 5m in altitude, 3 degrees for roll and pitch, and 5 degrees in yaw. Camera parameters received uncertainties of several percent. For simulation, errors in measured values were drawn from these distributions. For consistent comparison, a trial is defined as 5 minutes of simulated elapsed time, during which time the target has moved approximately two thirds of the length of the roadway. Trials are stopped if a filter loses tracking (diverges or reports an incorrect estimate resulting in an irrecoverable trajectory) and logged as such. Each result presented is the average over several hundred trials with randomized initial conditions that place the target within the UAV's view, as though it had just detected the target during a search. An example visualization from the simulation is shown in Figure 4.17.

Figure 4.17: Visualization example from simulation showing a UAV pursuing a Jeep. The green trapezoid is the ground region visible to the camera. The orange dot indicates the geolocation filter's best estimate of the target's position at the center of the red smear denoting the surrounding uncertainty distribution.

As a baseline comparison, four unconstrained filters were implemented and evaluated: a constant-velocity model Kalman filter using pseudo-observations with uncertainty generated by Jacobian linearization, a constant-velocity model Kalman filter using pseudo-observations with uncertainty generated by the Unscented Transform, a fixed-cell (1m in size) discrete Bayesian filter using a diffusing constant-position target model and pseudo-observations with uncertainty generated by the Unscented Transform, and the particle filter described in Section 4.2.2 using a constant-velocity target model and range-bearing pseudo-observations and corresponding Gaussian uncertainty. For the first set of experiments, the very simple control law of directing the UAV to orbit the filter's best estimate is used. Results summarized in Table 4.2 suggest first that a better strategy is sorely needed given the high rate of target loss (avoided by the Bayes filter given its large cells and slow diffusion). As expected, filtering using the Unscented Transform performs slightly better, and the Bayes grid filter performs slightly worse due to its low spatial resolution. The particle filter performed particularly well when it worked, but it suffered frequent particle collapse despite stratified resampling and outlier rejection (that may be improved upon by greatly increasing the particle set size and unreasonable scenario-specific process model tuning).

The same set of filters were again compared, this time incorporating the road constraint, and once again applying the simple control law of orbiting the best estimate. Results summarized in Table 4.3 show marked improvement in target loss rate and moderate improvement in accuracy. The particle filter continues to exhibit substantial fragility.

Finally, this experiment was repeated yet again while now applying observation-

	KF + JT	KF + UT	Discrete Bayes	PF + RB JT
Trials losing target	27.0%	20.5%	0%	65.0%
Time in view	98.1%	97.5%	96.5%	98.3%
Avg view loss duration	1.8s	2.4s	1.6s	1.3s
Avg target error	10.5m	9.8m	15.5m	8.6m

Table 4.2: Simulation results for baseline filter comparison while orbiting the mean target location reported by each estimator.

	KF + JT	KF + UT	Discrete Bayes	PF + RB JT
Trials losing target	10.9%	9.5%	0%	20.5%
Time in view	97.4%	97.2%	95.9%	97.7%
Avg view loss duration	3.0s	3.3s	0.9s	2.1s
Avg target error	7.5m	7.4m	15.1m	7.9m

Table 4.3: Simulation results for road-constrained filter comparison while orbiting the mean target location reported by each estimator.

optimizing trajectory planning. For this, only the Kalman filter using uncertainty generated by the Unscented Transform was selected, as the overall best performing filter in the preceding experiments. Here, two optimization metrics were evaluated: maximizing the average probability of viewing the target over the course of the trajectory, and minimizing a-posteriori filter uncertainty. For simplicity, trajectory optimization was performed by brute-force breadth-first search in the action space of steering angles (held for 2s between action branches). For both the depth-1 (greedy) and depth-3 optimizations, planning in realtime provided no difficulty. Results summarized in Table 4.4 expose several interesting observations. First, unsurprisingly, greater look-ahead improves performance, however it does not do so by an immense margin for a moderate increase in depth, implying that given for instance a constrained onboard processor, it may be reasonable to use single-step look-ahead for at least this specific task. This is plausible because the growth in target location uncertainty during long-term look-ahead produces an uncertainty distribution that is flatter, and hence less informative. Particularly interestingly, one may observe that optimizing for even short-term minimum uncertainty produces better performance (primarily a much lower target loss proportion) than optimizing for more likely observations out to a greater horizon. This highlights the importance and power of accurate target motion prediction, as reduction of uncertainty (and hence, presumably error) in target velocity greatly aids prediction when it is not being viewed, ever

more importantly for a target moving at a relatively consistent speed for even brief periods such as present in this scenario.

	Min Posterior Uncertainty		Max Viewing Likelihood	
	Depth 1	Depth 3	Depth 1	Depth 3
Trials losing target	2.5%	0.8%	15.0%	4.5%
Time in view	80.9%	88.4%	77.1%	86.6%
Avg view loss duration	3.6s	1.4s	1.6s	1.0s
Avg target error	7.3m	6.6m	12.1m	9.7m

Table 4.4: Simulation results for the road-constrained Kalman filter taking observations with Unscented Transform generated uncertainty, comparing search depths and metrics for trajectory planning.

As a useful direct illustration of the impact of the application of the road constraint, Figure 4.18 compares the overall trajectory of a UAV performing depth-3, posterior uncertainty minimizing planning with the constraint (as in the last simulation experiment) and without (using two-dimensional Gaussian filter uncertainty) it. The latter exhibits moderate elongation of portions perpendicular to the road, indicating a larger fraction of the total trajectory is spent collecting observations benefiting from the constraint and providing greater certainty.

4.3.2 Cellular road-network decomposition

Simply extending road segment pursuit to more complex environments, such as a winding road with segments looping near one another or networks of roads as would likely be found in a common urban environment, is not practically straightforward given that the assumption of having a means to correctly select the true segment on which the target lies is unrealistic if ambiguity is presented by observation uncertainty overlapping multiple nearby segments. Additionally, if junctions providing multiple possible directions for a target to move are introduced, multi-modal prediction uncertainty results.

One possible means to address this is to use a cellular Bayesian estimator (used as a compared filter in the preceding experiments) and plan trajectories maximizing the average probability of observation, essentially executing a continuous local probabilistic search, previously referred to in Section 3.3 and described further in Section 5.2. Pursuit in this representation primarily differs from search in that a single target is explicitly assumed for a given estimator, and the probability across all cells is assumed to be normalized. If target motion between cells is predicted

(a)

(b)

Figure 4.18: Comparison of a typical overall UAV trajectory having pursued a moving vehicle as in the preceding experiments without (a) and with (b) application of the road constraint, each performing trajectory planning to minimize expected short-term a-posteriori geolocation filter uncertainty. Although superficially quite similar, the trajectory using the road constraint makes approximately 22% fewer orbits in the same interval and exhibits approximately 20% greater elongation of each orbit (corresponding, very roughly, to that proportion greater viewing time from poses perpendicular to the road). Intuitively, this is due to attempts to acquire low-uncertainty observations, which as depicted in Figure 4.16 are more likely as the UAV moves along the perpendicular direction to the road, but which appear to be equally possible from any viewing direction if knowledge of the road is unavailable.

during planning, the result is pursuit with the goal of maximizing target observation frequency. An alternative metric that may be applied is to minimize expected a-posteriori uncertainty defined using the discrete weighted sample covariance over all cells.

Figure 4.19 provides an exploratory simulation example of live pursuit using such

a cellular representation. Here, probability of observation is maximized using single-step (greedy) steering direction control, Gaussian diffusion informed by an assumed maximum speed is applied as prediction between observations, and for simulation the target executes a random policy at junctions, otherwise accelerating randomly as in the preceding experiments. In this example, a target moving along a straight segment slows briefly, requiring that the UAV turn to prevent overflying it. In the process, the target continues moving and leaves the field of view. Until its reacquisition, the filter smoothly tracks diffusion into adjacent road segments and incorporates negative observations as otherwise-likely locations are found to not contain the target. Upon detecting the target, the probability of cells rapidly re-aligns to a small uncertainty region around the true location.

Figure 4.19: An example of pursuit in a road network using cellular decomposition. The UAV anticipates loss of view (area within the green trapezoid) and begins to turn sharply. While out of view, it predicts diffusion in possible target location (red probability mass) and greedily steers to maximize the probability of re-detection.

To a large extent, then, such a cellular decomposition provides a total solution to pursuit in complex environments. Most importantly, it naturally handles multi-modality that results from an observation ambiguously lying across multiple road segments–cells along each simply receive a high observation likelihood–as well as that resulting from predicting target motion near a junction–adjacent cells lying on connecting road segments will received increase likelihood through diffusion. It also naturally presents a solution to the subtasks of recapture (and, if required, search) since if a target is not re-acquired before it could have escaped into several distinct regions that cannot be swept in a single continuous motion, it provides a means to track remaining likely locations as each is visited.

At the same time, however, cellular decomposition is completely impractical for optimizing all but very coarse approximations of Equation 3.6. A large number of cells (each constituting a hypothesis) produces a large number of possible observations over which to take expectation or branch for planning. Likewise, the complexity

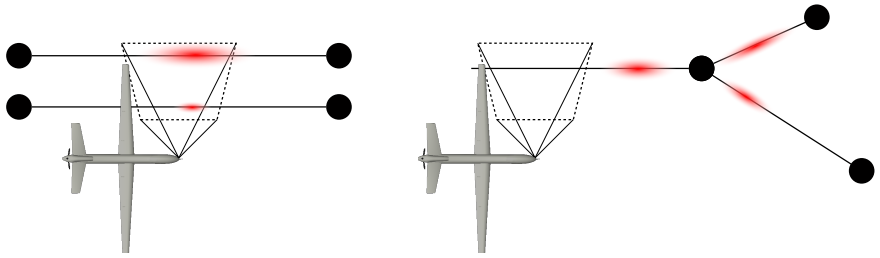
of motion prediction and observation updates grows at least linearly with the number of cells. Further, a large number of cells *is* required, for several reasons: large cells provide poor geolocation resolution, a cell size at least on the order of possible inter-timestep target motion distance is necessary for meaningful discrete-time approximation of its motion, and additional dimensions for target speed or direction are needed for stateful estimation beyond unrealistic random motion. As a result, optimization by branching on hypothetical observations entails a large branching factor, substantial computation for applying these observations and ensuing target motion prediction, and copying a large amount of data to clone filter state. Worse, the typical simplification of assuming only negative observations will be received is inapplicable to an active pursuit scenario, as the particular impact of hypothetical positive observations on uncertainty must be considered. Therefore, drastic measures such as assuming a stationary target probability distribution or planning to very short horizons are necessary. As an example, the simulation of Figure 4.19 utilized both these measures (greedy planning assuming a stationary target) and still provided considerably slower than realtime execution.

To overcome this, parametric representation of a road network would be ideal, and the following subsection proposes building upon the same existing constrained and multi-modal estimation literature from which the preceding road segment pursuit representation derives inspiration to cast complex road network pursuit as pursuit over a discrete collection of single road segments. In doing so, it is anticipated that similar performance benefits might be attained while motivating further exploitation of fundamental environment topology.

4.3.3 Extension to continuous road networks

Clearly, direct extension of road-constrained parametric estimation representations as used in Section 4.3.1 for single-segment pursuit is desirable. However, these do not immediately generalize to road networks for several reasons. The two main, and indeed very common, sources of difficulty are shown in Figure 4.20. In the first, an observation with relatively high uncertainty may overlap more than one road segment, with differing likelihood of association, and an increase in target presence probability must be applied to both. In the second, particularly when out of view, a target location distribution may approach an intersection of roads, and the possibility of the target choosing each of the branches must be modeled, by splitting the distribution. An additional complexity not shown is that of merging distributions that may collide either within a single edge or at a junction joining two segments.

Together, these naturally suggest the use of some form of multiple-hypothesis



(a) Ambiguous observations (b) Prediction through junctions

Figure 4.20: Examples of the two typical sources of multiple hypotheses in road-network-constrained parametric estimation.

estimation, and indeed principles of existing constrained and multi-model estimation used in existing work in geolocation from high-altitude radar reviewed in Section 3.3.2 may be applied to reap specific benefits for small-UAV pursuit. Specifically, as already implied, road-segment-constrained estimation may be implemented in a multi-model framework such that a one-dimensional geolocation estimator is run for several road segments in parallel each having a relative likelihood of being the correct hypothesis (each defined as a model in this context), observations received are applied to each filter, relative likelihoods are recomputed given each filter's agreement with each observation, and an overall best estimate is derived through a likelihood-weighted average of each estimator's mean. A common means for accomplishing this generically is via the Interacting Multi-Model (IMM) filter, [11] implemented in GMTI radar applications as the variable structure IMM (VS-IMM). [72]

Alternatively, this may be implemented as a particle filter by any of several means referenced in Section 3.3.2, such as either running a distinct particle filter for each model wrapped by an overall IMM filter as above, using one single set of particles constrained to the graph in which the process model produces multi-model behavior by choosing an outgoing road for each particle at junctions, or through an unconstrained set of particles (having two-dimensional state) on which a combined observation likelihood is produced through a weighted sum over models. These too have been previously explored in a radar tracking context. [37]

However implemented, the result is a small number of hypotheses—represented as a set of road-constrained filter estimates or particles lying on one or more road segments—each individually deriving the accuracy benefits of a road-constrained filter. If pursuit is performed using an objective function branching on or taking expectation over this set of hypotheses, only a linear factor larger (by the size of this set) additional computation is required by a fixed hypothesis set over a single

segment, and in practice unpruned hypotheses over only up to several road segments at a time seem likely. During trajectory planning, the expected hypothesis set grows exponentially with the number of junctions that may be reached by a target within the horizon considered, but for any typical environment with road segments at least one hundred meters long, this number too is small. Thus, the result is a tractable means to retain the accuracy and robustness provided by road-segment-constrained pursuit while expanding to larger environments.

In the interest of scope, implementation and analysis of such methods are not pursued in this thesis. Doing so represents an interesting area of work, however, since though these have seen much study as generic multi-hypothesis estimation and application in target tracking from aircraft, little if any study has been performed in scenarios in which the output of such estimators are used to direct control actions (and thereby future observations). Typically, observations have been generally treated as incidental output from pre-planned or operator-provided paths, whereas here the choice of path depends on the quality of the current estimate and dictates, for better or worse, the likelihood and relative target-observer pose of future observations.

4.4 Conclusions and Future Work

A number of conclusions may be drawn from both study of related work and field and simulation experience thus far. First, with appropriate filtering, even highly uncertain observations may be used to derive an accurate instantaneous target location estimate, but many may be required to do so. Rapidly changing target state (as for agile moving targets) or estimation of higher-order states such as velocity (as observations provide only direct observability in location space) worsen this considerably, resulting in for instance particularly poor accuracy for freely moving targets in open areas. Meanwhile, observation-predictive trajectory planning is hindered where many possible observations exist, due either to high-dimensional target uncertainty, uncertainty simply spread over a wide area, or unconstrained target models permitting rapid divergence in target location. The practical effect of this is immense computational complexity when optimizing trajectories for expected observations. For target pursuit, limited UAV maneuverability (such as a long turn-around time) implies a need for many UAVs or highly accurate predictions to prevent target loss, as without these, the result is poor robustness to periodic observation drop-outs of an individual UAV. For search having the goal of guaranteed capture, large environments or fast targets result in rapid recontamination requiring many UAVs, imposing high resource demands and poor scalability. As a more positive observation, the use

of environment structure to provide additional information about possible target position, motion, or capability has proven fruitful, motivating its use wherever possible. Succeeding chapters attempt to specifically overcome the aforementioned limitations in the limited but common scenario of road networks providing such structure.

CHAPTER 5

Coverage and Efficient Search

A common task for aerial observers is that of mapping an area or searching for a target that may be assumed to be non-adversarial (that is, non-cooperative but oblivious). This chapter explores several means to do so that are both well-suited to small UAVs and operate in a time-optimizing—minimizing total path length—manner to provide coverage or detection as quickly as possible. For this reason, these are also referred to as "efficient search" methods.

One simple but highly-effective algorithmic extension providing a practical means of area coverage search is conveyed in Section 5.1, which though most applicable to open areas rather than road networks, is particularly optimized for the class of small UAVs considered and inspires the use of existing motion-primitives such as orbiting of points provided by high-level commercial autopilots as a means to build more complex behaviors. Application of classical recursive Bayesian estimation for probabilistic road network search is briefly explored in Section 5.2, along with a field-of-view approximation providing substantial planning performance increases with minimal task performance degredation. Finally, a time-minimizing total-coverage strategy for road networks built upon several variations of the Traveling Salesman Problem is presented in Section 5.3.

5.1 Limited-Maneuverability Open Area Coverage

When performing an area coverage search, a particular challenge arises in the detection or designation of targets in that a sufficiently clear image must be observed for a sufficient duration. This is difficult for the class of UAVs considered for several reasons. First, noisy or lossy video may require a number of viewings for a clear, recognizable image. Simultaneously, human operators are necessarily slow to react to seeing something of interest and designating it. Meanwhile, viewing from multiple angles might be necessary for automatic or human recognition of a target either given possibly asymmetric placement of prominent object features or unanticipated directional environmental occlusions. On top of this, if an object of potential interest is perceived but passed by, a lengthy vehicle turnaround time to return for re-inspection may be necessary given a large turning radius.

Therefore, a practical form of area coverage search was developed as a means to provide temporally-contiguous redundant observations of every point in an environment from multiple viewing directions. This is accomplished through the use of a coverage search based upon orbit motions rather than the more common use of linear sweeping described in Section 3.4.1. This search strategy requires only that an onboard autopilot provide an orbit-point control mode in which a side-looking camera is centered on the instantaneous orbit point center and is thus quite appropriate for small field-reconnaissance UAVs, unlike some of these aforementioned methods that require high-frequency steering updates. It is very similar in this sense to the strategy proposed by Goodrich et al. [54] that continuously directs the orbit point issued to an existing autopilot in an outward spiral to provide uniform coverage of an area.

Given a boustrophedon decomposition [24] of a polygonal search area $\mathcal{P}$ such as that in Figure 5.1 and a distance-optimal ordering of cell visitation, rather than directing the UAV to follow the plowing motions directly itself, the autopilot is fed a continuous stream of new desired orbit point (and hence sensor footprint) centers moving along this path. In practice, since a uniform area search suitable for a static target is unlikely to be particularly time sensitive given that the target must not be evasively escaping, all but large environment obstacles may be eliminated, greatly simplifying the decomposition and cell ordering problem. As an additional measure, obstacles causing plow segments to be shorter than two sensor footprint diameters are reduced or eliminated.

The "plow width" is chosen to be slightly less than the diameter of the circular area that would be covered by the rotating sensor footprint during a single orbit of a stationary point for slight overlap between adjacent path sweeps and thereby

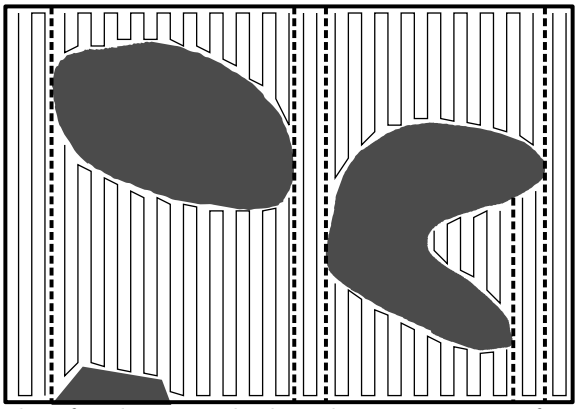


Figure 5.1: An example of a boustrophedon decomposition of an area containing two internal obstacles and an obstacle adjacent an area border, along with a possible coverage path through each region.

complete area coverage. Both the plow width and (to a greater extent) precession rate at which the orbit point is moved along the path are tunable parameters controlling the level of coverage redundancy desired. Nominally, a precession rate of one sensor footprint diameter per time taken to complete an orbit assures coverage of every point from all viewing directions. New orbit points are provided to the autopilot at a rate of one every three seconds to ensure that the underlying orbit controller maximizes the time spent in its steady-state control region. A typical simple area search example is depicted in Figure 5.2, with examples of varying coverage redundancy parameters compared in Figure 5.3.

As stated, the resulting trajectory is parameterized by time, resulting in a fixed path for the desired orbit point, independent of the UAV's actual success in maintaining the intended orbit trajectories. In the presence of high wind, segments of a given orbit may take longer than nominal as the UAV faces a head-wind that reduces its ground-speed, while others less time given a tail-wind providing increased speed. Additionally, side-winds may induce trajectory tracking error that requires additional time to overcome. Therefore, an alternative parameterization providing some level of robustness is to continuously track actual angular position around each orbit and compute the matching point along the orbit point coverage path for the nominal case. Thus, in the extreme case, a hovering UAV paralyzed by wind would be perpetually provided an unchanging orbit point.

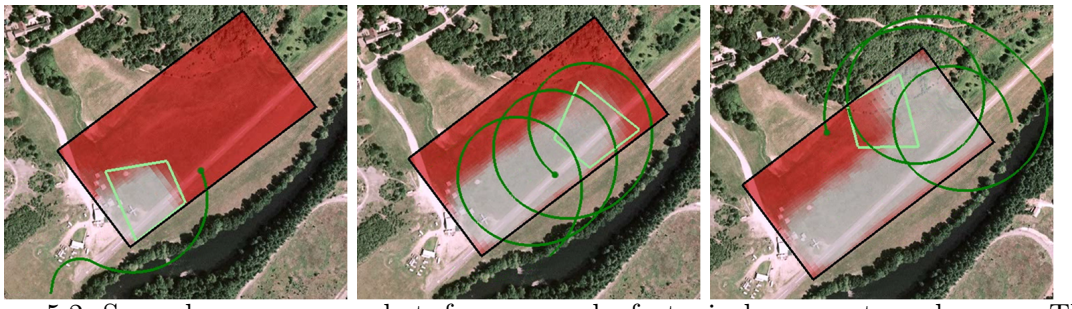
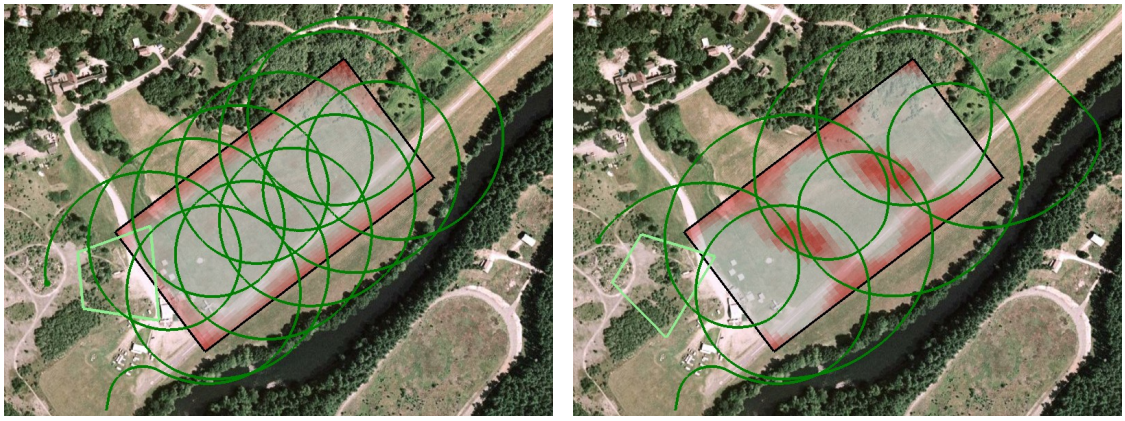


Figure 5.2: Several coverage snapshots from a search of a typical open rectangular area. The level of coverage is indicated by red shading, with completely uncovered areas displayed in the darkest shade of red, and sufficiently (exceeding an adjustable threshold on probability of detection) covered areas as white. The path of the UAV is drawn in dark green, its current location marked by a green dot, and the current sensor footprint as a light-green trapezoid.



(a) High-redundancy coverage (b) Speed-optimized coverage

Figure 5.3: A comparison of two extremes in the resulting coverage pattern during a search, using (a) a high-redundancy (tight overlap) search and (b) a speed-optimized (low overlap) search providing possibly insufficient coverage of several interior regions. The level of coverage redundancy may be varied by changing a single parameter, the orbit-center precession rate. Colors and markings are as defined in Figure 5.2.

5.2 Probabilistic Search in Road Networks

As first introduced in Section 3.3, the probability distribution of one or more targets may be represented by some discrete means such as the density of particle placement or the value contained in cells tessellating areas of interest. Search strategies may then be produced by planning paths through the environment whose observations, in expectation, optimize some metric such as maximal likelihood of target observation, minimal time to target observation, or maximize a-posteriori certainty in target location (e.g. by minimizing distribution variance), each of which have subtly different definition. Typically, this is known as information-theoretic control applied to classical recursive Bayesian estimation and has received extensive study in various forms, several of which are cited in aforementioned related work. A particularly relevant instance is that proposed by Ryan, et al. [106] (practical implementation of which is discussed further by Tisdale, et al. [124]). This concept is fundamentally identical to that suggested for pursuit in Section 4.3.2 except that if desired, the probability of individual cells may treated independently to represent the possible presence of an arbitrary number of targets, whereas for a single target they may be normalized to sum to one so that each indicates the *relative* probability of target presence. As noted in that section, trajectory planning with such an underlying estimator

This section does not seek to advance the state of the art in discrete Bayesian estimation or information-theoretic planning methods applied thereto. Rather, it highlights a key difficulty in the execution of such approaches and evaluates an approximation that may be applied to greatly improve their performance as a demonstration of a means by which approaches such as these that might otherwise be easily dismissed for their computational intractability or planning myopia may be made useful in practice. Further, it demonstrates that such existing methods may be readily applied to environments comprised of road networks.

5.2.1 Classical formulation

For convenience of discussion, a typical instance of this formulation is now stated, with slight simplification.

To form the discrete estimator, a set of cells $\mathbf{X} = \{x_0, \ldots, x_n\}$ is defined for the entire environment, and for brevity the term $P(x_i)$ is used to indicate the probability of the target lying in cell i. In application to road networks, a cellular discretization identical to that suggested in Section 4.3.2 may be applied: road segments may be divided along their length at specified intervals to produce a tessellation. Given the generality of this representation, additional pockets of open areas may likewise

receive coverage via any desirable tessellation—rectangular, triangular, hexagonal, etc.

A series of observations $\mathbf{Z}$ is received, of which the current one may be just referred to as z. The observation z may take one of two forms: a positive observation with an associated pixel location in camera image coordinates (or some equivalent local coordinates for an alternative sensor) at which a target was observed, or a negative observation indicating that no target was detected in that batch of sensor data. A simple detection model including false-positive and true-positive detection probabilities is often applied, stated as $P(detect|\neg visible)$ and $P(detect|visible)$ respectively. Roughly speaking, with each observation, $P(x_i)$ is updated using a simple application of Bayes' rule,

$$P(x_i|z) = \frac{P(x_i, z)}{P(z)} = \frac{P(z|x_i)P(x_i)}{P(z)}. \tag{5.1}$$

This is used to produce the post-update value of a cell, the term at left, using known or computable quantities on the right. $P(x_i)$ is simply the prior value of the cell, $P(z)$ is a normalization constant that is the same for all cells (and as such may be neglected if all cells are to be updated, if their sum is renormalized to one post-update), and $P(z|x_i)$ is the observation model—the likelihood of the observation received if the target indeed lay in cell i—and is the single interesting term.

For a positive observation at pixel (u, v), the model may be defined as

$$\begin{aligned} P(detect@(u,v)|x_i) &= P(detect@(u,v)|visible)P(visible|x_i, pose)+ \\ &\quad P(detect@(u,v)|\neg visible)P(\neg visible|x_i, pose) \\ &= P(detect|(u,v)) \cdot \\ &\quad P((u,v)|x_i, pose, visible) \cdot \\ &\quad P(visible, x_i, pose)+ \\ &\quad P(detect@(u,v)|\neg visible) \cdot \\ &\quad (1 - P(visible|x_i, pose)), \end{aligned} \tag{5.2}$$

and for a negative observation as

$$\begin{aligned} P(\neg detect|x_i) &= P(\neg detect|visible)P(visible|x_i, pose)+ \\ &\quad P(\neg detect|\neg visible)P(\neg visible|x_i, pose) \\ &= P(\neg detect|visible)+ \\ &\quad (1 - P(detect|\neg visible)) \cdot (1 - P(visible|x_i, pose)). \end{aligned} \tag{5.3}$$

All terms except for $P((u,v)|x_i, pose, visible)$ (a camera projection uncertainty model) are either from the detection model or the term $P(visible|x_i, pose)$, which answers the vital question: is cell i in view when the UAV is at a given $pose$.

5.2.2 Field-of-view approximation

Naively, one determines whether cell i is in view by projecting its location into local sensor coordinates given an estimate of UAV pose. If all values were known with total certainty, this would be a binary fact, but in practice uncertainty in UAV pose, sensor mounting, and sensor parameters (e.g. camera calibration) are known with uncertainty and induce uncertainty in the result of this projection. In common implementation, linearized Gaussian uncertainty propagation as first stated in Equation 3.4 is used to generate a Gaussian in sensor (e.g. image) coordinates over which a CDF within the visible area (e.g. the image extent) may be taken to produce a visibility probability, $P((u,v)|x_i, pose, visible)$. As the Jacobian is a function of the cell location, this procedure must be performed for each cell. Though usually tolerable for live estimation purposes, it is intractable to repeat this computation for hypothetical map updates or even mere visible-probability-summation when evaluating candidate trajectories for path planning.

Instead, it is proposed that rather than iterating over all cells, projecting each into sensor coordinates, and evaluating the probability of visibility, instead the field of view may itself be represented once overlaid upon the map as a probability density, with each cell evaluated against this density to determine likelihood of visibility. Intuitively, this distribution will have weaker probability near its edges than its center, as points near the center would still be in view if the UAV's pose were perturbed, while points at its boundary may not be. As the distribution for the field of view may be quite complicated, particularly in the presence of vehicle roll or a gimbaled sensor, it must be approximated, most conveniently as a kernel centered at the ground look-at point (center of field of view). Stated this way, the probability of target detection is then merely the convolution of this kernel with the probability map evaluated at the look-at center, which may greatly simplify trajectory metrics attempting to maximize short-term detection probability.

Three kernels, shown in Figure 5.4, each providing particular intuitive properties were evaluated. The first is a simple box-kernel representing a hard binary bound on field of view: cells within the kernel's area are considered in view with probability one, and zero otherwise. This provides very simple computation but does not account for reduced visibility likelihood near borders. A second kernel is a truncated Gaussian distribution, which roughly captures higher probability at the center than

borders. The third is an Epanechnikov kernel, commonly used in mean-shift visual tracking [27], one method used in field experimentation to produce geolocation observations described in Appendix A. Roughly, this is an inverted quadratic with sharper fall-off to zero at the boundaries while still weighting the center more highly. One final note is that as described, kernels are radially symmetric (square the box-kernel) and are elongated (eccentricity is added) along the appropriate direction to better fit the field of view. As necessary, this step might be neglected to further reduce computation.

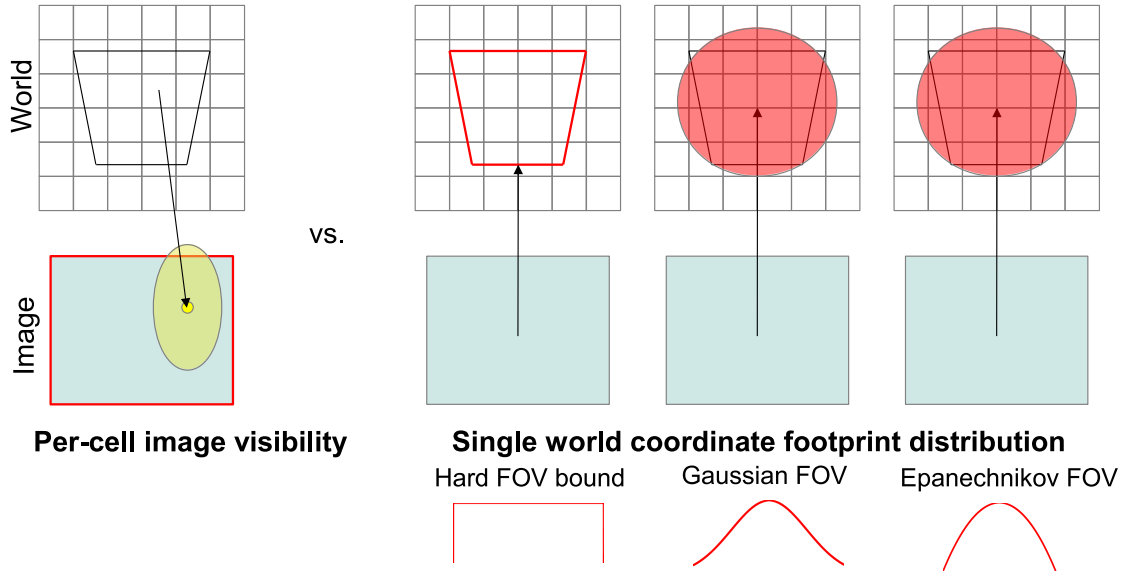


Figure 5.4: Graphical depiction of the proposed approximation of field-of-view (FOV). On the left, traditional per-cell uncertainty propagation is shown, in which uncertainty in UAV and sensor state is used to map a single cell location to a Gaussian in sensor coordinates, over which the CDF across the viewable sensor area is taken to determine cell visibility probability. On the right, three options in the alternative mode are shown, in which a representation of the field of view (a distribution representing likelihood of being in view) is itself projected once onto the world map, and each cell may be evaluated rapidly against this distribution.

Several hundred simulations using realistic vehicle models and injected uncertainty as used throughout this thesis were performed for each of the three proposed approximations as well as traditional linearized uncertainty propagation. The environment was the same as that used for simulations in Figure 4.19, with random initial placement of a single UAV for each iteration. Two very simple trajectory planning methods were evaluated, both evaluating trajectories generated from an action tree of

possible steering directions maintained with zero-order-hold for 5 seconds. The first produced single-step plans, and the second produced plans with a 5-step horizon. In simulation, the box-kernel produced unacceptable results—frequently failing to clear the map by consistently and riskily placing remaining areas of probability near the edges of the field of view, where they were often not viewed—and was dismissed as nonviable.

Results for the remaining simulations are given in Tables 5.1 and 5.2. The prior provides per-iteration planning time, which as expected is exponentially worse for the 5-step horizon. Likewise, as expected, planning time using this naive method and per-cell uncertainty propagation is substantially below that required to execute such a mission in real-time. By very greatly reducing the per-cell projection cost, however, even with modest exponential growth the total planning cost remains tractable. At the same time, time to map clearance (required mission duration) may be compared. Also as expected, single-step planning produces far worse results than horizon planning given that the prior will fail to efficiently reach remaining pockets of probability outside its single motion.

What is of greatest importance is that planning time is not substantially worsened when using the approximations. Given relatively high variability of map clearance (owing mostly to sensitivity in initial conditions), all values for single-step planning and 5-step planning are within experimental error of one another, respectively. This indicates that the approximation had little if any measurable impact on task performance, which provides strong encouragement for its use in practice. However, within the available data, a slight trend towards worse clearance time for single-step planning may be seen, which is expected given the approximate nature of these visibility models. For a 5-step horizon, the Gaussian approximation provides somewhat worse results, while the Epanechnikov kernel produces actually slightly better results. This may be explained by the fact that per-cell uncertainty propagation is itself a form of approximation (albeit an assumed better one), while this kernel is particularly aggressive about ensuring that high-probability regions lie near the image center, maximizing their chances of usefulness.

	Single-step	**5-step horizon**
Per-cell projection	6.3ms	7.4s
Gaussian FOV approximation	0.3ms	0.34s
Epanechnikov FOV approximation	0.3ms	0.32s

Table 5.1: Per-iteration planning time in repeated simulation comparing traditional per-cell uncertainty propagation against the proposed approximations.

	Single-step	5-step horizon
Per-cell projection	1905s ± 663s	711s ± 115s
Gaussian FOV approximation	2174s ± 727s	957s ± 181s
Epanechnikov FOV approximation	2562s ± 906s	670s ± 90s

Table 5.2: Time to map clearance in repeated simulation comparing traditional per-cell uncertainty propagation against the proposed approximations. High variability precludes detailed analysis, however the conclusion that little overall impact on map clearance time when using the approximations is may still be made.

5.3 Efficient Road Network Coverage

Fundamentally, a coverage task is one of producing a path along which to steer the vehicle–and hence the relatively small footprint of an onboard camera–that results in all locations of interest passing under the sensor footprint at least once during the motion. This differs from most planning tasks in that the objective is not to reach an explicit goal location in minimum time or to optimize a short-term information-theoretic or visibility objective as in probabilistic search, but rather to simply generate a path of minimal length providing coverage. Example applications include mapping, landmark detection, and search for stationary (in contrast to evasive) targets. Existing strategies typically include either the use of pre-determined spiral or lawnmower-like patterns to exhaustively sweep an area (resulting in a very conservative search that may be very inefficient in sparse environments containing large areas known to not be of interest, yet without a straightforward means of incorporating such knowledge) or the planning of deliberate motions as in probabilistic search to produce a time-optimal coverage pattern given a prior likelihood distribution of area importance (which, unfortunately, as just discussed represents a complex continuous-space trajectory optimization task that must be grossly approximated to maintain tractability). This section instead explores the alternative approach of planning coverage patterns in the abstract space of the area to be covered, with specific focus on the common case of road networks, which are then mapped back to a sequence of high-level UAV actions.

5.3.1 Relevant abstractions

Directly solving a graph coverage task within a road network's underlying connectivity graph produces, conceptually, an optimal coverage path in the space of the road network that may provide vital insight for aerial coverage. Given that intersections (graph nodes) are of relatively infinitesimal size compared to road segments

(edges) and may in practice simply be considered endpoints of road segments, the most direct phrasing of the abstract problem statement is that of producing a minimal covering path in edge space (simply, a minimum-length connected sequence of edges containing each edge at least once). This is known in the graph-theoretic literature as the *Chinese Postman Problem* (CPP), a generalization of finding Eulerian paths (requiring that each edge be traversed exactly once), which has been shown to be solvable in polynomial-time [36] for strictly-undirected (most physically realistic) or strictly-directed graphs and for which fast practical algorithms exist. [123] However, though edges may be assigned costs indicating for instance coverage traversal time, a CPP-produced edge sequence may correspond to a highly suboptimal coverage trajectory. Most importantly, the CPP framework cannot capture the true UAV motion cost of moving *between* edges (instead assuming this cost to be zero) that in actuality depends on the relative orientation of edge endpoints. Further, it artificially constrains aircraft to the connectivity of the graph as though it were a ground vehicle, failing to take advantage of cases in which skipping across unconnected space could reduce total coverage time. Similarly, the fact that the true field of view is a finite area rather than a sweeping point is ignored, providing no means to represent overlapping coverage of multiple points in the environment from one vantage location. Finally, the CPP task does not scale directly to multiple searchers, as all complexity results and fast exact algorithms of which the authors are aware are stated for the single-vehicle case. Oh et al. [91] explore several ad hoc heuristics for extending CPP solutions to multiple vehicles, with limited success.

To address several of these limitations, the study of the CPP problem has previously been extended to several variants including the *modified CPP* (mCPP) that treats the graph edges as a basket of segments without artificially constraining their coverage to the graph's connectivity and the *modified Dubins CPP* (mDCPP) that further adds an inter-edge cost derived from the motion cost for a Dubins vehicle to move between them. Oh et al. [91] also explored phrasing the mDCPP as a multi-choice multi-dimensional knapsack problem using mixed integer linear programming, which fully represents the minimization of the longest Dubins path of any vehicle but is clearly intractable for any substantially-sized environment. Oh et al. [90] later presented an alternative approach to the mDCPP task that greedily builds up a coverage sequence by incrementally extending the current sequence, inserting each edge at the current-best point within the existing sequence while accounting for vehicle maneuvering costs between each edge. This suffers from several weaknesses including inevitably local optimality, potentially excessive maneuvering and hence further reduced optimality due to fixing the choice of sweep direction when an edge is added, runtime quadratic in the number of edges that is somewhat slow in practice, example

simulations demonstrated only on unrealistically widely distributed edges not typical of any real road network, and no proof of performance guarantees for multiple vehicles. Nonetheless, implementation by the authors has shown this algorithm to work quite well in many scenarios, and it is therefore used as a primary point of comparison in this paper.

A related path-optimization problem in graphs is the well-known *Traveling Salesman Problem* (TSP), in which an agent wishes to visit each node in a graph while minimizing intermediate transit costs. It is typically phrased for a single agent on complete graphs, so the additional constraint of visiting each node exactly once is added without loss of generality assuming edge costs adhere to the triangle inequality. A similar problem is the *Vehicle Routing Problem* (VRP), a multi-vehicle generalization modeling delivery scheduling that is typically phrased so as to minimize total team travel cost while not exceeding a per-vehicle capacity corresponding to sums of per-node demand. Neither of these problems directly solves the coverage task of interest as they operate on nodes rather than edges and, as with classical edge-oriented problems, do not capture path-dependent vehicle motion cost. Further, both are NP-complete for even relatively restricted cases, with the TSP itself being a canonically hard problem to reduce from when demonstrating the NP-hardness of other problems. However, their natural form as representing the cost of paths through abstract states and their many physical instantiations have led to fast heuristics, with a common Lin-Kerninghan implementation [8] providing good solutions for thousands of nodes in several seconds. Recent work in the aerial domain for mission planning and point-of-interest visitation have studied the integration of true motion costs and the development of bounds on sub-optimality if they are neglected. [110, 80, 39] This paper proposes a coverage strategy building upon the TSP that maps edge coverage to a node-visitation problem that corresponds well to the inherent capabilities of small UAVs.

5.3.2 Visibility model

Though largely applicable to downward- and forward-facing cameras as well, the proposed strategy focuses on side-facing cameras owing to several convenient practicalities. Most importantly, while the UAV orbits a point with the camera pointed inward, a finite region (conservatively describable as a circle having radius approximately half the orbit radius for typical vehicle parameters described in Section 5.3.5) around the center of the orbit remains in view at all times, even taking into account vehicle roll aerodynamically required to produce the turn. For conciseness, the "radius of field of view" of this consistently viewed interior region is henceforth written

R_{FOV}. The immediate implication is that to view an area around a given point, the aircraft needs only to momentarily lie anywhere on the orbit surrounding that point. This is depicted pictorially in Figure 5.5. More generally, an arbitrary viewing trajectory (covering a finite region of width typically equal to the orbit radius) may be followed, even one containing arbitrarily sharp turns: the trajectory may simply be followed with a lateral offset of one turning radius, and where the turning rate is exceeded, the UAV need only continue an orbit around the current location until aligned with the desired tangent vector.

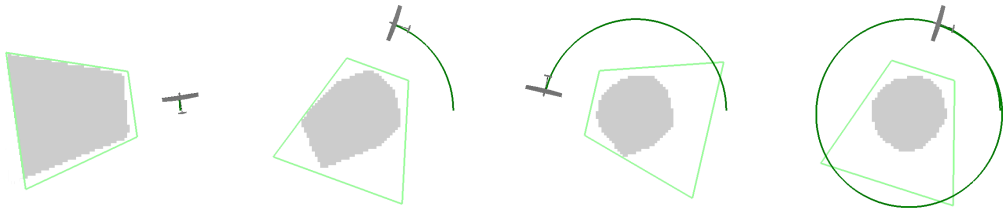


Figure 5.5: Depiction of the consistently viewable region while orbiting a point using a fixed, left-facing camera. The dark green trace indicates the UAV's trajectory, the light green trapezoid is its instantaneous field of view on the ground, and the gray region shows the area that has continuously been in view since the start of the orbit. As can be seen, at any point around an orbit, regardless of when the orbit was begun, a circularly-shaped area having radius approximately half the orbit radius will always be in view.

5.3.3 Basic tessellated node coverage strategy

The proposed strategy takes inspiration from the fact that under the models described, a UAV need only reach some point in an orbit around a location to cover a surrounding region. By judiciously placing and sequencing orbit centers so as to minimize their number and inter-orbit motion costs, efficient coverage may be obtained. Though designed to be appropriate for Dubins vehicles, it does not fully optimize the Dubins motion cost of the coverage path. Section 5.3.4 explores ways to improve upon this to better capture true motion costs.

Algorithm description

At a high level, the strategy may be described as consisting of several simple steps.

1. **Tessellate regions to be covered with sensing-footprint-sized circles**

The road graph $(\mathbf{I}, \mathbf{S})$ of intersections $\mathbf{I}$ and interconnecting segments $\mathbf{S}$ is discretized, with circles of radius R_{min} placed along each edge having centers spaced at most distance d apart producing a set $\mathbf{O}$. Given the prior observation that the intersection of sensor footprints around an orbit may be conservatively represented as a circle with half that of the orbit, $R_{FOV} = R_{min}/2$ is chosen as the value of d. Additional regions of interest not lying on roads may be added, for instance by overlapping tiling. This provides applicability to environments not strictly in the form of a road network, such as one containing small pockets including lots, parks, or fields. The resulting discretization represents a collection of orbits which, if the UAV enters each at least momentarily, provide full coverage of the environment.

Where edges lie in close proximity (such as near intersections and more frequently in dense environments) and with possibly large values of R_{FOV} (more than 60m for the parameters considered in Section 5.3.5), this coverage discretization may contain substantial redundancy, in that many circles may be safely removed without sacrificing coverage completeness. Optimally choosing a minimal subset of these is a special case of the *Planar Sensor Cover Problem*, itself a very hard problem. [51] For the purposes of evaluation in this paper, simple greedy circle elimination is used, still achieving substantial reductions.

Figure 5.6 provides an example reduced tessellation of a small road network representable as a graph.

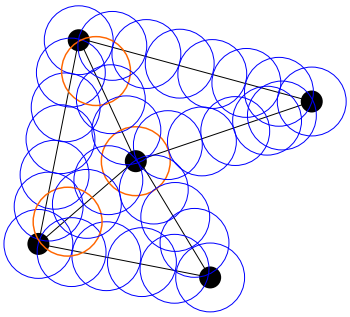


Figure 5.6: Example tessellation by orbits of a small environment, with easily-eliminated redundant orbits highlighted in orange. Larger environments and better tessellation provide more opportunities for redundant coverage elimination.

2. Provide as visitation points to graph tour generator, producing an orbit sequence

The problem is now one of choosing a sequence of and path between this set of orbits. To accomplish this, it is treated as an instance of a discrete

routing problem (a TSP or VRP). A complete graph is synthesized, with nodes corresponding to orbits ($\mathbf{V} = \mathbf{O}$) and edges to the action of traveling between them ($\mathbf{E} = \mathbf{O} \times \mathbf{O}$). As the true cost of moving between orbits is path-dependent (upon which orbits immediately precede and succeed the current one), the substantial simplification is made to use the Euclidean distance between orbit centers as the edge cost. The impact of this on path optimality is studied in the next subsection, with mitigating improvements then suggested.

For a single vehicle, this directly corresponds to a traditional TSP, and with the use of a complete graph and Euclidean distance metric, the use of existing heuristic solvers highly-optimized for this common scenario is possible. For multiple vehicles, the problem is technically a *mini-max multiple TSP*, since it is most likely desired that the length of the longest path among all vehicles be minimized so as to optimize for overall mission time (rather than, for instance, total distance traveled or roughly equivalently, total flight time or energy expended). This problem is substantially less well studied, and though more elaborate solutions may be applied, repeated invocations of a capacitated VRP were used to approximately search for a multi-vehicle solution for the purposes of comparison in this paper. VRP problems are also typically phrased to assume a single "depot" location (a visitation point shared by all vehicles), which may be taken to be either a common launch location (if all are launched from one place) or an artificially chosen location whose distance to all initial vehicle locations is minimized.

3. Choose the nearest point on the tour for each vehicle and tour direction

The output of a routing problem solver is a closed sequence of orbits to pass between for each vehicle. This tour does not necessarily pass through the initial location of each vehicle (though it may be explicitly directed to do so, for instance by adding an additional visitation point), and so a point on the tour must be selected. For simplicity, the orbit having lowest Dubins path cost to reach is selected. Likewise, the expected distance to travel if the tour were followed in the forward and reverse directions (these may differ due to the path following scheme next described) are compared, and the lowest-cost direction is selected.

4. Follow the orbit sequence with each vehicle in parallel

For UAVs equipped with a side-looking camera, once on the tour, orbits are passed through in the chosen sequence by maintaining the current orbit until

the vehicle's heading vector aligns with the tangent vector between the current orbit center and the next one. Once aligned, a straight-line path is taken to the next orbit, and since all orbits are of the same radius, the vehicle will eventually lie on the next orbit, and the procedure may be repeated. Alignment followed by straight-line motion will always be possible because this represents an instance of either the LSL or RSR Dubins motion case (consisting of an orbit plus a motion between orbit outer tangents).

For UAVs equipped with a forward- or downward-looking camera, any of several published strategies to best follow TSP tours with Dubins vehicles may be invoked. Most naively, the locally optimal Dubins path from the current pose to the next orbit center may be followed, with a terminal orientation that must be chosen, either by search or by arbitrarily fixing it to the direction towards the succeeding orbit center, for instance. A formalization of the latter is the "alternating algorithm" proposed and analyzed by Savla et al., [110] who also proposes a "bead-tiling" algorithm more appropriate for very dense collections of points.

5. **Return vehicles to their start locations**

   Being a closed path, the tour of orbits for each vehicle will terminate at its initial orbit, which by design either was or is close to its starting (e.g. take-off) location. Thus, an optimized overall mission plan is provided, leading from launch to recovery.

Analysis

Naturally, one wishes to evaluate the performance of this strategy, particularly in the sense of overall mission duration optimality. From the start, this is not necessarily optimal among all possible coverage strategies, as the choice to tessellate with orbit-sized circles was made as part of a larger effort to trade off optimality for feasibility and is not easily evaluated. However, the optimality of the path cost through the sequence of orbits (and hence mission duration) may be studied, with overall strategy performance left to experimental comparison. The fundamental source of suboptimality is the use of Euclidean distance between orbit centers as an approximation of the motion cost for a Dubins vehicle to move between orbits.

For the special case of side-facing cameras forming the focus here and the aforementioned path following strategy of maintaining an orbit until aligned with the next one in the sequence, some simple analysis may be applied to determine the amount of path length inflation over a hypothetical vehicle having no motion constraints.

Specifically, we consider the worst-case additional distance that must be traveled while moving between orbits over the cost of moving directly between orbit centers. For a given sequence, an upper bound of π radians per orbit on average may be easily established because if the average were higher, simply following the tour in the opposite direction will reduce this below π (in general, if an orbit is entered at absolute heading θ_1 and left at absolute heading θ_2, traversing $\Delta\theta$ in the interim, then entering the orbit at θ_2 and leaving at θ_1 will traverse $2\pi - \Delta\theta$). Meanwhile, a lower bound on a worst case average may also be seen to π radians through the construction of an adversarial environment. Consider an environment consisting of only parallel segments of length R_{FOV} forming a set of stairs. Regardless of direction, every other corner requires maintaining an orbit for either $\frac{\pi}{2}$ or $\frac{3\pi}{2}$ to align with the next segment, for an average of π for a sufficiently long sequence. Thus, this bound is tight, and the additional maneuvering required may reach $\pi R_{min}|\mathbf{O}|$. This is substantial and undesirable, both because it can be quite large and because it grows linearly with the size of the environment. However, it is claimed that environments containing highly excessive maneuvering are contrived rather than typical as well be shown in simulation, and the following section suggests several extensions to improve upon this.

For forward- and downward-looking cameras, for which orbit visitation implies momentary visitation by the UAV itself (possibly with some posterior standoff), existing path following strategies and accompanying optimality bounds may be considered. For instance, Savla et al. [110] present an inflation bound for the the "alternating algorithm" given by an increase over the Euclidean path length of at most $\tau\pi R_{min}$ per alternate pair of visitation points (so that the total path cost is approximately $\frac{N}{2}\tau\pi R_{min}$) where $\tau \in [2.657, 2.658]$ and N is the number of points. Note that this exceeds πR_{min} on average, and hence that the inflation provided bound for side-facing cameras is lower.

5.3.4 Coverage strategy extensions

To address the potentially substantial path suboptimality shortcomings of the basic proposed method, several extensions are considered that better represent the true Dubins motion cost of moving between coverage visitation locations. For the first two, the same orbit-based tessellation and path following is considered, but the synthetic graph fed to the abstract tour generator is varied to produce more physically optimal sequences. The third operates directly on the set of road segments, synthesizing a graph based upon it.

Exact (Pairwise) Motion Cost Encoding

Due to the path-dependent nature of true motion costs, a graph consisting of merely positions is insufficient to capture these costs. Instead, a hypergraph may be constructed in which each vertex represents the state of being at an orbit having come from a prior orbit and edges represent the exact total Dubins cost of (a) maintaining the current orbit from an initial orientation aligned with the prior orbit to an alignment with the next orbit followed by (b) the straight-line motion to the next orbit. Formally this may be stated as

$$\begin{aligned} \mathbf{V} &= \mathbf{O} \times \mathbf{O} \\ \mathbf{E} &= \{(O_i, O_j) \to (O_j, O_k) | O_i, O_j, O_k \in \mathbf{O}\}, \end{aligned} \tag{5.4}$$

noting that this is no longer a complete graph and that edge costs are given by

$$cost\left((O_i, O_j) \to (O_j, O_k)\right) = 2\pi \cdot angle\left(\frac{O_j - O_i}{|O_j - O_i|}, \frac{O_k - O_j}{|O_k - O_j|}\right) + |O_k - O_j|, \tag{5.5}$$

where $angle(v_i, v_j)$ represents the angle between two vectors v_i and v_j. This construction is depicted graphically in Figure 5.7.

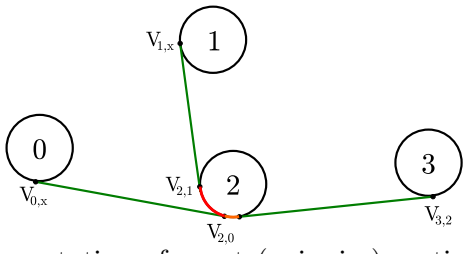


Figure 5.7: Graphical representation of exact (pairwise) motion cost encoding. Note the path segments highlighted in orange and red. The cost of moving from node 0 to 3 via 2 includes the maneuver around node 2 in orange, and the cost of starting at node 1 instead further includes the additional maneuver in red. In contrast, using only the Euclidean distances between orbit centers as inter-node cost neglects both, capturing only the green segments.

A graph of this form presents several complications. The first is that an orbit sequence may not be derived by simply applying a TSP solver, as this will return a tour containing *all* nodes, revisiting each node $|O|$ times. Instead, this may be seen as a *Generalized TSP* or a discrete instance of a *TSP with neighborhoods*, since nodes may be separated into $|O|$ disjoint clusters or neighborhoods, defined by

$$\mathbf{C_i} = \{(O_j, O_i) | O_j \in \mathbf{O}\}, \tag{5.6}$$

providing a separation into clusters each representing the physical state of presence at a given orbit, grouping all possible predecessor orbits.

Using a similar strategy employed by Isaacs et al., [63] this may be converted from this form of TSP that is not easily solved to a more common *asymmetric TSP* (ATSP) using the Noon and Bean transform [87] without any growth in the graph. Finally, an ATSP may be further transformed as described by Kumar and Li [77] (building upon the idea incompletely proposed by Jonker and Volgenant [69]) to a symmetric TSP, at the cost of doubling the number of nodes so that fast heuristic symmetric solvers may be applied.

A second complication is that a graph on $2|\mathbf{O}|^2$ nodes is much larger than the original graph, which itself may be large already as a tessellation of a vast environment. Indeed, for sufficiently large coverage area, this may prove intractable. Where this is the case, the approximation proposed as the next strategy extension may prove adequate. However, sparse environments containing up to one hundred nodes produce graphs on less than ten thousand nodes that require tractably many edge cost evaluations and are still readily solved by modern TSP heuristics.

It should further be noted that using a heuristic symmetric TSP solver (returning inexact, suboptimal solutions) as the foundation may result in node sequences that are not technically valid solutions to the full problem with the aforementioned transformations applied (such as including node re-visitation). However, for the purposes of this application, this may be considered simply further weakening of the heuristic output and may be evaluated empirically. In practice, it was noted that invalid solutions were returned extremely rarely, and within several repeated iterations using differing random seeds for the heuristic solver, valid solutions were always found.

Approximate Quantized-Pose (Angle-discretized) Motion Cost Encoding

Where quadratic growth in the graph provided to the TSP solver is impractical, an alternative may be used as an approximation. For some chosen number of samples n, equally spaced points around each orbit are selected, forming a neighborhood of locations, one of which for each orbit must be visited to achieve total coverage. A complete graph is then formed, with edge costs defined by the cost of following the first orbit starting at the initial pose until aligned with the second orbit, moving onto the second orbit, and then following the second orbit until the terminal sampled pose is reached. Formally, this may be stated as

$$
\begin{aligned}
\mathbf{V} &= \mathbf{O} \times \{1, ..., n\} = \{V_{i,j} | i = 1..|\mathbf{O}|, j = 1..n\} \\
\mathbf{E} &= \mathbf{V} \times \mathbf{V} \\
\theta_j &= \tfrac{2(j-1)}{n}\pi \\
loc(V_{i,j}) &= loc(O_i) + R_{min} < \cos\theta_j, \sin\theta_j > \\
cost(V_{i,j} \rightarrow V_{k,l}) &= 2\pi \cdot angle\left(< \cos\theta_j, \sin\theta_j >, \tfrac{O_k - O_i}{|O_k - O_i|} \right) \\
&\quad + |O_k - O_i| \\
&\quad + 2\pi \cdot angle\left(\tfrac{O_k - O_i}{|O_k - O_i|}, < \cos\theta_l, \sin\theta_l > \right),
\end{aligned}
\tag{5.7}
$$

where ϕ is $\frac{\pi}{2}$ for left-facing cameras, 0 for forward-facing cameras, etc. and counterclockwise-positive heading angles are assumed (as for instance arise in a North-East-Down coordinate convention).

As with the prior hypergraph formulation, a GTSP may be formed by clustering vertices corresponding to the same orbit, defined formally as

$$
\mathbf{C_i} = \{V_{i,j} | j = 1..n\}, \tag{5.8}
$$

and solved by the same means. This produces a graph with $n|\mathbf{O}|$ vertices, with n selectable based on the desired resolution, providing a means to variably trade off between optimality and computational complexity. This formulation is shown in Figure 5.8.

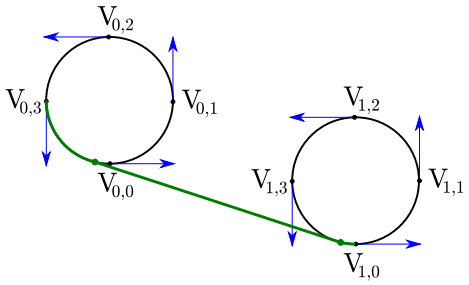


Figure 5.8: Graphical representation of approximate quantized-pose motion cost encoding with $n = 4$. In general the phase angle of pose samples is a free variable, here chosen so that poses are axis-aligned. Blue arrows indicate locations and corresponding orientations sampled around each orbit, and the green path shows the resulting path between two sample poses, comprised of a straight-line tangent segment having the same length as the Euclidean distance between orbit centers and two maneuvers for each orbit linking this segment to the desired quantized poses.

Cost-aware Road Segment Sequencing

Taking inspiration from both the mDCPP algorithm of Oh et al. and clustered graph formulations on which a GTSP is solved, an alternative graph may be synthesized directly from the set of road segments $\mathbf{S}$. In this variant, two nodes are assigned to each segment, one corresponding to the pose at the start of the edge along each direction. A complete $|\mathbf{S}|$-partite graph is then formed by connecting each node to all nodes except its opposite-direction partner, with edge costs defined by the Dubins motion cost between poses. Formally, this may be stated as

$$\begin{aligned}
\mathbf{V} &= \mathbf{S} \times \{0,1\} = \{V_{i,j} | i = 1..|\mathbf{S}|, j = 0..1\} \\
\mathbf{E} &= \mathbf{V} \times \mathbf{V} \\
[P_{i,0}, \theta_{i,0}] &= [S_{i_0}, atan(|S_{i_1} - S_{i_0}|_x, |S_{i_1} - S_{i_0}|_y)] \\
[P_{i,1}, \theta_{i,1}] &= [S_{i_1}, atan(|S_{i_0} - S_{i_1}|_x, |S_{i_0} - S_{i_1}|_y)] \\
cost\left((V_i, j) \rightarrow (V_k, l)\right) &= |S_{i_0} - S_{i_1}| + dubins_cost([P_{i,(1-j)}, \theta_{i,j}], [P_{k,l}, \theta_{k,l}]),
\end{aligned} \tag{5.9}$$

where $(S_{i_0}, S_{i_1}) \in \mathbf{S}$ and $dubins\_cost([(x_1, y_1), \psi_1], [(x_2, y_2), \psi_2])$ represents the motion cost of the optimal Dubins motion between two planar poses.

Nodes clusters are then defined by

$$\mathbf{C_i} = \{V_{i,0}, V_{i,1}\}, \tag{5.10}$$

producing a graph with $2|\mathbf{S}|$ nodes, with each cluster containing two nodes. This is intuitively presented in Figure 5.9.

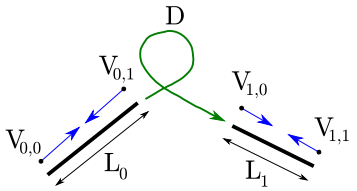


Figure 5.9: Graphical representation of cost-aware road segment sequence encoding. Each edge is assigned two nodes in the synthetic graph, corresponding to being located at opposite endpoints with pose pointed inward (as though about to sweep that edge). The motion cost between any two synthetic nodes is the cost of sweeping the edge the first node lies on plus the cost of a minimal maneuver between the endpoint pose of that sweep and the pose of the second node.

Solving the GTSP problem on this graph produces a minimal-length path that sweeps all road segments, fully taking into account inter-segment motion costs, en-

coded as a sequence of segments and a direction in which to sweep. This directly and globally solves the mDCPP problem, without a need to approximate motion costs or resort to a greedy solution, and it is highly tractable since a GTSP on up to several hundred road segments may be solved quickly with even exact TSP solvers. This method is henceforth referred to as the sweeping generalized TSP (SGTSP).

This method does have several caveats. First, as phrased, road segments are assumed to be linear, however this may be easily extended to arbitrary UAV-sweepable trajectories whose arclength is known. More importantly, tessellation of any open areas must be done with short segments rather than by placing orbits, and redundant coverage is not easily eliminated as coverage of only parts of road segments may overlap. Finally, since physical vehicles cannot actually execute tight Dubins maneuvers containing discontinuous curvature, the resulting trajectory may not be feasible, whereas the other methods built upon orbits are more conservative.

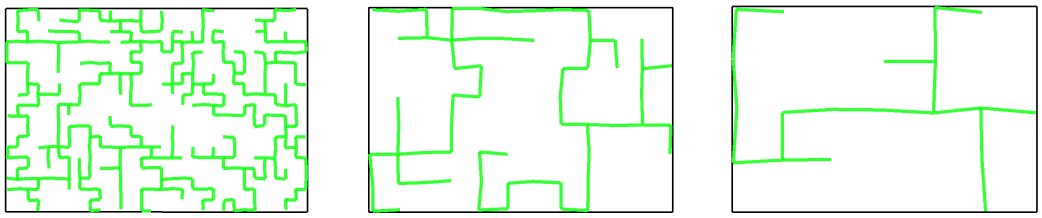
5.3.5 Experimental results

The proposed method and its extensions were evaluated in extensive realistic simulation using of fixed-wing aircraft with side-angled cameras as elsewhere in this thesis. For the purposes of visualization and evaluation only, a cellular Bayesian estimator similar to that described in Section 5.2 was used, with road segments finely discretized and each cell containing the probability of a target remaining assuming initially total contamination ($p = 1.0$ for all cells).

Three methods are initially primarily compared. The first is the aforementioned basic tessellated node coverage strategy, abbreviated OTSP. This is evaluated against a standard lawnmower-like Zamboni coverage pattern, which interleaves alternating axis-aligned sweeps of the entire length of an environment so as to avoid turns exceeding R_{min}. The third is the previously described mDCPP algorithm proposed by Oh et al., [90] which may be summarized as iteratively growing a sequence of road edge sweeps by greedily adding the edge whose insertion into the sequence (taking into account inter-edge Dubins maneuvers) increases the path cost by the least, with deconfliction between multiple vehicles provided by an auction-based scheme assigning the conflicted edge to the vehicle whose path cost would worsen the most if it did not receive that edge.

Comparative simulations were performed over a spectrum of randomly-generated environments intended to realistically mimic physical road networks. These were generated by starting with a dense 4km $\times$ 6km grid network with specifiable block size (edge length), jittering intersection locations, and removing a specifiable fraction of edges. The combination of edge length and fraction of edges removed conceptually

correspond to environment density. In these simulations, 30% of edges were retained, and edge lengths were varied between 100m and 1km, corresponding to a range encompassing dense urban, suburban, and sparse rural. Example environments are shown in Figure 5.10. Very roughly, for the purposes of comparison here, environments with edge lengths within the range 100-300m will be referred to as urban, 300-700m as suburban, and 700m-1km as rural.

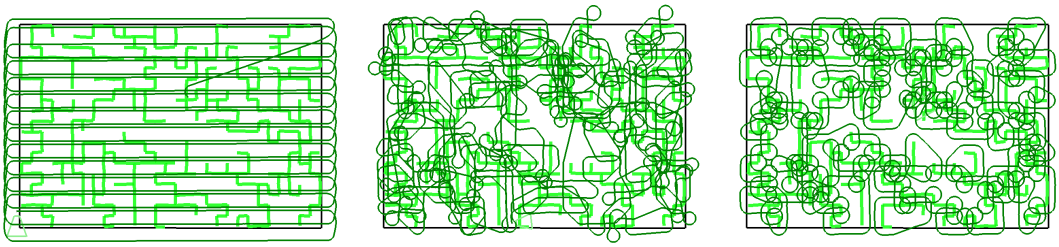


(a) Urban (b) Suburban (c) Rural

Figure 5.10: Sample environments along the density spectrum

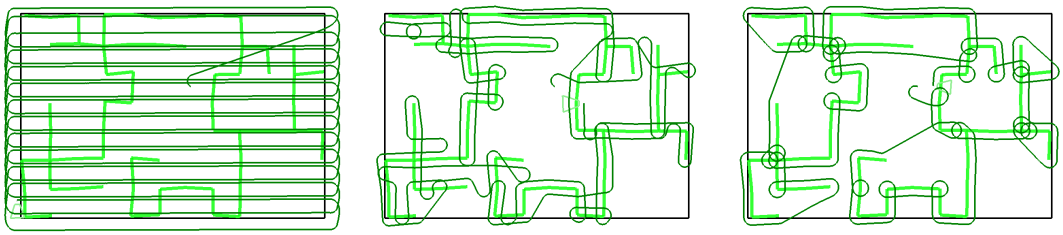
Several hundred simulations for each of 10 levels of environment density were performed, each with differently randomized environments and vehicle starting poses. For each run, the algorithm precomputation time, time to mission completion, and mission success rate (whether total coverage was achieved) were recorded. By design, total coverage was achieved in all cases. A typical run highlighting the strengths and weaknesses of each method over urban, suburban, and rural environments is shown in Figures 5.11, 5.12, and 5.13 respectively.

Results for a single vehicle are summarized in Figures 5.14 and 5.15. From these it may be seen that for dense environments (having average road segment length less than approximately 250m), Zamboni patterns are 2-5 times faster than either OTSP or mDCPP coverage. However, for less dense environments, this is inverted, with OTSP and mDCPP performing up to about 4 times as quickly. For a large density range roughly spanning sparse urban through suburban, OTSP is faster than both Zamboni and mDCPP, with a peak improvement of approximately 20% over mDCPP (at which point it is also $\frac{1}{3}$ faster than Zamboni). This is largely due to the elimination of redundant coverage in dense areas by the removal of some orbit tessellations, while in sparser environments many orbits are still required. For suburban through rural densities, SGTP is superior to all others, by up to 10%, while matching the performance of mDCPP for denser environments (indeed, any loss to mDCPP is attributable to experimental error and the use of a heuristic TSP solver). As this can be seen as a global version of mDCPP, this highlights the good performance of greedy assignment within mDCPP in areas having much to cover in all



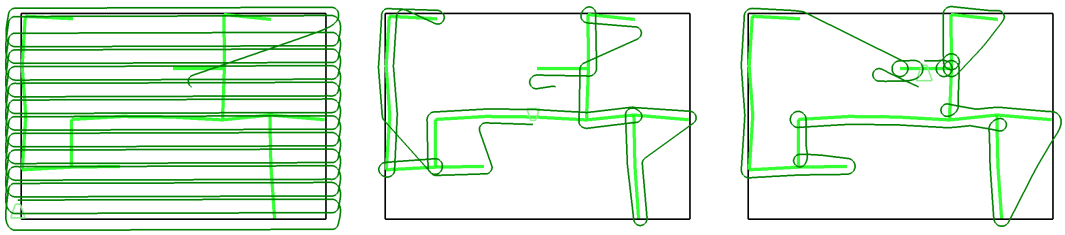
(a) Zamboni: 12,232s (b) mDCPP: 20,465s (c) OTSP: 16,177s

Figure 5.11: Sample execution results for three coverage algorithms in an urban-density environment. Zamboni patterns operate independently of the presence of any roads and cover all areas, even those not of interest. The mDCPP and OTSP strategies consider only roads, with the OTSP algorithm exhibiting better macroscopic path optimality owing to its use of a global solver but performs more maneuvering. Typically for this density, Zamboni outperforms the others.



(a) Zamboni: 12,318s (b) mDCPP: 5,299s (c) OTSP: 4,490s

Figure 5.12: Sample execution results for three coverage algorithms in a suburban-density environment. Zamboni performs particularly poorly relative to the others because it is unvarying (up to random variation, exhibited here) in its path. Once again, OTSP shows better macroscopic optimality which is sufficient to outperform mDCPP, though some excessive maneuvering remains.

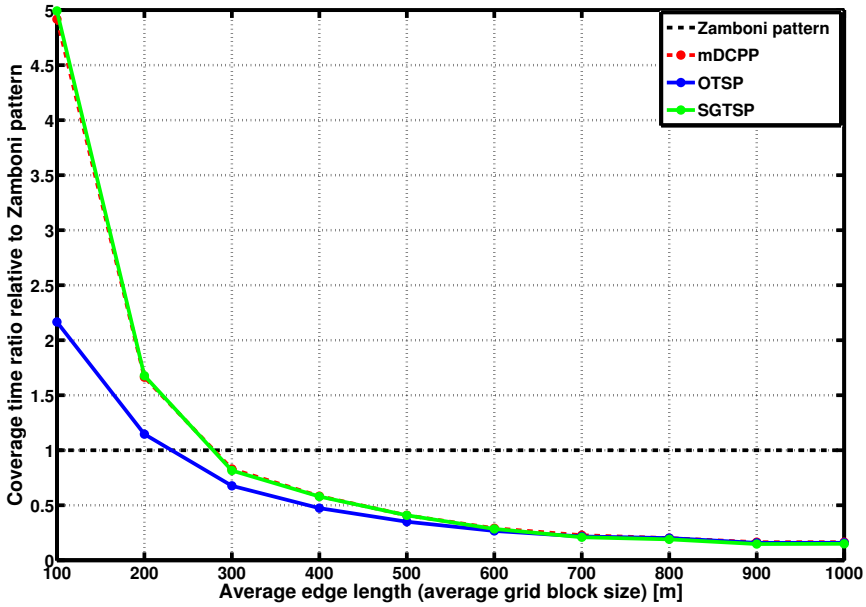


(a) Zamboni: 12,336s (b) mDCPP: 2,393s (c) OTSP: 2,426s

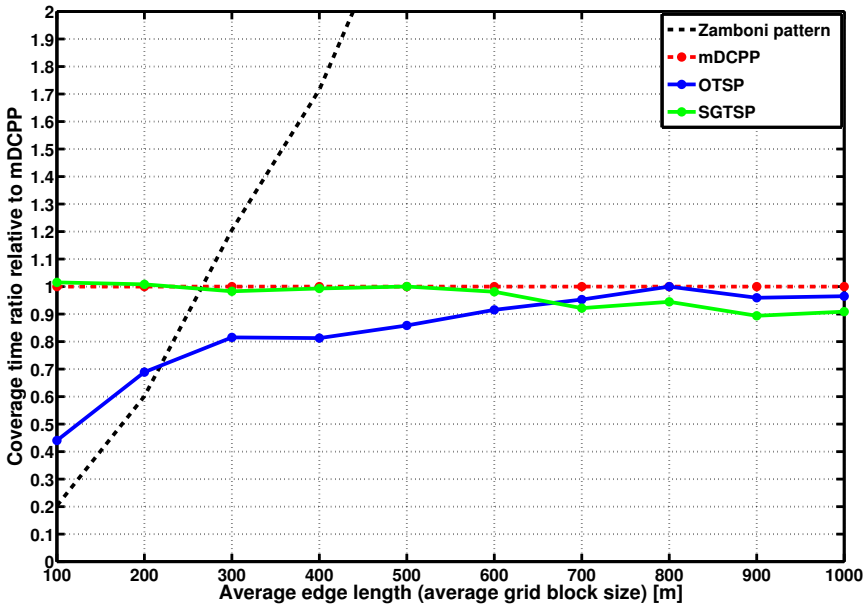
Figure 5.13: Sample execution results for three coverage algorithms in a rural-density environment. Here again, Zamboni performs particularly poorly relative to the others given the very small area of actual interest. Because it considers only full edges regardless of their length, mDCPP outperforms OTSP, which sequences more minute motions, though only by less than 2%.

directions (urban) and the consequences of poor assignment for longer sweeps (rural). The pairwise-cost extension is omitted as it proved intractable for all but rural environments, and the quantized-pose extension is not shown as it performs poorly owing to a large problem space and sparse connectivity between synthetic nodes, both properties that reduce heuristic solver performance. Precomputation time for Zamboni patterns is categorically negligible and is minimal (several seconds) for both OTSP/SGTSP and mDCPP for suburban and less dense environments. However, due to its $O(|E|^2)$ greedy assignment of edges, very dense environments containing hundreds of segments result in precomputation times of several minutes on average for mDCPP. Though this too is likely insignificant in practice but may prove unsuitable for situations requiring frequent replanning. SGTSP precomputation time is generally better than OTSP given that the state space is entire edges rather than tessellated orbits, while increasing for very dense environments in which orbit elimination maintains low OTSP runtime. This is of no consequence, however, as in this density range Zamboni patterns are preferable.

These behaviors may be contrasted with traditional strategies for environment-aware and so-called information-theoretic trajectory planning. Executions of several simple instances of such are compared in Figures 5.16 and 5.17 for urban and rural environments. The first instance is a continuous control law that chooses an instantaneous steering angle corresponding to the probability gradient of a cellular Bayesian probability map maintained during the coverage search (and identical to that used for visualization), derived by convolving a Gaussian derivative kernel with the map at the center of the field of view with $\sigma = kR_{FOV}$ for some small integral k. The second is a single-step (greedy) horizon planner that searches among a set of discrete



(a) Coverage execution time relative to Zamboni



(b) Coverage execution time relative to mDCPP

Figure 5.14: Coverage execution time comparison between a Zamboni pattern and the greedy mDCPP competitor, orbit TSP (OTSP), and sweeping generalized TSP (SGTSP) algorithms for a single vehicle. Note that for very dense environments, simple a Zamboni patterns is most appropriate. For less dense ones, OTSP is best. Finally, for very rural environments, SGTSP is preferable.

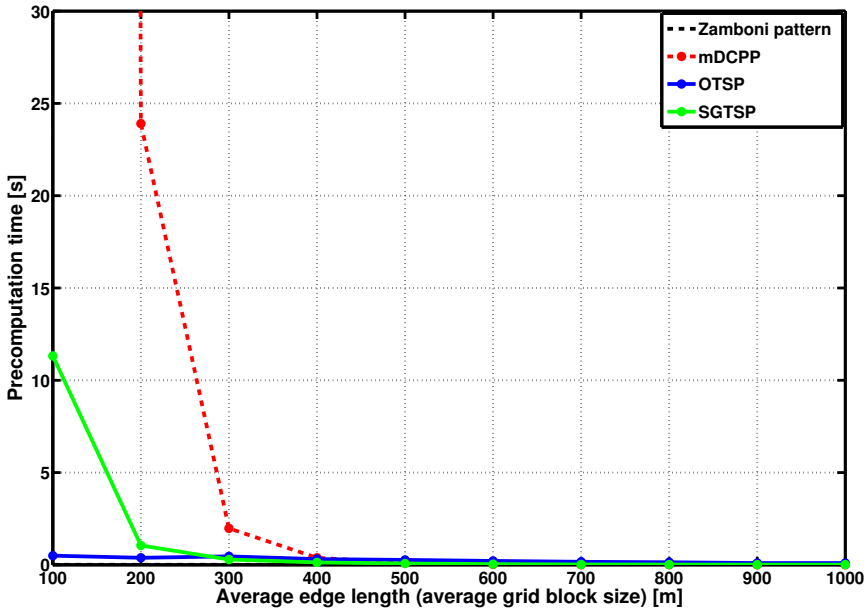
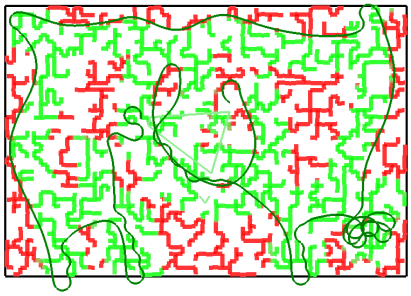


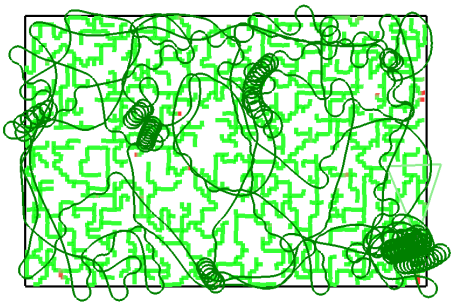
Figure 5.15: Precomputation time comparison between a Zamboni pattern and the greedy mDCPP competitor, orbit TSP (OTSP), and sweeping generalized TSP (SGTSP) algorithms for a single vehicle. Note that computation time for Zamboni patterns is negligible.

steering commands, which when held for a specified duration (zero-order hold control), maximizes the probability of detection in a similar Bayesian probability map within that next time-step, sampled at several points along the predicted trajectory. The third is a multi-step horizon planner performing an identical search, but in an action tree of steering angles. The hold duration for the latter two methods is chosen to be proportional to average road segment length so that the horizon is relevant for the scale of the environment (e.g. so that very short trajectory lookahead is not used in large sparse environments). Though Bourgault et al. [17] have demonstrated the efficacy of greedy planning for UAVs in Bayesian probability maps, its strength is primarily in producing semi-coordinated behaviors that quickly cover high-likelihood areas with minimal computation, while the primary weaknesses of such methods are that they will rarely provide *complete* coverage instead leaving scattered pockets of moderate target likelihood and are prone to getting stuck in local minima from which they will never reach the remaining pockets. Meanwhile, while multi-step horizon planning reduces such issues through non-minimum-phase behaviors enabled by lookahead, these too suffer inevitable myopia in sufficiently large environments and quickly become intractable with increasing horizon length. More advanced methods such as randomized planning are certainly available, however all fundamentally

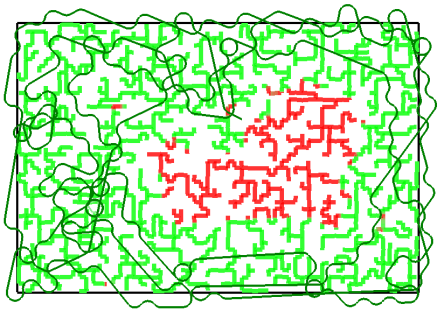
attempt to solve a continuous-space trajectory planning problem in possibly large environments and cannot offer any but asymptotic guarantees of complete coverage. In the sample executions shown, only multistep planning completes coverage successfully, and faster than an exhaustive Zamboni pattern.



(a) Probability gradient: initially captures a large amount of target probability rapidly (6,064s)



(b) Probability gradient: stuck in active local minimum, leaving small regions uncovered (15,000s)

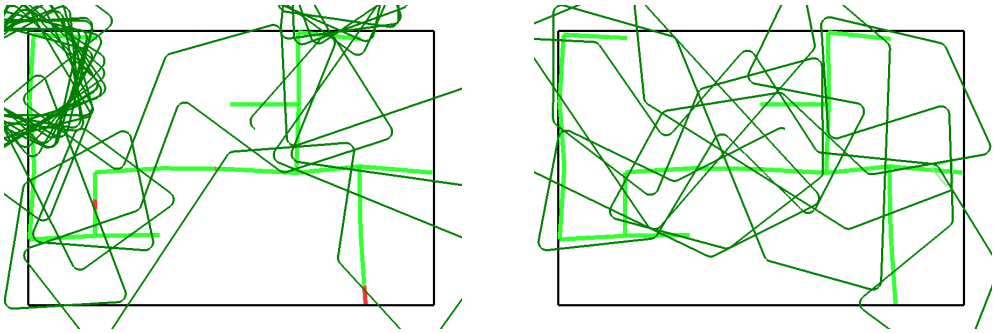


(c) Single-step (greedy) planning: stuck in local minimum (7,037s)

(d) Multi-step (horizon) planning: coverage complete (8,260s)

Figure 5.16: Sample execution results for three traditional environment-aware algorithms attempting to maximize short-term probability of detection in an urban-density environment. Of these, only multi-step planning (requiring 3.8s per planning step, which is infeasible for realtime execution) completes coverage.

Coverage using multiple-vehicle teams was also evaluated. The primary property of note in expansion to multiple vehicles is the means by which coverage of the environment is distributed among them. For Zamboni patterns, simple axis-aligned geometric slices of the environment along its major axis are chosen, and each vehicle performs a Zamboni pattern in parallel within its slice. The mDCPP algorithm is intrinsically multi-vehicle via its deconfliction auction mechanism. For strategies



(a) Single-step (greedy) planning: stuck in local minimum (8,297s)

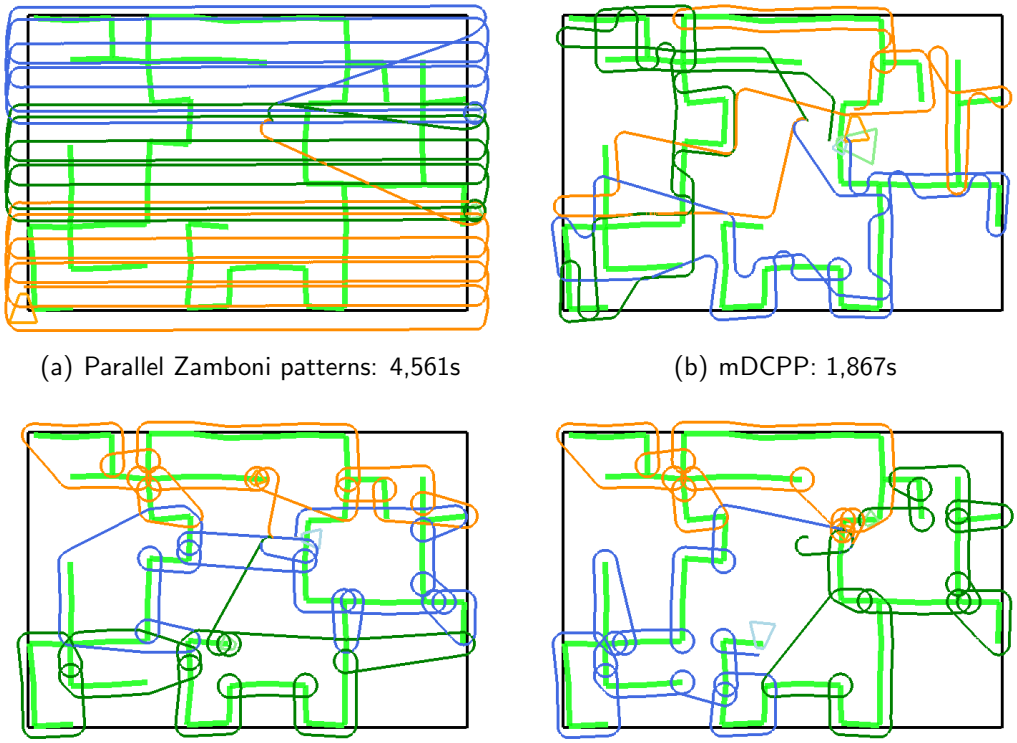
(b) Multi-step (horizon) planning: coverage complete (7,196s)

Figure 5.17: Sample execution results for two traditional environment-aware algorithms attempting to maximize short-term probability of detection in a rural-density environment. Of these, only multi-step planning completes coverage. The probability gradient rule is not even shown as it was unable to make any progress whatsoever, lacking a meaningful gradient at its start location.

involving a TSP solver, ad-hoc cooperation generated by the same geometric separation as used for Zamboni patterns in which each vehicle solves the coverage problem independently within its slice is one mechanism considered. The other is the use of a VRP solver, which simultaneously performs assignment and sequencing. However, strategy extensions requiring the use of a generalized TSP cannot make use of this given their formulation as a transformation to a standard TSP and are thus limited to ad-hoc cooperation. Example executions for three vehicles on a suburban-density environment are shown in Figure 5.18.

Experimental results for three vehicles are summarized in Figures 5.19 and 5.20. These indicate the continued preference for Zamboni patterns for sufficiently dense environments. Another trend similar trend to single-vehicle coverage is present in that OTSP again outperforms all others for sparse urban through suburban densities, while SGTSP is best for less dense environments. Of note here is that VRP-generated cooperation is moderately superior to ad-hoc geometric splitting at a fair substantial (but not intractable) increase in precomputation time. The weakness of ad-hoc cooperation relative to reasoned assignment may be seen explicitly in the better performance of the mDCPP strategy relative to SGTSP in dense environments: other than vehicle assignment, SGTSP should perform better.

Finally, explicit comparison of the first two proposed strategy extensions – angle-discretized OTSP and pairwise OTSP – is presented in Figures 5.21 and 5.22 for three-vehicle coverage. The angle-discretized strategy always performs somewhat



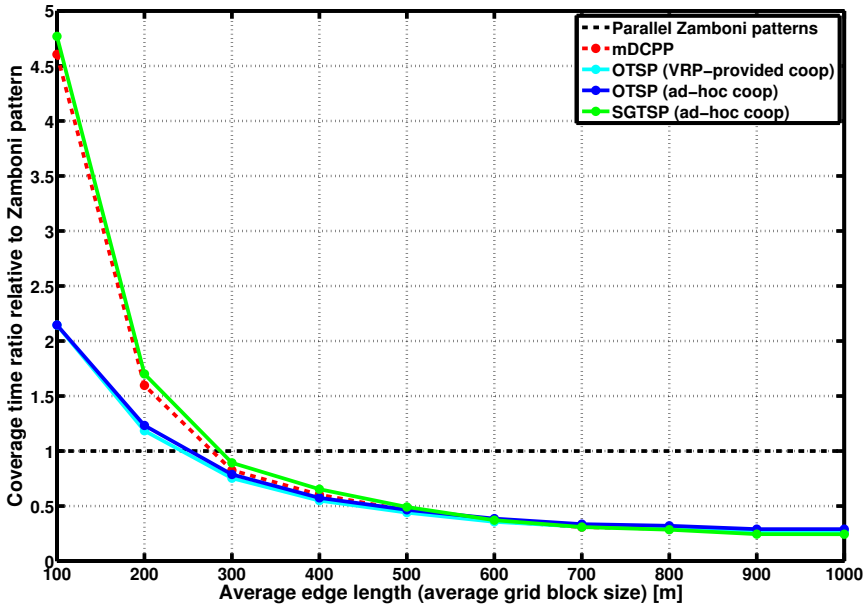
(a) Parallel Zamboni patterns: 4,561s

(b) mDCPP: 1,867s

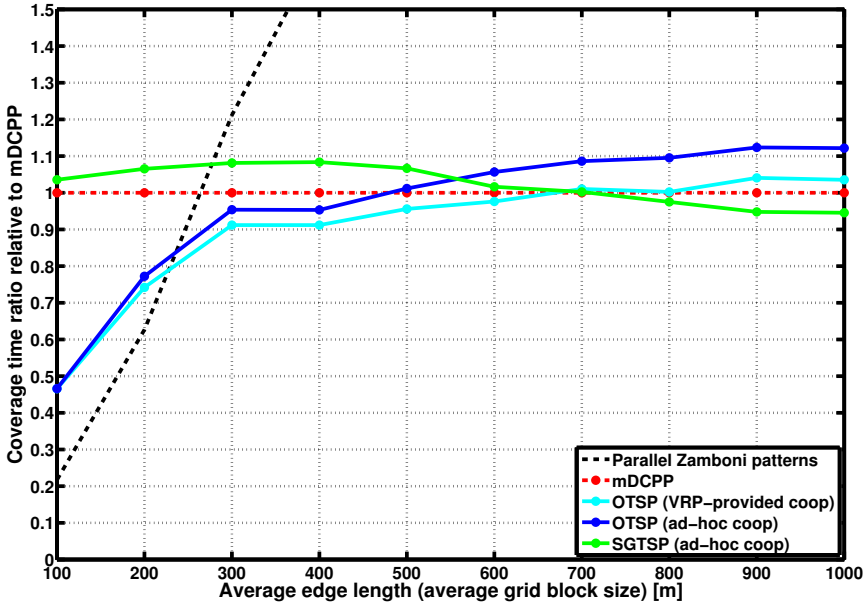
(c) OTSP with ad-hoc cooperation: 2,158s

(d) OTSP with VRP-provided assignment: 1,825s

Figure 5.18: Sample execution results for three vehicles in a suburban-density environment comparing four coverage algorithms: parallel Zamboni patterns splitting the environment into equally-sized axis-aligned areas, the mDCPP algorithm which internally uses an auction-based mechanism for assignment, the basic OTSP strategy using ad-hoc cooperation in which three OTSP problems are generated from a geometric split of the environment identical to that used for the Zamboni patterns, and the basic OTSP strategy using a VRP solver internally that simultaneously solves the vehicle assignment and TSP sequence problems. This example is typical in that Zamboni patterns are unnecessarily exhaustive, mDCPP performs somewhat better than OTSP using ad-hoc cooperation, and OTSP using the VRP solver performs best. Note the clear benefit of reasoned area assignment over a hard geometric split and the continued superiority of OTSP over mDCPP for environments of such density with the effect of assignment removed by the VRP solver.



(a) Coverage execution time relative to Zamboni



(b) Coverage execution time relative to mDCPP

Figure 5.19: Coverage execution time comparison between Zamboni patterns and the greedy mDCPP competitor, orbit OTSP (OTSP) using both a VRP solver and an ad-hoc spatial split of the environment into independent TSPs, and sweeping generalized TSP (SGTSP) algorithms for 3 UAVs running in parallel.

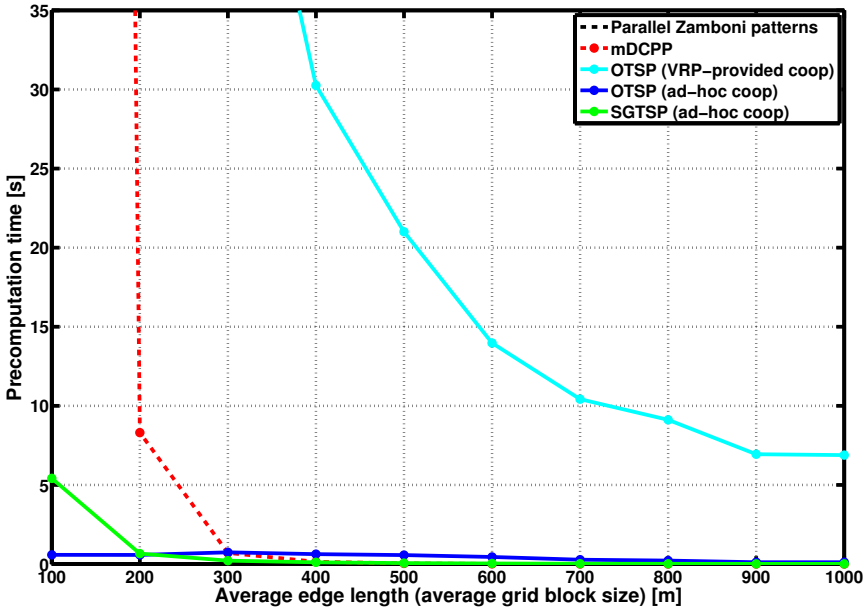


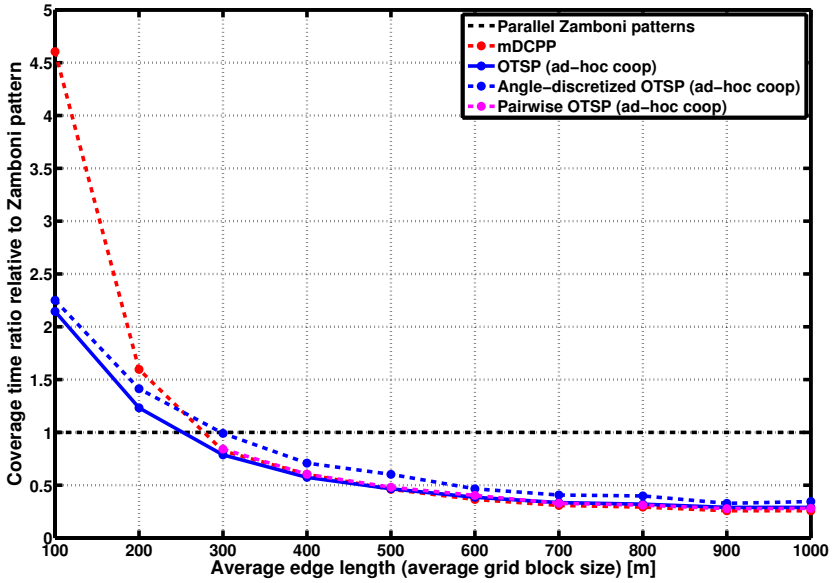
Figure 5.20: Precomputation time comparison between Zamboni patterns and the greedy mDCPP competitor, orbit OTSP (OTSP) using both a VRP solver and an ad-hoc spatial split of the environment into independent TSPs, and sweeping generalized TSP (SGTSP) algorithms for 3 UAVs running in parallel. Note that precomputation time for Zamboni patterns is negligible.

worse than the others, owing to several factors including misalignment between sampled poses and the vector of the road segment being swept, a moderately large state space reducing the quality of heuristic solutions, and sparse connectivity between states (given the clustering formulation) that further ill-conditions the graph provided to the heuristic solver. Capturing pairwise motion costs slightly improves performance, but at great computational cost due to the greatly increased problem size. Indeed, as environment density increases, performs actually worsens due to the inability of the heuristic solver to navigate the large graph, followed by total intractability at even higher densities. Overall, these results vindicate the use of the basic OTSP method over reasoned but complexity-introducing extensions, excepting SGTSP which these inspired.

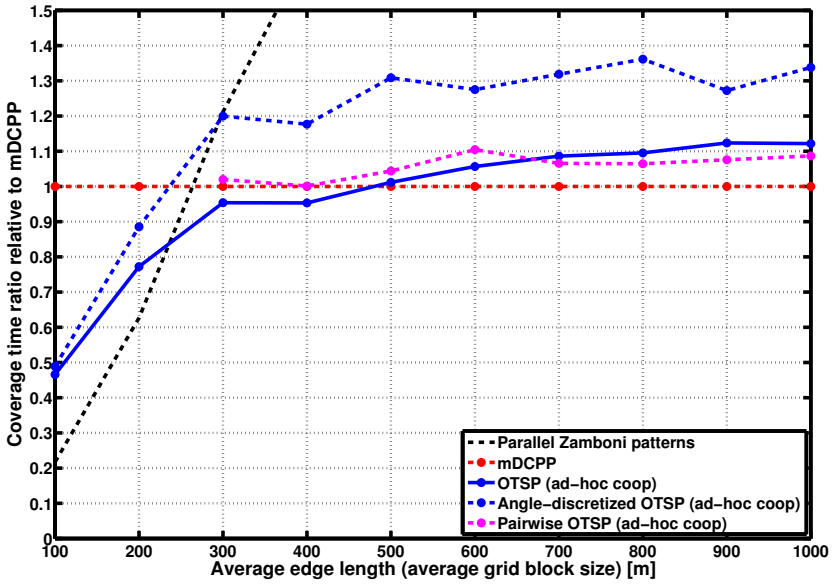
5.4 Conclusions and Future Work

Inspirations for future work primarily lie in the area of reasoned and tractable extension to multiple-vehicle teams, for all three topics herein. For road network coverage of Section 5.3 in particular, current strategies include ad-hoc geometric splitting, an auction-based scheme, and the use of the VRP generalization of the TSP where applicable. Better strategies, particularly applied to the sweep-sequencing SGTSP method proposed, should be sought. One possible idea is a hybrid mDCPP-SGTSP strategy in which alternating steps of auctioning edges greedily and solving the TSP on each vehicle's current set of edges are taken. Another avenue for future exploration is better formalization of environmental properties such as density as a means for coverage algorithm selection, beyond merely simulating several in parallel prior to physical execution to make this choice.

Efficient handling of occlusion by terrain obstacles is also an area of potential improvement, though if such occlusion can be modeled by fixed directional visibility known a-priori, the described strategies can be easily extended to capture this. Orbit-based coverage described in Section 5.1 already provides views of every point from every direction. Probability updates and generation of trajectories for information-theoretic search planning as in Section 5.2 may be modified to include occlusion in the detection model so that unoccluded perspectives will be naturally sought out, while occluded locations will not receive reduced probability from what would otherwise have been observation-triggered updates. Finally, instances of road network coverage of Section 5.3 utilizing a *generalized* TSP may be readily modified to incorporate occlusion by eliminating occluded cluster elements (e.g. points around angle-discretized orbits or a particular sweep direction for road segment sweeping).



(a) Coverage execution time relative to Zamboni



(b) Coverage execution time relative to mDCPP (note that pairwise is intractable in dense environments)

Figure 5.21: Coverage execution time comparison between the basic orbit TSP (OTSP) method and the two proposed extensions (angle-discretized and pairwise-orbit), alongside Zamboni patterns and the greedy mDCPP competitor algorithm for reference, for 3 UAVs running in parallel. Note that the angle-discretized strategy is always worse than the basic one and that the pairwise method is not always better than the basic one due to suboptimal heuristic solutions for the large resulting graph.

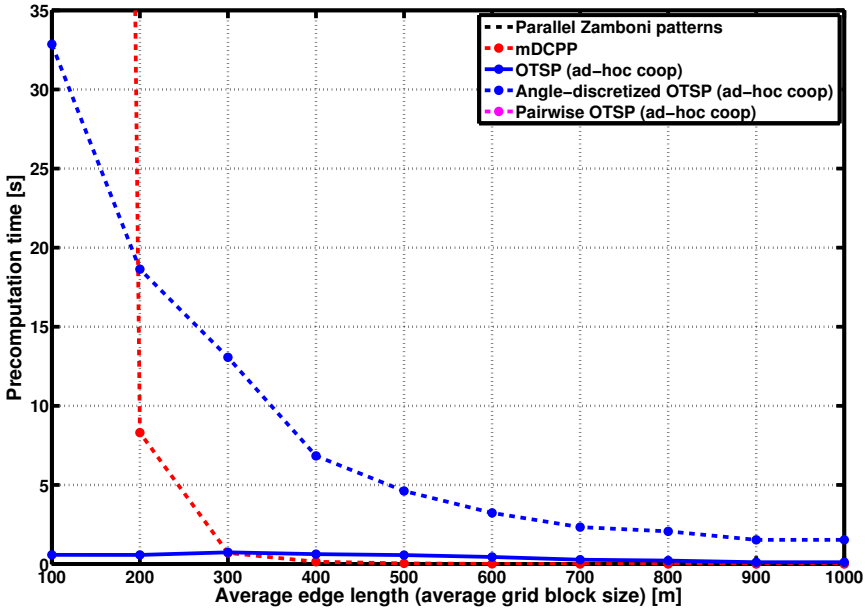


Figure 5.22: Coverage execution and precomputation time comparison between the basic orbit TSP (OTSP) method and the two proposed extensions (angle-discretized and pairwise-orbit), alongside Zamboni patterns and the greedy mDCPP competitor algorithm for reference, for 3 UAVs running in parallel. Note that the precomputation time for Zamboni is negligible, while pairwise is non-realtime and lies beyond this graph.

CHAPTER 6

Guaranteed Search and Recapture

This chapter considers the scenario of an evasive (also referred to as adversarial or simply fleeing) target within an urban-like road network and the need to locate it with one or more UAVs, with the foremost goal being to guarantee its capture (or provide a pre-mission determination that this cannot be assured) and an immediate secondary one being to minimize the time necessary to do so. First, Section 6.1 provides the statement of this task more precisely and motivates the choice of strategy. Section 6.2 then presents an algorithm and simulation results for guaranteed search for targets having infinite speed (or simply unmodeled motion). This is built upon in Section 6.3 to develop two strategies for guaranteed search (or, in a local form, recapture) applicable to targets having some known bound on their speed. Finally, Section 6.4 summarizes these results and suggests several avenues for extending these ideas using alternative abstractions. This work represents some of the first efforts to apply pursuit-evasion formalisms to search with physical UAV teams, particularly in realistic evaluation using motion models and control interfaces corresponding to real fielded vehicles.

6.1 Problem Formulation

For the purposes of this problem, a target is assumed to lie wholly within the edges of the graph underlying a physical road network, corresponding to presence on road segments. Intersections (graph nodes) are treated as infinitesimal, and any physical

extent may in practice be assigned to lie on an incident edge. UAVs are of course not constrained to the road network themselves (and as such, this task is distinguished from most pursuit-evasion abstractions) but are free to move only along the graph if this provides conceptual simplification useful to generate tractable search strategies. This notion inspires the proposed strategies, built upon sweeps of graph edges but not requiring adherence to the graph structure otherwise, that may not be optimal in terms of the number of agents required or time to capture or may declare that no capture guarantee is possible when in fact some sequence of motions exists to do better. This is taken as a tradeoff against the ability to rapidly provide a solution when the only immediate alternative is a large combinatorial search, typically within a game tree of pursuer and evader actions, to a horizon that is unlikely to reach the necessary endgame states to provide an equivalent capture guarantee.

Targets are assumed to be adversarial, which is here taken to mean that they may (and are expected but not required to) move within the limits of their ability so as to avoid capture completely or maximize the time until captured. That is, targets may be expected to perform actions that are most inconvenient to the searchers, typically modeled as minimax reasoning. In the abstract search and pursuit-evasion literature, this is also commonly extended to include omniscience, or total awareness of both the strategy and state of the pursuers. Though perhaps occasionally a desirable property to provide additional confidence in a search strategy, this is likely an unrealistic concern in typical practice for aerial search and will necessitate more conservative strategies than actually required. One side-effect of the conservatism of the strategies considered herein is that target omniscience is permissible.

Another parameter is that of target motion. Within the search literature, "adversarial" is also often taken to mean omnipotent in that targets may move instantaneously in any direction at any speed. Clearly, this requires highly conservative strategies that constantly corral targets into area subsets that are eventually reduced to a point. In practice, assumptions regarding limits on target capability are highly reasonable, including most prominently a bound on achievable speed as well as possibly limits on changes in direction (e.g. rate of turning). Nevertheless, treatment of infinite-speed targets provides substantial conceptual simplification and may be appropriate when little is known about the target. Additionally, infinite-speed targets have two other properties that must be noted. The first is that environments containing them must be bounded, since otherwise targets will immediately escape. Physically, this could be achieved by a perimeter cordon or deliberately "retreating" and extending the environment to a feasible set of choke points. Second, the presence of infinite-speed targets introduces time-invariance into the search task, since however fast a team of searchers may move, targets will always move faster. This is

highly appealing for teams of small UAVs that may both be relatively slow-moving as well as be constrained by long repositioning times caused by limited maneuverability.

An additional critical aspect to define is that of the meaning of guaranteed capture. As the means by which a UAV is assumed to determine the presence or non-presence of a target (the act of detection) is through the use of a target detector (automated or human) operating on data provided by a camera-like sensor, false positive or negative detections are inevitable, precluding a perfect guarantee. Instead, using some stochastic model of the detection system, a probabilistic guarantee of capture may be provided. For the purposes of this work, the simplification is made that if a UAV's field of view passes over the location of a target, then it is detected with certainty (false positives are assumed to be unimportant, if reported detections are thoroughly vetted before terminating the search). The proposed strategies may be readily extended to some other forms of detection certainty such as minimal dwell time by, for instance, regulating forward speed.

Closely related is the necessity of being certain that a location was in fact in view before it can be treated as clear and free of targets. With small UAVs having large state error, that are readily subject to disturbances from wind, and which possess limited control authority, this too is clearly a challenge. The primary proposed means by which to address this is to simply be very conservative in placing the field of view—keeping the target near its center at all times so as to maximize the likelihood that it is in fact in view despite any state error or disturbance. This may be assisted by ensuring that the field of view is as large as possible, for instance by maximizing altitude while still maintaining sufficient resolution to distinguish targets. Additionally, trajectories may be designed so as to minimize aggressiveness by minimizing acceleration and curvature so as to provide lower-level controllers as much headroom as possible. All of these are implemented to some extent in the simulations presented via variable standoff from desired viewing locations and the use of minimal models for aircraft motion. Further means to increase robustness remain open questions for future work.

The motion sequences produced by the proposed strategies are search schedules generated prior to execution, which is a side-effect the adversarial target assumption and an assumed lack of input regarding target location during the search. Executed to completion, these schedules will provide an assurance that no target lay in the environment unless one was detected along the way, at which point the search may be terminated or continued to seek additional targets. Upon detection, the search is assumed to be concluded by reporting target location or with a system transition to a pursuit mode. As implemented here, these search schedules seek first to provide a guarantee of capture, minimize the number of agents required for this guarantee,

and then as a secondary objective attempt to minimize the search mission time. This is assumed to be reasonable if the number of UAVs available is the greatest constraint—either simply limited in number or an expensive resource—and time to capture is less important given the existence of a guarantee. Given additional agents, the presented strategies can reduce search time by simply running portions of the search in parallel. An alternative approach advocated by Hollinger [61] is to use any redundant agents to perform a simultaneous efficient probabilistic search (e.g. minimizing time to expected detection).

6.2 Infinite-speed Targets in Bounded Environments

For targets known to be extremely fast or agile relative to UAV searchers, UAVs with slowly-responding high-level autopilots, targets of grossly unknown type (e.g. human vs. vehicle) providing no meaningful speed bound, or any occasion for which mission time invariance is otherwise desirable, modeling targets as having "infinite motion" abilities may be appropriate. Explicitly, targets are assumed to be able to move at any, including infinite, speed and may move in any direction within the graph: instantaneously reversing direction on edges or choosing any branch at a junction, regardless of incident angle. Given this formulation, the area possibly containing targets can be represented as contamination that diffuses infinitely rapidly, and it must be via placement of searchers at nodes or on edges that (re-)diffusion is prevented, while the contaminated area can only be shrunk by pushing forward ("sweeping") along contaminated edges to compress it inward.

6.2.1 Basic strategy

For abstract undirected graphs and such edge-bound evaders, a very straightforward guaranteed search algorithm long studied in the abstract graph search literature and well-described by Barrière et al. [13] exists for the special case of trees. Intuitively, the search begins at a node chosen to be the root and progresses as a typical depth-first tree traversal, sweeping edges during descents. The only source of any (conceptual) complexity is that nodes with more than one subtree (not lying along a simple path) must be guarded while subtrees are searched, lest contamination from unsearched subtrees return to previously swept subtrees via that node. This basic algorithm is stated formally in Algorithm 6.1, which returns the *search number* (number of agents required) of the graph and a valid search schedule. In search theory terminology, this

schedule produces a connected search, in which the cleared area grows monotonically without any recontamination.

```
Input: root of tree to clear
Output: search schedule and search number (required team size)

search_schedule = [ ]
Function search_number = clear_tree(root, parent)
begin
    if parent ≠ NULL then
        search_schedule.append(SWEEP[parent,root])
        search_number = 1
    else // handles trivial case of single-node tree
        search_number = 0
    if root.children.size > 1 then
        search_schedule.append(BEGIN_GUARD[root])
    foreach child ∈ root.children do
        search_number = MAX(search_number,clear_tree(child, root))
    if root.children.size > 1 then
        search_schedule.append(STOP_GUARD[root])
        search_number = search_number+1
    return search_number
end
```

Algorithm 6.1: Guaranteed tree edge search for infinite-speed targets

Note that what the tree actually represents is a topological hierarchy indicating a dependency tree capturing which nodes must be visited before others if starting from that root. Within this topological sort, searchers are in fact free to choose the sequence in which graph elements are actually visited, for instance in what order the subtrees of a node are searched. This admits an important but likewise straightforward optimization for nodes with multiple subtrees that may be made by sorting the subtrees in increasing order of search number and searching the largest last. At the point at which the last, largest subtree is about to be searched, all other subtrees have been cleared, and the guard may be relieved once the sweep down the last subtree has begun (or, as a special case, as long as it does not lose continuous view the graph to do so, the agent that was guarding the node may itself perform this sweep). Note that this optimization and the notion of a single search number for a given tree assumes that agents are interchangeable, that is may perform both guarding of nodes and sweeping of edges. If this is not the case, a set of guards equal to the search number minus one is needed, and one additional sweeper is always adequate.

Applying this basic algorithm to physical environments, especially to search by UAV teams, naturally requires several extensions. Most prominent is that physical environments rarely form convenient trees and instead contain cycles. Unfortunately, even computing the search number of general graphs is intractable, [78] and the direct generation of a search schedule in the same manner as trees cannot be applied. Instead, an intermediate solution similar to that proposed by Hollinger [61] is proposed. Specifically, a search schedule is generated in an anytime fashion by iterating over randomized spanning trees of the entire graph, computing the search number for each as the that of the tree plus the number of additional guards required to break all cycles by placing a guard at one end of each edge discarded by that spanning tree, storing the one with the running best search number, and returning this as the solution for the original graph once an iteration count or time limit is reached. This algorithm is briefly summarized in a high-level fashion in Algorithm 6.2.

Next, this abstract search schedule must be applied to a team of UAV searchers. Depending on the vehicles available, this may be performed fairly readily. The search may be seen to have two fundamental primitive actions: guarding of nodes (intersections)and sweeping of edges (road segments). To perform a guard action, a UAV (or an assemblage of several) must be able to loiter over an area at least as large as an intersection within the environment and provide consistent viewing of it. For rotorcraft, this is straightforward given the ability to hover. Fixed-wing aircraft with either gimbaled or fixed side-pointing cameras can, with a sufficiently large field of view and an appropriate pointing angle, orbit around the point to be guarded to keep it in view. Fixed-wing aircraft with downward or fixed-angled cameras pointing along the forward axis (e.g. downward or ahead) are more challenging and must be able to either loiter (e.g. fly in a tight figure-eight pattern) while maintaining a sufficiently large viewing area or join others in doing so such that the union of their sensor footprints covers the guard point at all times. Such aircraft may also simply be disqualified from guarding if sufficiently many other vehicles capable of guarding are available. Likewise, to perform a sweep action, a UAV must be able to move its field of view to follow the entire path of an edge. Edges comprised of a simple line segment are clearly straightforward in that the UAV need merely line up with the start of the edge (with a planar offset as necessary to position the field of view) and move forward along the edge. Edges whose path contains complex curvatures cannot in general be followed by vehicles with turning-radius constraints and may be treated as containing intermediate nodes so that the curvature of any single edge does not exceed these constraints. Two cases then typically arise. For UAVs with gimbaled or fixed side-pointing cameras, a guarding orbit around these intermediate nodes may be briefly entered until aligned with the next edge in the path, preventing loss of view.

```
Input: graph to clear and index of root at which to begin
Output: set of cycle-breaking guard nodes and search number and schedule
        for remaining tree

guard_nodes = [ ]
search_schedule = [ ]
Function search_number = clear_graph(graph, rooti)
begin
    best_searchnum = ∞
    itercount = 0
    while itercount < max_iters AND time_elapsed() < max_time do
        // e.g.  randomly ordered Kruskal's
        tree = get_random_spanning_tree(graph)

        guards = [ ]
        foreach edge ∈ graph do
            if edge ∉ tree then
                // This may also be chosen to (heuristically)
                // optimize impact of guard presence on search,
                // e.g.  i = argmax_j edge.endpoint[j].degree
                i = rand({0,1})
                nexus = tree.nodes[edge.endpoint[i]]
                parent = tree.nodes[edge.endpoint[1-i]]
                new_leaf = duplicate_node(nexus)
                tree.insert_node(new_leaf)
                tree.insert_edge(parent, new_leaf)
                guards.insert(nexus)

        // additional parameter to clear_tree:
        // neglects guard placement at already-guarded nodes
        searchnum = clear_tree(tree.node[rooti], guards)

        if searchnum+guards.size < best_searchnum+guard_nodes.size then
            best_searchnum = searchnum
            guard_nodes = guards

        itercount = itercount+1
    return best_searchnum
end
```

Algorithm 6.2: Anytime guaranteed graph edge search for infinite-speed targets

UAVs having fixed downward or forward-pointing cameras may not be able perform such a re-alignment while also loitering at edge endpoints and require the assistance of an additional agent to guard these intermediate nodes during this interval. A convenient conceptual simplification in this case may be to assign a sweeper-guard pair of agents as a virtual sweeper that is able to sweep edges of arbitrary geometry, while internally performing slinky- or leap-frog-like maneuvers to accomplish this.

Additional properties of UAV searchers may also be considered. The first is that unlike ground agents performing a similar graph clearing search, they are not themselves constrained to the graph and may move between locations arbitrarily (subject to motion constraints), permitting substantial mission time optimization. For a guard moving between locations, this simply implies choosing a minimal-length trajectory between the two points. Sweep actions have greater freedom in that subtree search ordering may be chosen to minimize intermediate maneuvering. Additionally, where the opposite end of an edge in the search tree is not a source of diffusing contamination (e.g. it is a leaf node), the direction of the sweep may also be chosen to minimize overall maneuvering. Likewise, given a homogenous team, assignments and transitions between guard and sweep roles may be chosen to minimize flight distance given instantaneous poses at the time of transition. A further side effect of the lack of graph-adherence is that the team of searchers may begin the search from any point in the graph (that is, they may choose the root node rather than it be given as the "entry point" as would be the case for ground-based searchers in graphically modeled environment). This may be addressed by wrapping Algorithm 6.1 in an additional loop that evaluates search number and predicted (post-optimization) mission time for each node as the root, choosing the best. Likewise, this may be built into the internal loop of Algorithm 6.2.

Another distinction between UAVs and graph-constrained searchers is that aerial searchers may view multiple portions of the graph at once, and as a result could conceivably perform any combination of several sweep or guard operations at once with an appropriately chosen (possibly joint) trajectory. Given the nontriviality of detecting or performing such actions, these opportunities are neglected by the proposed strategy and remain an avenue for further exploration.

6.2.2 Extensions and demonstration

As applied to a team of UAV searchers, a progression of extensions to the basic described strategy is considered. In roughly increasing order of conceptual and implementation complexity, these are now described, as motivation for the demonstrated algorithm used for experimental comparisons.

- **Sort subtrees.** Subtrees of a given node may be searched in increasing order of search number to maximize the number of free agents available for the largest subtree, including in particular the guard, if present, at the parent node. Given the choice of homogenous teams in the experiments performed herein, this has the specific effect that the parent guard—often the nearest vehicle to the initial endpoint of the last subtree—may transition smoothly and immediately to a sweep role.

- **Preposition sweepers.** Conceptually, a sweeper dependent on a guard must wait for the guard to begin its loiter before the sweep can begin. Implemented naively, this results in lengthy idle periods between the activation of a sweeper and the start of the actual sweep as it maneuvers into position. By estimating the maneuvering time required for a guard and dependent sweeper to begin their roles, the sweeper may prematurely begin its maneuver towards the target edge in anticipation of the guard's arrival before it, allowing both to move in parallel and eliminate such idle periods.

- **Parallel searches.** When sufficient team size allows it, subtrees may be searched in parallel. Doing so effectively is difficult for several reasons, however. First, unless an additional subtree is begun only if agents sufficient to meet the sum of the search numbers of the currently-searched trees are available to them, deadlock may occur wherein a guard is needed by two subtrees at once but can only be provided by abandoning one of the searches. Violation of this requirement is naturally tempting in the case of deep subtrees with many agents only required late in its search, leaving these idle until then. Further, a choice must be made (with accompanying look-ahead) whether to commit spare agents—and how many to provide—to an existing subtree or to begin a new one, also requiring determination of how many to provide to each.

- **Action optimization.** Subject to the topology of the dependency tree, edges may be swept by any available agent, in any order, and (when permissible) in either direction. This produces an assignment, sequencing, and direction-choice optimization problem that may be solved to minimize total mission time. Unfortunately, this represents a large constrained combinatorial optimization and is intractable to solve exactly for any substantially-sized environment. Partial and heuristic optimizations provide an intermediate solution, and one such method is considered here.

The strategy evaluated here represents an essential demonstration of a tractable subset of the above extensions to the basic algorithm. Specifically, sorting of subtrees,

prepositioning of sweepers, and limited action optimizations are performed, without parallel searches of peer subtrees. The optimization is motivated by the simplifying assumption that the optimal set of actions and assignments thereto within a subtree is optimal globally. That is, optimizations may be made in a single-step fashion independently within each subtree, without inter-subtree scheduling or parallelism. Therefore, action optimization is only applied to leaf subtrees (i.e. comprised of only a path) of a node, and larger subtrees are searched recursively, independently, in increasing order. Given the presence of a guard above the leaf subtrees in question, this optimization fundamentally solves a coverage task on the paths forming these leaf subtrees, as no contamination can re-enter a path once searched from either direction. Thus, the SGTSP algorithm described in Section 5.3 may be directly applied, taking each subtree path to be a single contiguous edge, to produce a fully time-optimal agent-assignment and direction-annotated sequence among these subtrees. Excluding sweeper prepositioning, the overall algorithm is summarized in Algorithm 6.3. This strategy is chosen as a reasonable balance of complexity and approximation in that it is readily implemented while capturing key extensions. A typical example of its execution on a small demonstration environment is shown in Figure 6.1.

6.2.3 Experimental results

To evaluate this strategy and for comparison, repeated realistic simulations were performed on varying maps and with varying-sized teams. As in the rest of this thesis, UAV teams were assumed to be comprised of fixed-wing aircraft approximated by a Dubins vehicle with fixed, side-angled cameras with parameters similar to that of the widely-fielded AeroVironment Raven used in field experimentation described in Section 4.1.

Environments were selected as in Section 5.3: a spectrum of randomly-generated maps formed from perturbed Manhattan grid subsets intended to realistically mimic physical road networks, parametrized by overall size, fraction of the entire grid present, and block size (edge length). Once again, the combination of edge length and fraction of edges removed conceptually correspond to environment density, and environments with edge lengths within the range 100-300m will be referred to as urban, 300-700m as suburban, and 700m-1km as rural. Examples of such maps for each of these categories may be reviewed in Figure 6.2. As this and Figure 6.8 show, this encompasses a wide variety of environment classes and forms. Nevertheless, some amount of bias must be assumed, primarily due to either consistent edge lengths or the construction from an underlying Manhattan grid. One clear omission in coverage by this generation scheme is long, winding paths that may frequently

```
Input: root of tree to clear
Output: search schedule and search number (required team size)

Function [search_schedule,search_number] = clear_tree_optimized(root,
parent)
begin
    if parent ≠ NULL then
        search_schedule.append(SWEEP[parent,root])
        search_number = 1
    else // handles trivial case of single-node tree
        search_number = 0

    if root.children.size > 1 then
        search_schedule.append(BEGIN_GUARD[root])

    foreach child ∈ root.children do
        [child_search_schedule[i],child_search_number[i]] =
        clear_tree_optimized(child, root))
    search_number = MAX(child_search_number[...],search_number)

    // Returns indices of argument array by increasing value
    child_ordering = sortby(child_search_number)

    // Returns indices of argument array matching condition
    leaf_subtrees = find(children[child_ordering] == 1)
    nonleaf_subtrees = find(children[child_ordering] > 1)

    // Conceptually, an indicator to invoke SGTSP-like coverage
    // optimization (from Section 5.3) at schedule execution-time
    // when this point is reached to compute distance-minimal
    // agent assignment, edge sequence, and edge directions.
    search_schedule = [OPTIMIZE_COVERAGE(children[leaf_subtrees])]

    foreach child_index ∈ nonleaf_subtrees except last do
        search_schedule = [search_schedule,child_search_schedule[child_index]]

    if root.children.size > 1 then
        search_schedule.append(STOP_GUARD[root])
        search_number = search_number+1
    if nonleaf_subtrees.size > 0 then
        search_schedule =
        [search_schedule,child_search_schedule[nonleaf_subtrees.last]]
    return [search_schedule,search_number]
end
```

Algorithm 6.3: Demonstrated leaf-optimizing guaranteed tree edge search for infinite-speed targets

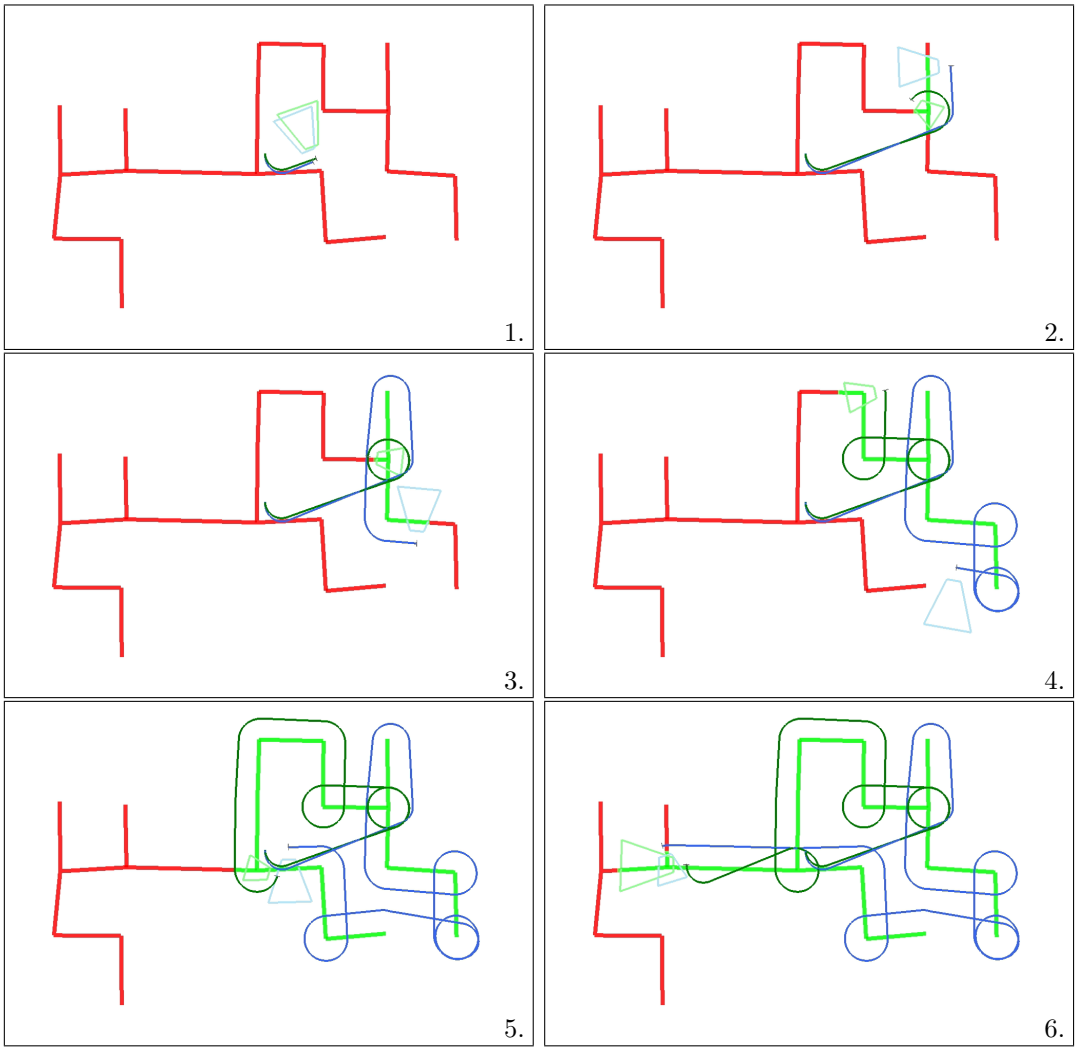
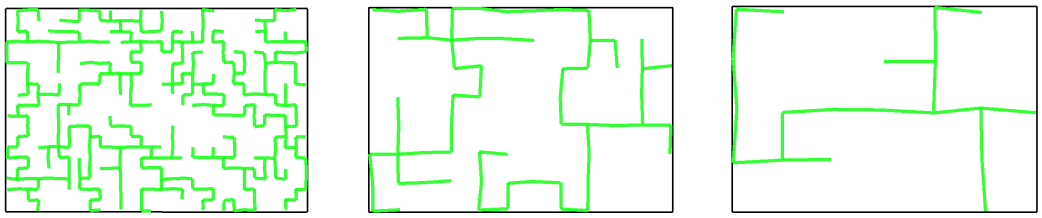


Figure 6.1: Snapshots of execution sequence of Algorithm 6.3 applied to a small demonstration environment, which is initially fully contaminated (designated as red rather than green area). Initially, (1) two UAVs start at a location near the center and fly towards a node at the top-right. Next, (2) one [green] UAV guards this node while the other [blue] sweeps a leaf subtree in an outward direction. While maintaining this guard, (3) the [blue] sweeper continues its computed optimal leaf subtree coverage pattern to sweep the other subtree in an outward direction. The guard [green] then transitions to a sweep role for the final subtree (4) while the original [blue] sweeper pre-emptively begins an inward sweep of the leaf subtree of the node at which the new [green] sweeper will terminate as a guard, at which it arrives (5) just after that node becomes guarded. Then, (6) the original [blue] sweeper smoothly continues its sweep into the last subtree, itself terminating as a guard at the next node while the other [green] UAV pre-positions itself for an outward sweep of this node's leaf subtree. This process continues until the entire environment is cleared.



(a) Urban (b) Suburban (c) Rural

Figure 6.2: Sample environments along the density spectrum, duplicated here from Figure 5.10 for convenience.

arise from roads built around hilly terrain or in older cities built up spontaneously prior to now-common planned street layouts. These may introduce varying effects, with no easily predicted overall bias. For instance, longer paths will possess reduced connectivity compared to a grid, simplifying the topology and reducing avenues for target escape. Simultaneously, however, bends in such paths may be more difficult to follow with fixed-wing aircraft and may substantially increase mission time over simple straight segments. Lack of parallel segments may introduce reduced opportunity for path optimization, more frequently requiring maneuvering rather than a chance to "fly-through," though jittering endpoints attempts to capture some of this effect. Additionally, at lower edge densities (fraction of grid present), an artificially large number of dead-end road segments is present compared to real-world environments. This is less of a concern than it seems since the search is performed after breaking cycles that are present using Algorithm 6.2, leaving (topologically) quite a few dead-ends. Indeed, results from many simulations across varying environment densities and sizes indicates approximately one dead-end child per node (that is, one leaf subtree) on average. True evaluation is best performed on samples of real-world areas of interest, but it is otherwise hoped that the diversity of environments generated via the described method is sufficient to capture essential trends and relative performance.

Except where noted, environments formed exclusively trees so that a fundamental comparison may be made between strategies without the additional complexity of cycle-breaking guards. For a given map, the search number is computed, and a team of this size is placed at a random launch location. Environments are treated as initially fully contaminated (equivalent to having been momentarily occupied by an infinite-speed evader without any searchers present to constrain it).

Four searcher-strategy combinations are principally compared:

1. The baseline case of a ground vehicle that is itself constrained to the graph,

in particular requiring it to backtrack to parent nodes before searching peer subtrees. No other constraints, such as on acceleration or turning rate, are imposed, allowing this to represent an idealized ground vehicle.

2. A UAV team comprised of agents as described executing the basic version of the proposed strategy, with subtree sorting, but without sweep prepositioning or optimization of assignment or sequencing beyond choosing the nearest agent to each required subtask at the time an agent is required.

3. An identical UAV team executing the proposed demonstration strategy, including subtree sorting, sweep prepositioning, and leaf subtree optimization.

4. An alternative UAV team also executing the proposed demonstration strategy, but comprised of agents without any motion constraints other than maximum speed, i.e. capable of unbounded acceleration and omni-directional motion. Lacking any form of holonomic constraint, this may be treated as a team composed of idealized rotorcraft, providing an upper bound on the performance of aerial teams.

For additional comparison, other guaranteed search strategies that do not consider the presence of a road network may be considered. For infinite-speed targets, clearly limited options exist, with two immediate strategies easily suggested. The first is to align agents adjacently to perform a parallel sweep directly across a rectangular convex hull of the environment, specifically its major axis to minimize the number of agents required, and proceed in the same manner of a classical 17th to 19th century battlefield march. Roughly, this will require $\lceil width/diameter \rceil$ agents, where $width$ is the length of the minor axis, and $diameter$ is the diameter of a single agent's sensor footprint. A second strategy is to encircle the are of interest, with agents placed around the circle so that their sensor footprints fully cover the perimeter of the circle, and then to move agents so as to reduce the size of the circle until a single collapsed point is reached. For an environment with major axis length of $length$, this requires $\lceil \pi length/diameter \rceil$ agents, which is necessarily more than the first strategy. Naturally, Manhattan or grid-like environments (of which variations of differing sparseness are evaluated here) are a particularly poor match for tree-based clearing strategies given their high connectivity and resulting large number of cycle-breaking guards required. For this special case of environment, yet another simple strategy is easily invented for further comparison: in an $m \times n$ grid of streets (with $n < m$), n agents may be used to follow the grid along its major axis, pausing (temporarily transitioning to guarding) simultaneously at each block

at intersections, while one additional agent sweeps across the minor axis to clear the intermediate trapped segments, for a total of $n + 1$ agents. For an environment with blocks of length $segment\_length$, the total number of agents required is $\lfloor width/segment\_length \rfloor + 1$.

First, the effect of environment density (taken here to vary block length while holding the fraction of edges selected from a dense grid at approximately 30%) on required team size and mission duration is evaluated. Several hundred simulations with random initial team locations and map seeds were performed for each of 10 levels of environment density within either a 3km $\times$ 2km or 6km $\times$ 4km area. For each run, the search number (required team size, and the same for strategies compared) and the time to mission completion was recorded for each of the four described search strategies. Precomputation time for all strategies was less than several seconds total, which is negligible in the context of such missions. In these comparisons, all agents have the same speed.

The effect of environment density on the search number is shown in Figure 6.3. It indicates, very promisingly, that for areas of this size, only rather small teams are required to complete the search once any necessary cycle-breaking guards are placed. Note that once the complexity of the graph and accompanying search number fall below a particular density threshold—occurring here at approximately 700m block lengths (roughly the transition from suburban to rural)—results become consistent. Similarly, the effect of density on mission duration is shown in Figures 6.4 and 6.5. As one may expect, motion-constrained UAVs running the basic (unoptimized) strategy perform worst. Applying the proposed optimizations in the demonstration strategy, approximately 1/4 of the difference between these and an ideal graph-constrained ground vehicle is eliminated at most densities. An idealized rotorcraft performs best of all, suggesting that using air vehicles with greater maneuverability or performing thorough path optimization to minimize the impact of any constraints may provide runtime comparable to that of an ideal ground vehicle. Further, since UAVs may in practice be substantially faster moving than their UGV counterparts, this gap may be eliminated almost entirely just by applying an appropriate speed scaling factor since mission time for these paths is almost exactly inversely proportional to speed.

Next, the effect of environment size on required team size and mission duration is considered. Similar simulations with environments of fixed-density with a block size of 300m (light urban to dense suburban) were performed with environments spanning from 1.2km $\times$ 0.8km to 6km $\times$ 4km. Once again, the search number and mission duration were recorded for each of the four strategies. Despite a small increase with environment size, precomputation time remained negligible.

Growth in the search number as environment size increases, shown in Figure 6.6

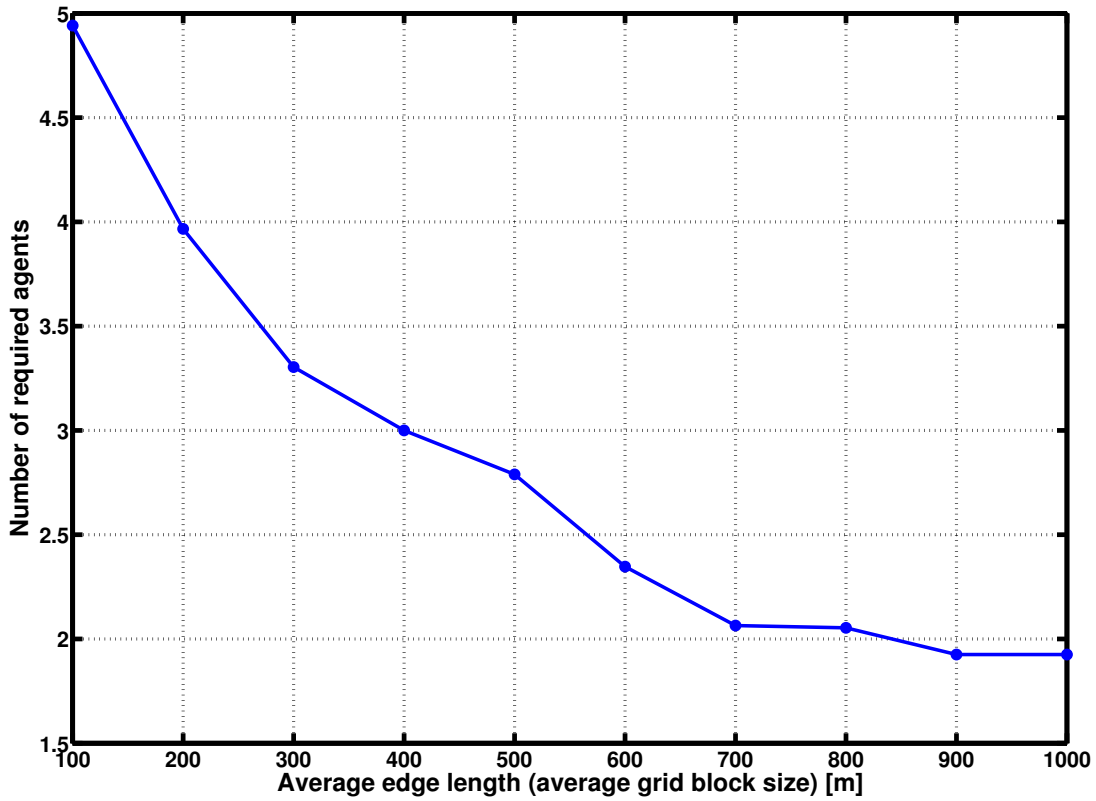


Figure 6.3: Required team size to perform a guaranteed search of a road network of fixed 6km × 4km size and 30% edge density but varying environment density (urban-length edges at left through rural-length edges at right). Even for dense areas, relatively few agents are required. Exploratory simulation on differently-sized environments suggests the trend remains roughly exponential in fall-off with merely a differing y-intercept. For smaller environments, in which edge length becomes comparable to environment dimension, the graph takes on a consistently simple structure, and the search number likewise becomes consistent. For comparison, $\lceil 4000/300 \rceil = 14$ agents are required (for an assumed 150m radius sensor footprint) if swept in parallel without regard to road network presence. Likewise, the simple Manhattan clearing strategy would require between $\lfloor 4000/100 \rfloor + 1 = 41$ (for edge length 100m) and $\lfloor 4000/1000 \rfloor + 1 = 5$ (for edge length 1000m) agents.

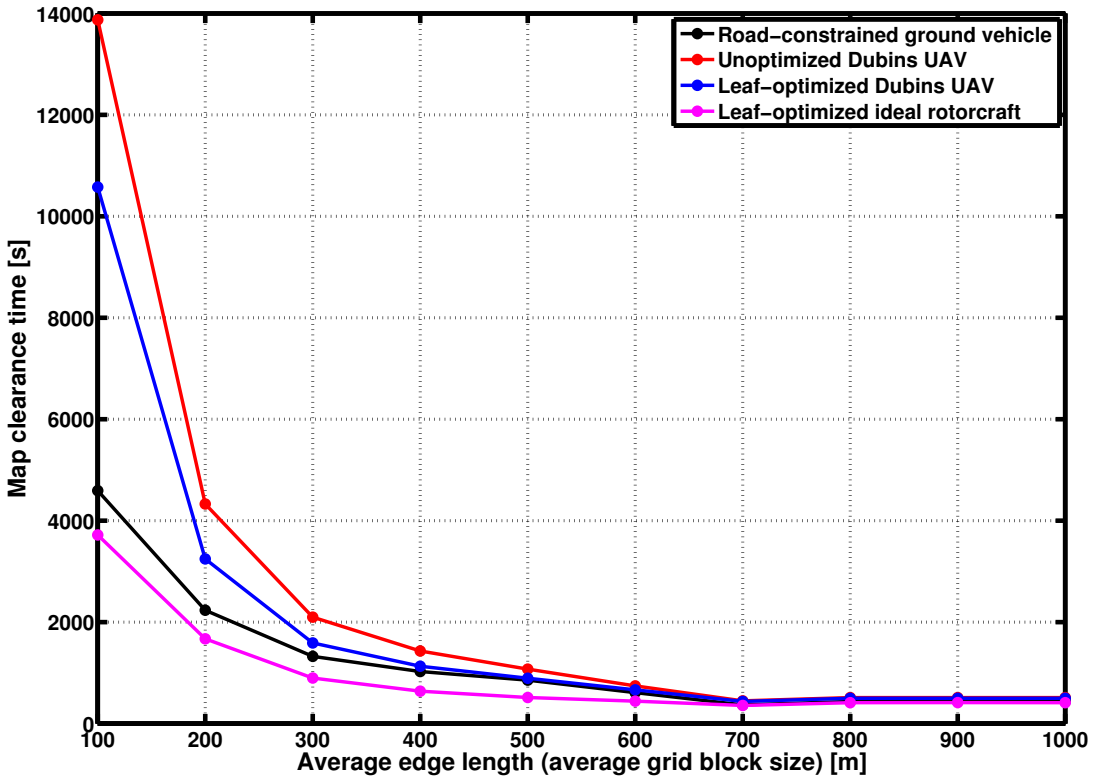


Figure 6.4: Absolute mission duration for guaranteed search of a road network of fixed 3km $\times$ 2km size and 30% edge density but varying environment density (urban-length edges at left through rural-length edges at right). For most densities, the proposed leaf-optimizing demonstration strategy eliminates at least 1/4 of the mission duration difference between the baseline UAV strategy and that of an ideal ground vehicle, which is itself outperformed by an ideal rotorcraft. Note that below a density threshold (above 700m blocks, or 30% of environment dimension), the graph takes on a consistently simple structure, and results become consistent. Exploratory simulation on larger environments indicates these trends and relative performance persist.

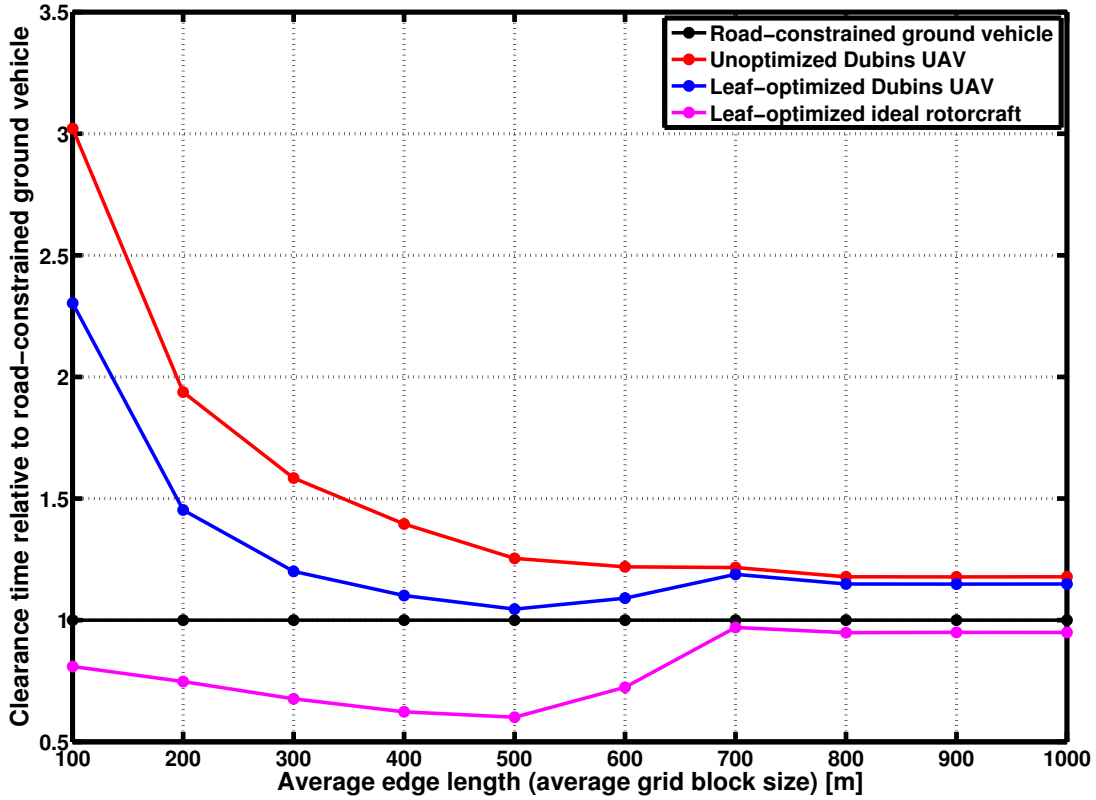


Figure 6.5: Another view of Figure 6.5: mission duration relative to ideal road-constrained ground agents for guaranteed search of a road network of fixed 3km × 2km size and 30% edge density but varying environment density (urban-length edges at left through rural-length edges at right). The proposed leaf-optimizing demonstration strategy eliminates about 1/4 of the mission duration difference between the baseline UAV strategy and that of an ideal ground vehicle, which is itself outperformed by an ideal rotorcraft. Note again that below a density threshold (above 700m blocks, or 30% of environment dimension), the graph takes on a consistently simple structure, and results become consistent. For high-density environments (at left), maneuvering time dominates, benefiting ideal ground vehicles. As density decreases (at middle), long paths to leaves requiring backtracking for ground vehicles dominates, benefiting aerial vehicles. Finally, for low density (at right), traversal time dominates, limiting the benefit of maneuver-minimizing path optimization. Exploratory simulation on larger environments indicates these trends and relative performance persist, with the leaf-optimizing strategy much more closely approaching the performance of the ground vehicle.

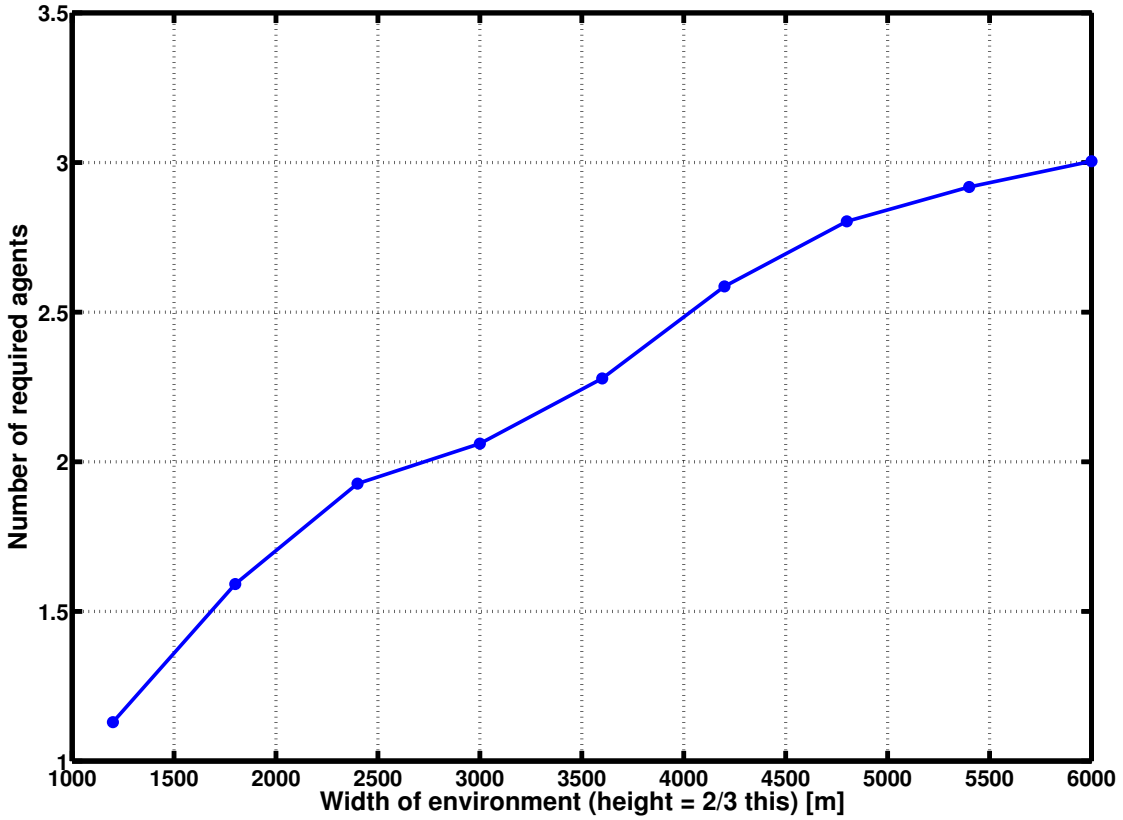
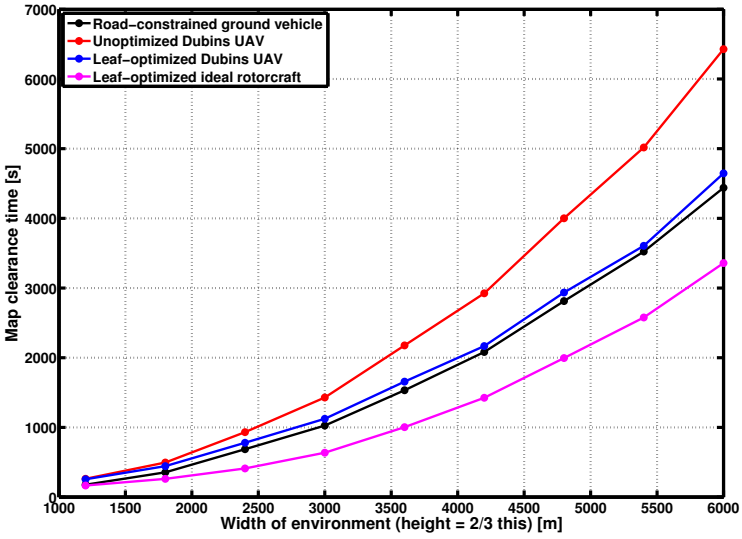
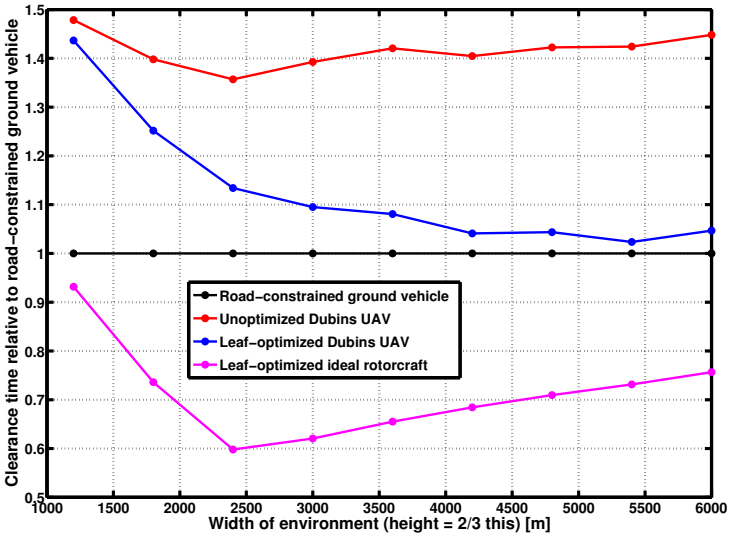


Figure 6.6: Required team size to perform a guaranteed search of a road network of fixed environment density (400m edge lengths) and 30% edge density but varying size, from 1.2km × 0.8km to 6km × 4km. Even for large areas, relatively few agents are required, and the growth rate is low. For comparison, if swept in parallel without regard to road network presence (whose curve not shown), between $\lceil \frac{2}{3}850/300 \rceil = 2$ (at far left) and $\lceil \frac{2}{3}6000/300 \rceil = 14$ (at far right) agents are required. Similarly, the simple Manhattan clearing strategy (also not shown) would require between $\lfloor 800/400 \rfloor + 1 = 3$ and $\lfloor 4000/400 \rfloor + 1 = 11$ agents.



(a) Absolute mission time



(b) Mission time relative to ideal road-constrained ground agents

Figure 6.7: Comparison of mission duration to perform a guaranteed search of a road network of varying size but fixed environment density (400m edge lengths) and 30% edge density using each of several strategies and agent types. Note the increasing relative performance of the proposed demonstration strategy against an ideal ground with increasing size, as well as the slowly worsening relative performance of an idealized rotorcraft, both likely due to increased traversal distances.

is relatively slow—apparently sublinear—once any necessary cycle-breaking guards are placed. Similarly, mission duration as a function of environment size is given in Figure 6.7, which suggests a slow relative convergence in search time among all but the basic UAV strategy. This may be explained by the overall increased scale of the search tree–both its depth and number of branches. Increased depth reduces the relative performance of ground vehicles due to the need to backtrack during search, and additional branches provide more opportunity for leaf subtree optimization in the proposed demonstration strategy. Meanwhile, the increased scale of the environment—simply, the distance between any two points at which a task must be started—reduces the relative benefit of unconstrained aircraft as linear traversal distance outweighs maneuvering time.

Finally, to evaluate the effect of the presence of cycles in an environment's graph representation, the method described in Algorithm 6.2 was applied to environments of varying edge density, recording both the number of additional cycle-breaking guards required as well as the search number of the remaining tree. Once again, for the set of environments considered, edge density is the fraction of edges in the complete grid graph upon which a given environment is based that are present. Samples of environments evaluated are shown in Figure 6.8.

Growth in the search number (due primarily to the presence of cycles) with increasing edge density is shown in Figures 6.9 through 6.11 for environments with edge lengths of 200m, 450m, and 700m respectively. In these plots, the total required number of agents to perform a search of the graph (and the number of cycle-breaking guards this includes) is compared against open-area parallel sweeping and Manhattan grid sweeping defined at the start of this section. For edge lengths of 200m, graph search is preferable for densities up to 0.5, at which point 7.5 agents are required on average (4 of which are guards), beyond which environments are best treated as open areas. For edges of 450m, graph search is preferable for densities up to 0.7, at which point 4.9 agents on average are required (2.5 of which are guards), beyond which Manhattan grid sweeping is fastest. Finally, for environments with edges 700m in length, graph search is best up to densities of 0.8 (3 total agents and 1.2 guards), after which Manhattan sweeping is again fastest. Overall, this suggests that for up to roughly dense suburban environment densities, graph search is viable and competitive. Note that if the environment were less grid aligned, Manhattan sweeping would be inapplicable, and graph sweeping would become preferable for even higher densities.

Similarly, growth in the search number (again primarily to the presence of cycles) with environment size but fixed edge density is shown in Figures 6.12 through 6.14 for varying sizes and fixed edge density of 0.5, again for environments with edge lengths

of 200m, 450m, and 700m respectively. For edge lengths of 200m, graph search is preferable for environments up to approximately 6km^2, above which treatment with open-area sweeping is better. For edge lengths of both 450m and 700m, graph search dominates for all evaluated sizes (up to 50km^2), though it is clear that for still larger environments, Manhattan grid sweeping will eventually become superior. This further suggests that for the edge density considered representative of physical environments, graph search is competitive for small urban areas and large suburban or rural ones.

For contrast, Figures 6.15 through 6.17 depict the same growth in search number with environment size, but at a much higher fixed edge density of 0.8. For urban areas (200m edge lengths), open-area sweeping dominates, including over Manhattan sweeping, suggesting that such dense areas are best treated as densely-connected open areas rather than as possessing a graphical structure. For suburban and rural areas (450m and 700m edge lengths), graph search dominates for small environments only. This retains usefulness for the proposed strategies to recapture bounded-speed targets described in the following section, but suggests the need for a less conservative abstraction in dense graphs.

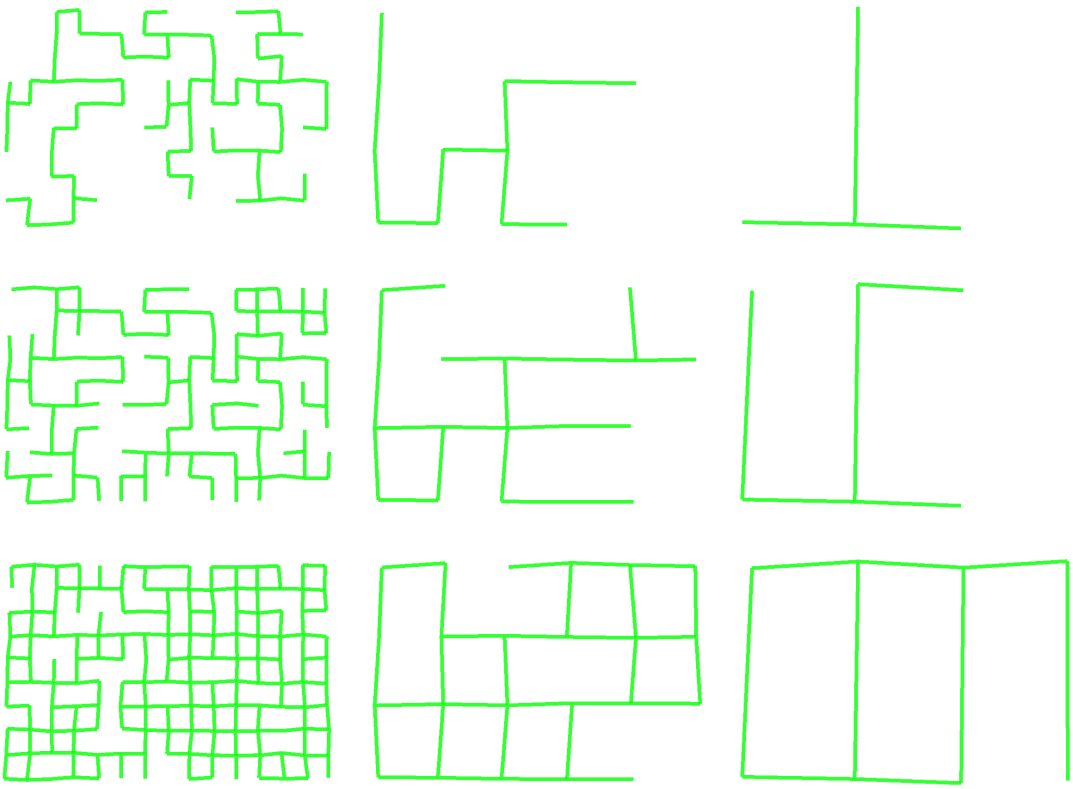


Figure 6.8: Examples of environments used for evaluation of the number of additional guards needed to break cycles before applying tree search, as generated by Algorithm 6.2. From left to right, the columns show environments containing 200m, 450m, and 700m edge lengths respectively. From top to bottom, the rows show environments with block density (fraction of edges from the complete grid present) of 0.2, 0.5, and 0.8. For reference, the center row contains environments typically considered canonical "urban," "suburban," and "rural," from left to right.

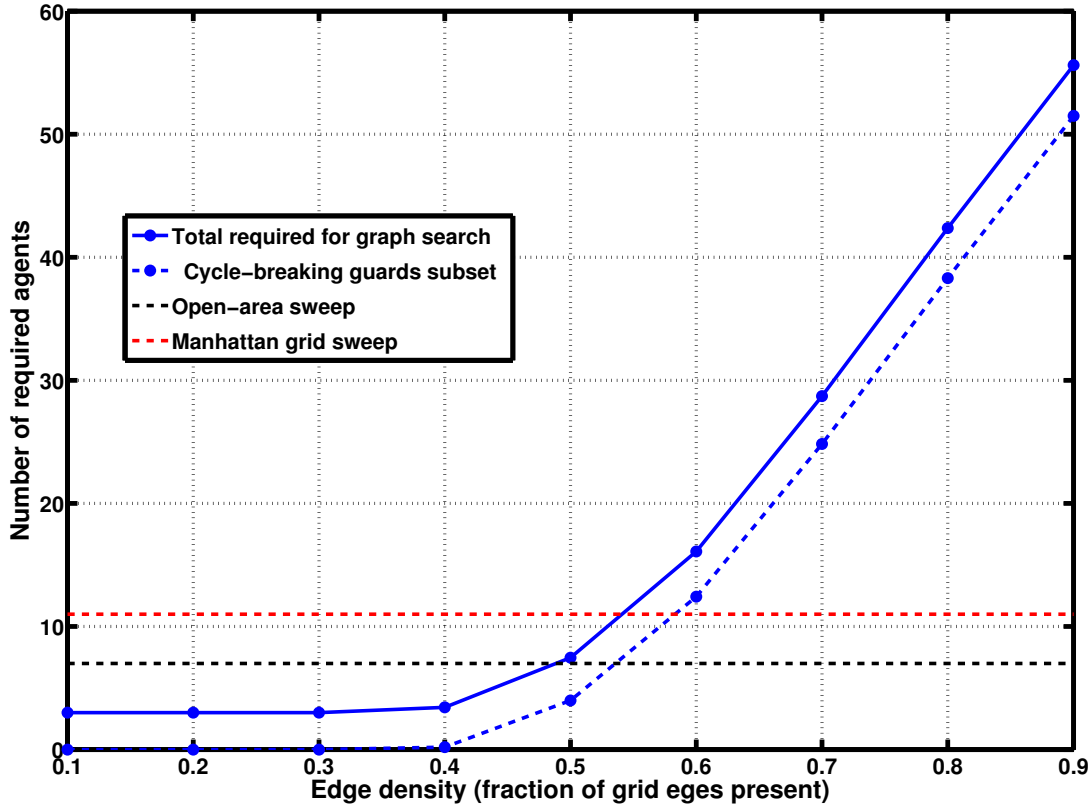


Figure 6.9: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network of fixed 3km $\times$ 2km size and 200m edge lengths but varying edge density, from 10% to 90% of all edges in the underlying grid present. For reference, note that environments with these parameters and edge density 0.5 is considered canonically urban in this document.

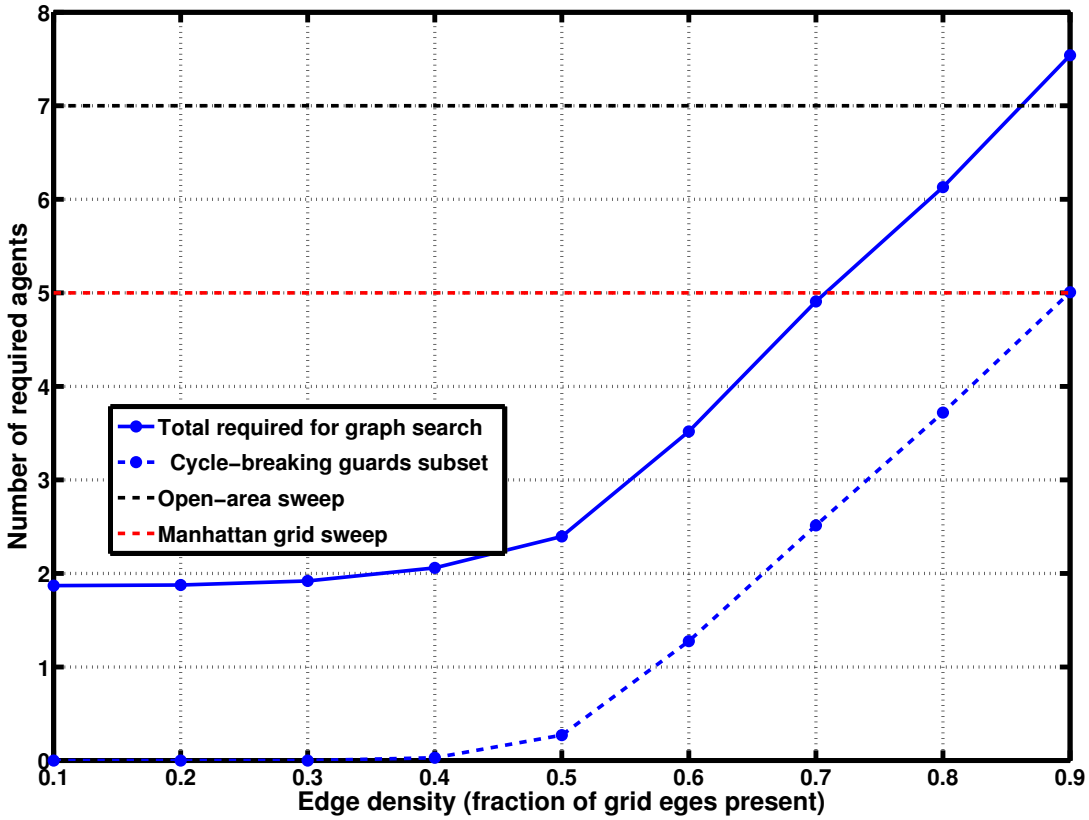


Figure 6.10: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network of fixed 3km × 2km size and 450m edge lengths but varying edge density, from 10% to 90% of all edges in the underlying grid present. For reference, note that environments with these parameters and edge density 0.5 is considered canonically suburban in this document.

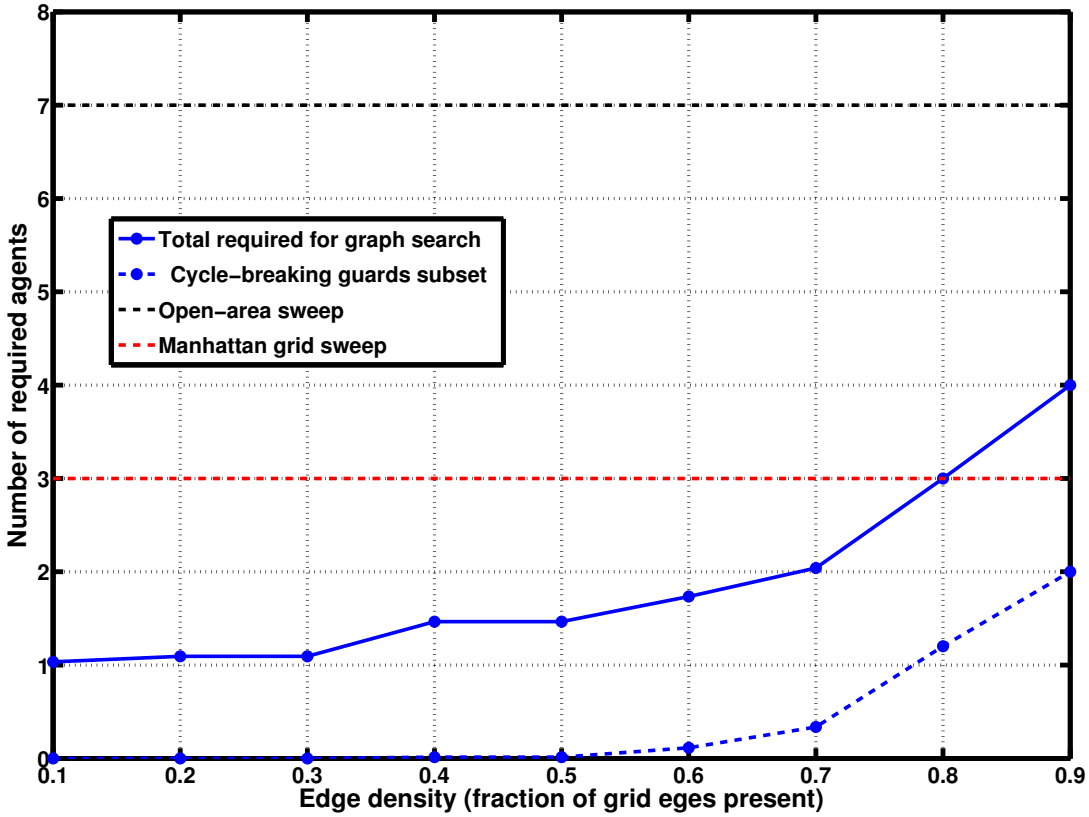


Figure 6.11: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network of fixed 3km × 2km size and 700m edge lengths but varying edge density, from 10% to 90% of all edges in the underlying grid present. For reference, note that environments with these parameters and edge density 0.5 is considered canonically rural in this document.

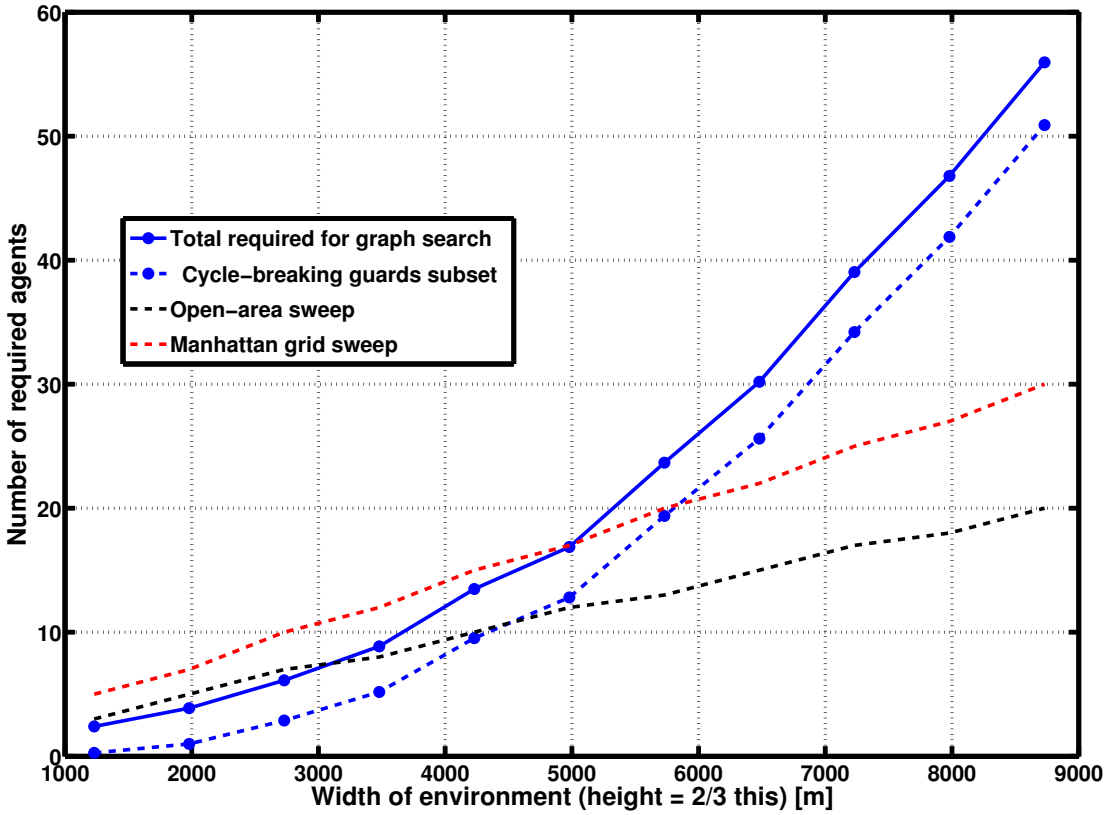


Figure 6.12: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network with fixed 200m edge length and edge density of 0.5 (50% of edges from the underlying grid are present), but varying size from 1.2km $\times$ 0.8km to 8.7km $\times$ 5.8km. For reference, note that environments with these parameters are considered canonically urban in this document.

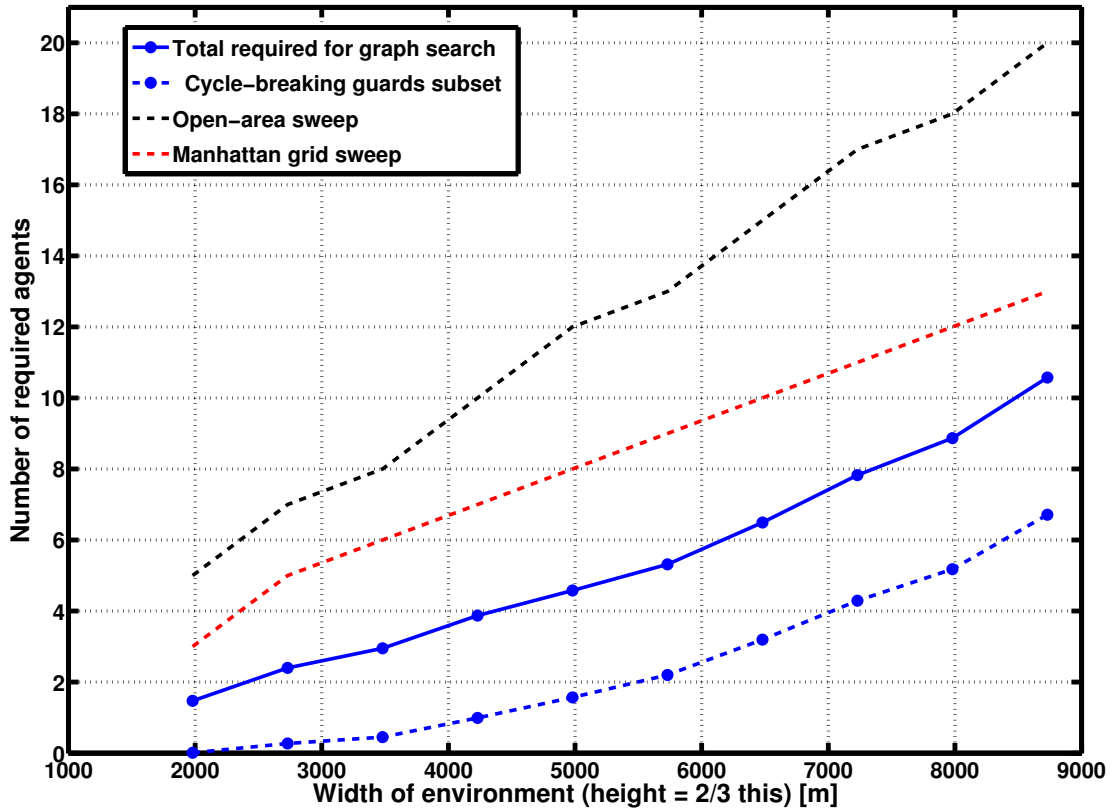


Figure 6.13: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network with fixed 450m edge length and edge density of 0.5 (50% of edges from the underlying grid are present), but varying size from 2km × 1.3km to 8.7km × 5.8km. For reference, note that environments with these parameters are considered canonically suburban in this document.

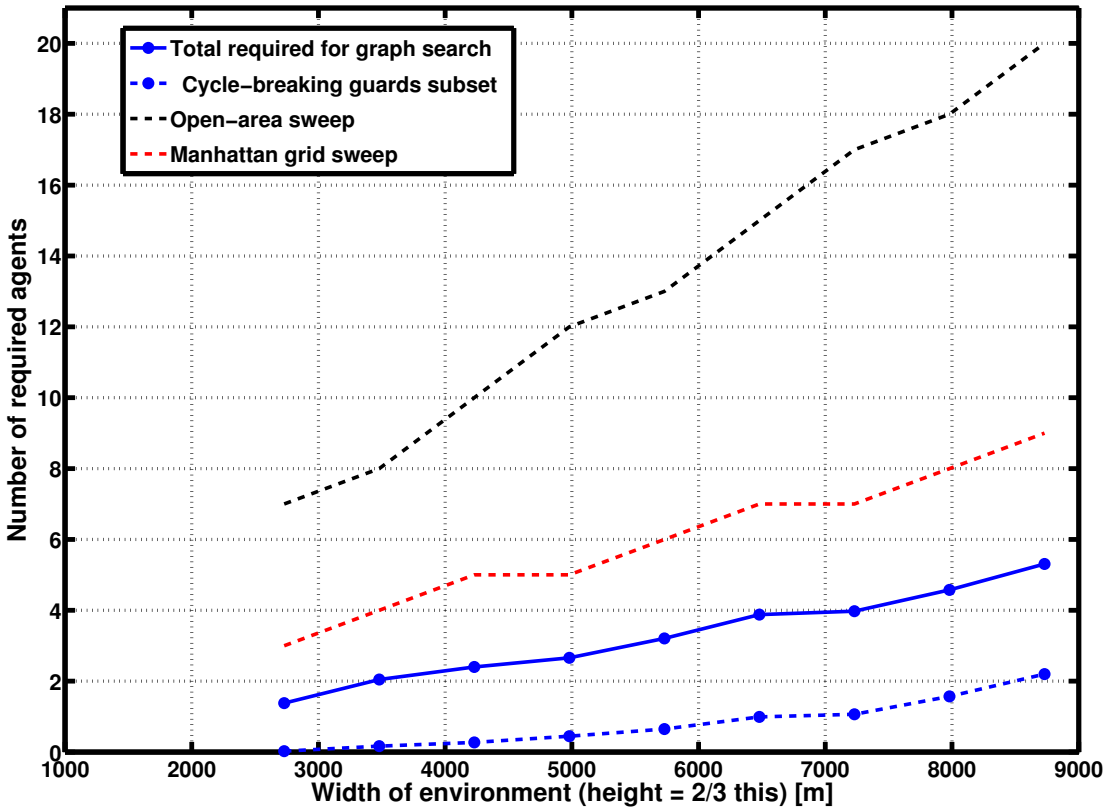


Figure 6.14: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network with fixed 700m edge length and edge density of 0.5 (50% of edges from the underlying grid are present), but varying size from 2.7km $\times$ 1.8km to 8.7km $\times$ 5.8km. For reference, note that environments with these parameters are considered canonically rural in this document.

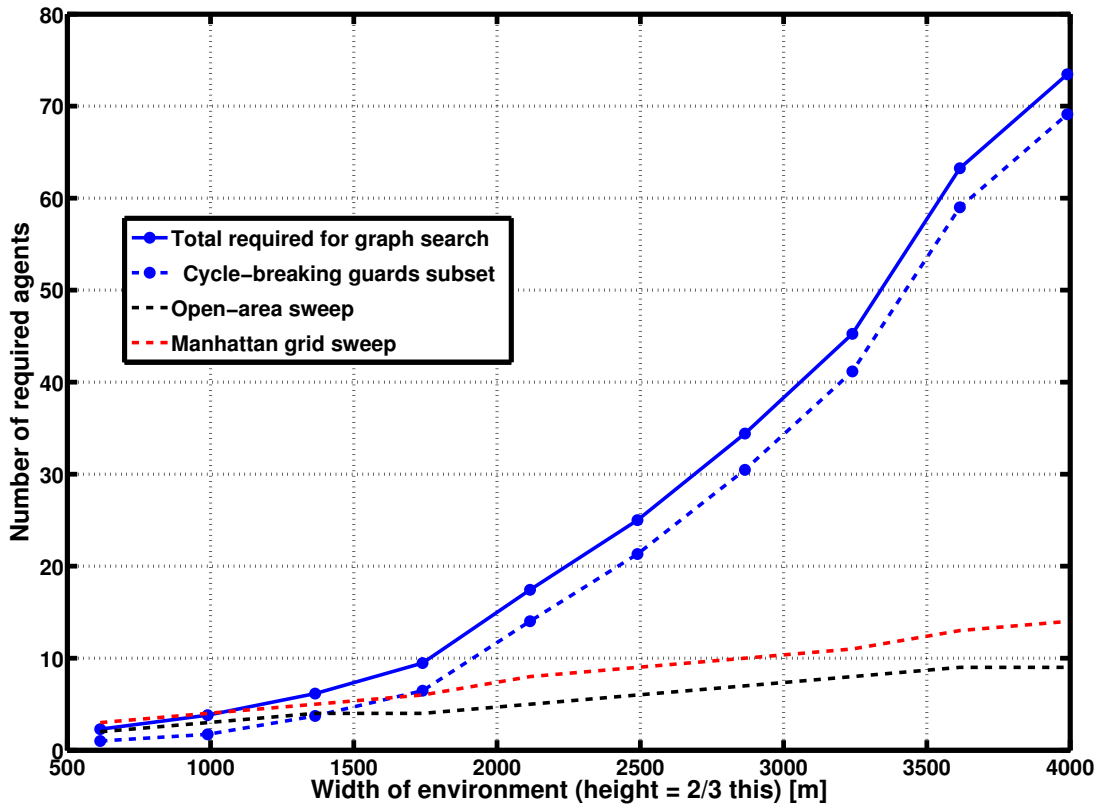


Figure 6.15: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network with fixed 200m edge length and edge density of 0.8 (80% of edges from the underlying grid are present), but varying size from 1.2km $\times$ 0.8km to 8.7km $\times$ 5.8km. For reference, note that environments with these parameters are considered canonically very dense urban in this document. Contrast this plot with Figure 6.12, having edge density 0.5. At this higher density, the proposed algorithm is outperformed by open-area sweeping for even very small environments. That open-area sweeping outperforms Manhattan sweeping as well is indicative that at this very high density, environments are best treated as densely-connected open areas rather than graph structures.

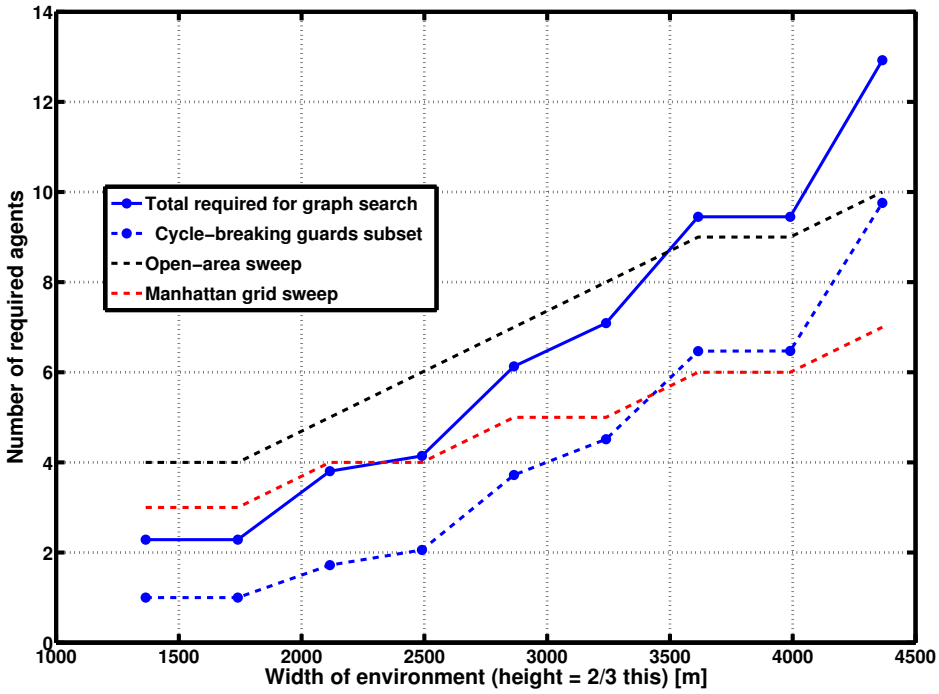


Figure 6.16: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network with fixed 450m edge length and edge density of 0.8 (80% of edges from the underlying grid are present), but varying size from 2km $\times$ 1.3km to 8.7km $\times$ 5.8km. For reference, note that environments with these parameters are considered canonically very dense suburban in this document. Contrast this plot with Figure 6.13, having edge density 0.5. At this higher density, the proposed algorithm outperforms others for only small environments.

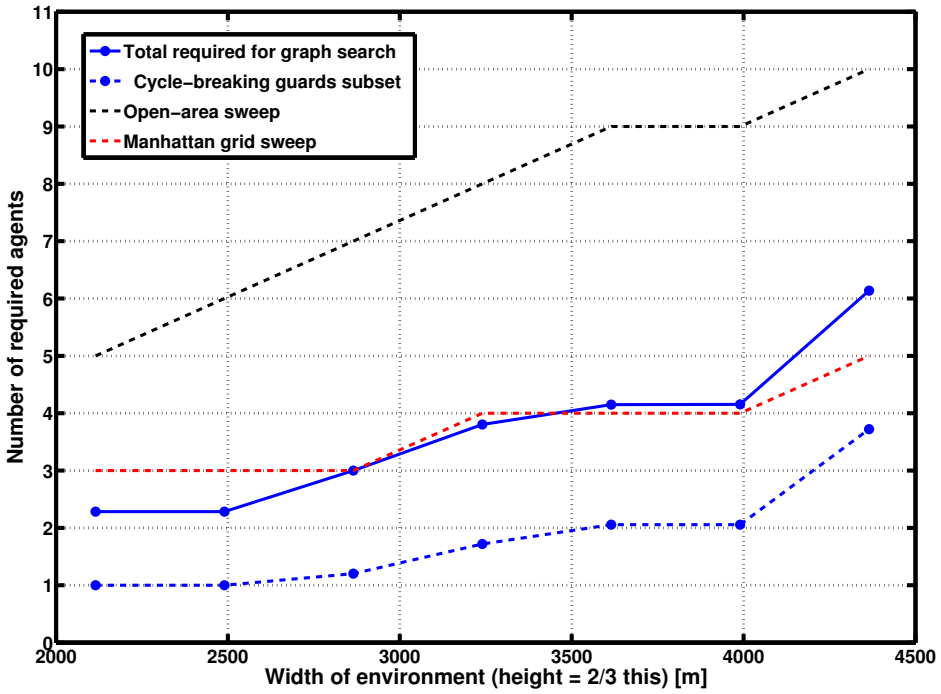


Figure 6.17: Comparison of required team size to perform a guaranteed search of a possibly cycle-containing road network with fixed 700m edge length and edge density of 0.8 (80% of edges from the underlying grid are present), but varying size from 2.7km $\times$ 1.8km to 8.7km $\times$ 5.8km. For reference, note that environments with these parameters are considered canonically very dense rural in this document. Contrast this plot with Figure 6.14, having edge density 0.5. At this higher density, the proposed algorithm outperforms others for only relatively small environments.

6.3 Bounded-speed Targets

In practice, infinite-speed search may be highly conservative and resource-intensive, requiring many agents (especially in highly cyclic environments) and long search times. Additionally, searching a large area contaminated by an evader presumed to diffuse instantaneously is unnecessary if additional information is available, such as a rough location and some speed bound, e.g. approximately 10mph (4.5m/s) for a human evader and 30-80mph (13.5m/s to 35.8m/s) for an automobile, in which case the search is reduced to a smaller, more slowly-diffusing one.

Relevant scenarios include, among others, the escape of a target from tracked view during pursuit, a sighting reported at a remote location, or the need to search a small portion of an unbounded area with connectivity to effectively arbitrarily distant locations (e.g. with access to an interstate highway) to which an infinite-speed target model simply cannot be applied. Each of these may be addressed by initial contamination of some small map subset that diffuses at a worst-case target speed.

Intuitively, infinite-speed search as proposed in the preceding section may be built upon somewhat readily to produce effective search strategies for bounded-speed targets. If, for instance, an infinite-speed search of an area can be performed before the initial diffusion can exceed the extent of that area, then capture is guaranteed. Alternatively, if diffusion can be stemmed by guarding appropriate boundary points, then a search may be completed at leisure of the now-bounded area to also assure capture. These two examples inspire the following two respective strategies. The first is best suited to smaller teams of fast agents, while the second is tolerant of slower agents as long as more are available.

6.3.1 Strategy A: live search

Conceptually, an initial partial map contamination corresponds to some subset of the underlying graph that is of interest that grows with time as contamination spreads with possible target motion. Provide a bound on the rate at which this spread occurs, the growth of this graph may be directly predicted as a time-varying graph $\mathbf{G}(t)$. If at any time this predicted graph subset can be searched by any guaranteed means (such as using the preceding infinite-speed search) within this time t, then subject to the validity of the speed bound the graph growth prediction depended on, guaranteed capture is assured.

This notion may be phrased as the formal statement of Algorithm 6.4. Informally, an alternation occurs between predicting map growth and estimating its search time.

Starting at $t_0 = 0$, an estimate is made of the time t_1 necessary to search the graph $\mathbf{G}_0 = \mathbf{G}(t_0 = 0)$. If $t_1 \leq t_0 = 0$ (unlikely), then success is declared. Otherwise, the growth in the map is predicted up to time t_1 as $G_1 = \mathbf{G}(t_1)$, the search time t_2 of $\mathbf{G}_1$ is estimated, success declared if $t_2 \leq t_1$, and iteration continued if not. If at any time the search number of $\mathbf{G}_i$ exceeds the number of available agents, failure is declared, as this will only increase with future iteration (since $\mathbf{G}_i \subseteq \mathbf{G}_j$ for $i < j$). Note that especially for large t_1 (which may typically correspond to lengthy initial maneuvers to the area of interest from some distant launch location), large jumps in the growth of $\mathbf{G}$ may result, which in practice result in few iterations before termination.

This alternation relies on the assumption that sweep_time($\mathbf{G}$) $\leq$ sweep_time($\mathbf{G}'$) for $\mathbf{G} \subseteq \mathbf{G}'$. Though intuitive, this may be further verified with the following reasoning. Suppose that a graph $\mathbf{G}$ is extended by the addition of an edge e to a new leaf node v to form $\mathbf{G}'$ (additional disconnected nodes are not relevant to edge search as performed here, and v must exist since search occurs only after cycles are broken). The new edge e must fall into one of two cases. It is either the first edge to be searched (so that v is the root of the search), or e is a new leaf subtree of some existing node v'. In the second case, the addition of e trivially represents additional work and hence search time. In the first, it may be the case that alignment to begin a sweep of e is faster under vehicle motion constraints than alignment with the next edge below it, resulting in a partial decrease in search time, but added to this is the time taken to sweep e. As long as the length of e exceeds the arc length of the eliminated maneuver, a net increase in search time still results. Under a Dubins model, the additional length of a maneuver beyond the Euclidean distance between the initial and final locations is never more than approximately three turning radii, and so as long as all edges (or at least per-iteration outward growth in the map) are at least this long, the necessary inequality holds. Nevertheless, to provide some measure of conservative assurance that no intermediate subset graph is missed due to this alternation, experimentation was performed with an alternate definition $t'_i = t_i/2$. A more extreme alternative is to reduce this alternation to a linear outward search of the graph, iteratively testing the search time of small incremental growth in the contaminated subgraph.

Arguably more important is the validity of the estimate of search time. Ideally, simulation may be performed of the candidate search to determine this value. However, this is both implementationally complex as well as computationally intensive if no analytic form of vehicle models exist (e.g. only a forward numerical model is available). An alternative is the use of a conservative approximation for the search time, e.g. a hypothetical execution of a more basic search strategy with worst-case maneuver times assumed. At the cost of over-estimating search time and thereby

Input: *tree* to clear, initial locations of searchers, and contiguous initially contaminated set of edges
Output: search schedule, predicted mission time, and search number (required team size) or FAILURE

```
search_schedule = [ ]
Function [search_number,duration] = bounded_recaptureA(tree,
max_target_speed, contaminated_edges[ ], agent_poses[ ])
begin
    new_search_time = 0
    repeat
        search_time = new_search_time
        // Straightforward and not defined here:
        tree_subset = diffuse_contamination(tree, contaminated_edges,
        max_target_speed, search_time)
        // Defined in Algorithm 6.1 and may be wrapped
        // in additional loop to choose best root node:
        search_number = clear_tree(tree_subset, NULL)
        if search_number > agent_poses.size then
            return FAILURE
        // Conceptually, a simulation of the returned search
        // schedule returning an over-estimate for admissibility,
        // but is implementation-dependent and not defined here:
        new_search_time = estimate_search_duration(tree_subset, agent_poses)
    until new_search_time ≤ search_time ;
    return [search_number, new_search_time]
end
```

Algorithm 6.4: "Live search" guaranteed tree edge search for bounded-speed targets

allowing the graph to grow larger than necessary or risking returning failure spuriously, this adds an element of robustness in that even in the face of disturbances or longer-than-expected maneuver time owing to slight pose changes, capture within the reported time is still guaranteed.

Note that this algorithm may be readily extended to disjoint subsets of the overall graph (e.g. multiple targets or initial hypotheses as to a target's location) by individually diffusing the initial contamination for each, merging as necessary (should contamination from one reach others), generating a search schedule for each, and estimating the search time resulting from an agent-to-subgraph assignment optimization.

6.3.2 Strategy B: bound and search

An alternate conceptual approach is to attempt to block the growth in map contamination, at which point a fixed bounded graph is contained within and may be searched by any guaranteed search strategy, regardless of search duration—such as, again, the infinite-speed search from the preceding section. In practice, this corresponds to guarding of strategic boundary nodes (or synthetic nodes inserted along partially-contaminated edges). In similar logic to strategy A just described, given a predicted graph subset $\mathbf{G}(t)$ corresponding to the contaminated set at time t and provided all bounding nodes can be reached within time t, then capture is assured if sufficient agents are available to simultaneously guard these nodes and perform the interior search.

Phrased formally as Algorithm 6.5, this progresses similarly to an incremental version of strategy A. The contaminated subgraph $\mathbf{G}_0 = \mathbf{G}(t_0 = 0)$ at time t_0 is computed, along with all boundary nodes linking nodes in $\mathbf{G}_0$ to nodes in the original surrounding graph. If sufficient agents are available and these can all be reached and guarding begun by time t_0 (unlikely), then the search number of the internal subgraph is also computed. If the total number of boundary guards plus internal searchers required is at most the total number of available agents, then success is declared. Otherwise, the algorithm iterates to an incremental t_1 corresponding to a minimal discrete growth in the contaminated graph (e.g. a single edge or quantized piece thereof), and the same test of guard time and internal search number is performed. Eventually, if the original environment is unbounded, the number of agents required to merely perform the internal search will exceed the number available, and by the reasoning in the preceding subsection, this is a failure condition, as a larger graph will only require more. If the original environment is instead bounded, the entire graph will eventually be reached, requiring no boundary guards (and hence no guard reach

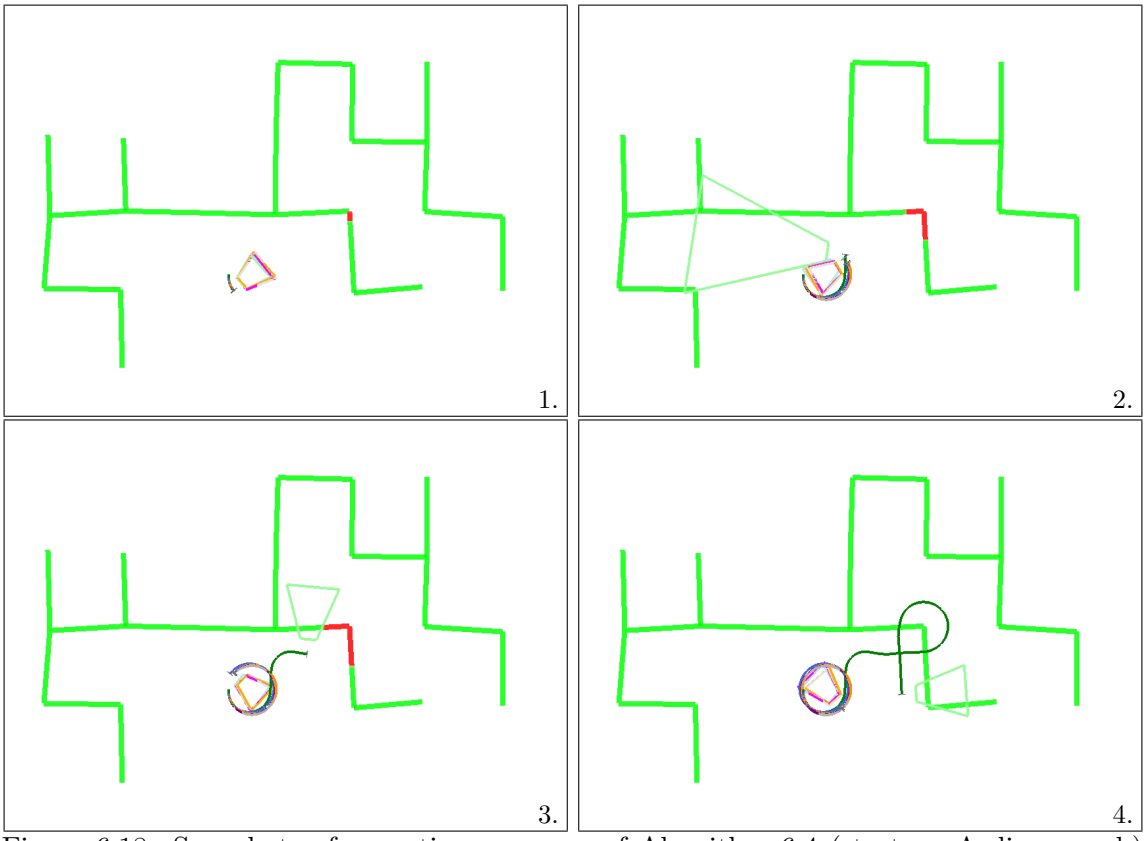


Figure 6.18: Snapshots of execution sequence of Algorithm 6.4 (strategy A, live search) applied to a small demonstration environment, in which a target sighting is initially reported, (1) contaminating a small area. One UAV within what happens to be a larger team is dispatched, (2) by which time contamination has spread around a corner. The UAV begins a sweep of the edge entering this corner, (3) stopping further contamination growth along this direction. The UAV continues its sweep inward, (4) around the corner and onto the next edge, knowing that contamination will not have spread beyond it by the time it is fully cleared.

time) and merely require as many agents to search as the original search number.

Note that boundary guards may be exploited by the interior search to improve its search number and duration. In full exploitation, guards may be maintained at a location only as long as necessary and actively participate in the search. This interaction may be arbitrarily complicated and can be seen as an instance of joint combinatorial optimization avoided here. Starting from minimal interaction, boundary guards may at least be treated as permanently guarding a given node should that prove useful during the search (e.g. if this node contains subtrees in the search graph). A next step motivates a small variation of this strategy that may be considered: boundary guards lying on leaf nodes in the contaminated subgraph sweep up the path on which they lie until a node is reached having more than one child. This step, which is time-invariant once each outer boundary node is reached, then places the guards at nodes that will have required a guard at some point during the internal search, providing a benefit. As the practical impact of this variation is not immediately clear beyond an occasional reduction in search number, it is included as a point of comparison in experimentation.

6.3.3 Experimental results

To evaluate and compare these two strategies, similar repeated realistic simulations were performed. Simulation parameters and map generation were identical to that described in Section 6.2.3. Four strategies variations were compared:

1. Live search (A) using a rapidly-computed conservative estimate of search time during the alternation phase.

2. Live search (A) that invokes offline (non-realtime) simulation of search applied to a candidate area to provide a lower-bound on the performance of admissible estimates

3. Bound and search (B) performed by sending guards to boundary locations and then performing a search in the contained area

4. The suggested variation of bound and search (B) in which boundary guards additionally sweep up the tree if lying on a path until reaching a node with degree greater than two, after which the remaining contained area is searched

In the first series of experiments, the effect of relative speed between searchers and evaders is evaluated. Several hundred simulations with random initial team locations and seeds were performed for a sampling of searcher to evader speed ratios

Input: *tree* to clear, initial locations of searchers, and contiguous initially contaminated set of edges
Output: boundary guard assignment, search schedule, and search number (required team size) or FAILURE

search_schedule = [ ]
Function [search_number,guard_assignment] = bounded_recaptureB(*tree*, max_target_speed, contaminated_edges[ ], agent_poses[ ])
begin
    growth_time = 0
    tree_subset = [ ]
    repeat
        `// Straightforward and not defined here:`
        [*tree_subset*,distance_extended] = diffuse_contamination_one_edge(*tree*, *tree_subset*)
        growth_time = growth_time + distance_extended / max_target_speed
        `// Returns nodes at the boundary between the larger graph`
        `// and a provided subgraph but is straightforward and not`
        `// defined here:`
        boundary_nodes = get_boundary_nodes(*tree*, *tree_subset*)
        if *boundary_nodes.size* $\geq$ *agent_poses.size* **then**
            continue
        `// Computes an (optimal) assignment of agents to nodes`
        `// and returns conservative estimate of time to reach`
        `// them.  Also straightforward and not defined here:`
        [guard_time,guard_assignment] = get_node_reach_time(boundary_nodes, agent_poses)
        if *guard_time* $\geq$ *growth_time* **then**
            continue
        `// Defined in Algorithm 6.1 and may be wrapped`
        `// in additional loop to choose best root node:`
        search_number = clear_tree(*tree_subset*, NULL)
        if *search_number* $>$ *agent_poses.size* **then**
            return FAILURE
    until *search_number* + *boundary_nodes.size* $\leq$ *agent_poses.size* ;
    return *search_number*
end

Algorithm 6.5: "Bound-and-search" guaranteed tree edge search for bounded-speed targets

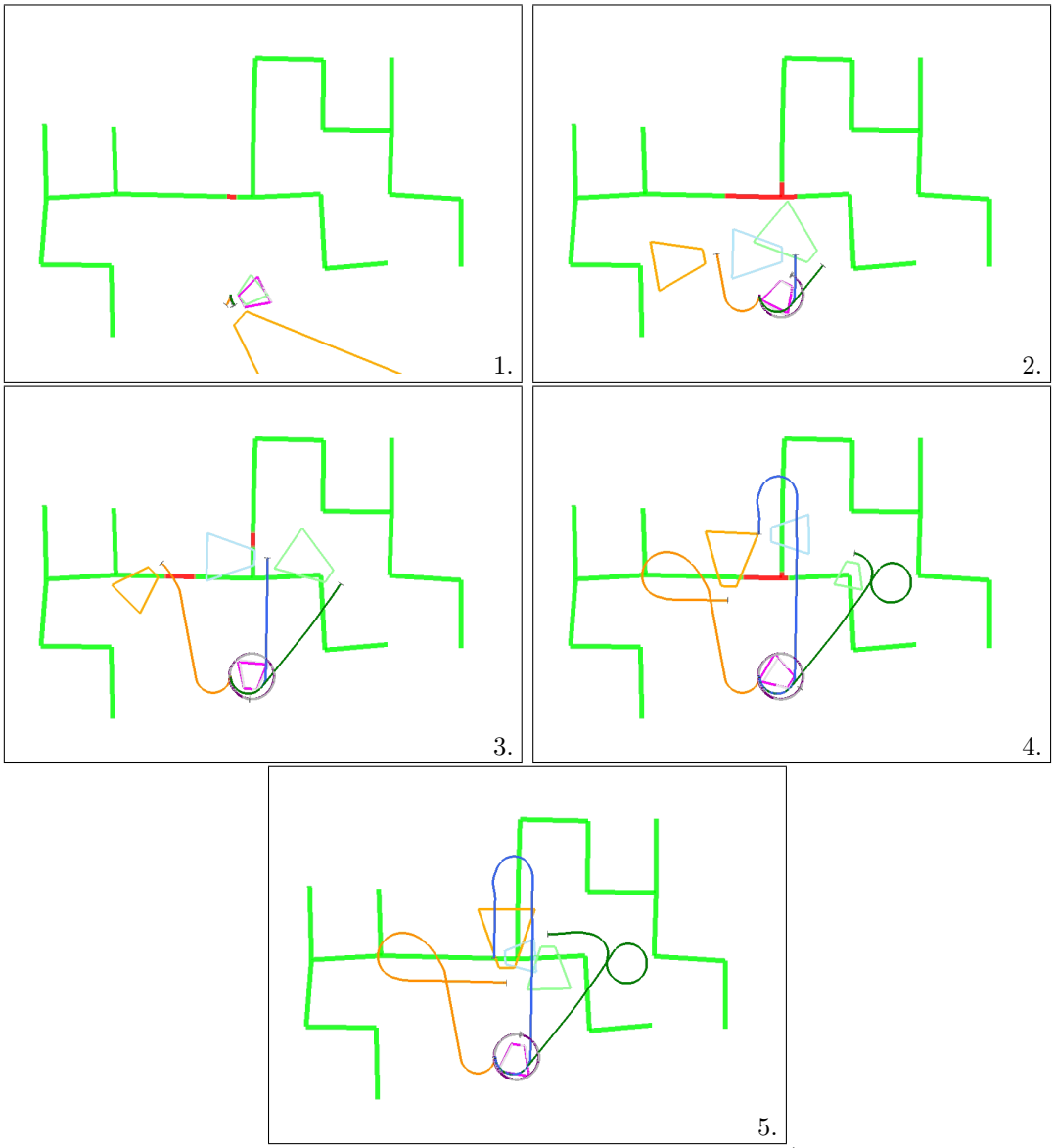


Figure 6.19: Snapshots of execution sequence of Algorithm 6.5 (strategy B, bound and search) applied to a small demonstration environment, in which a target sighting is initially reported, (1) contaminating a small area. Three UAVs within what happens to be a larger team are dispatched, (2) by which time contamination has spread through a junction to two other edges. The three UAVs move to external bounding points, (3) incidentally eliminating contamination from one edge. The UAVs then execute the "sweep-up" variant of the strategy described in the following subsection, (4) sweeping inward from these bounding nodes to the parent node of each. Having completed these inward sweeps, (5) the UAVs find themselves at the same node, leaving an empty graph to clear, thereby completing the search.

(all searchers on the team having the same speed) between 0.5 and 10 for 3km $\times$ 2km areas with fixed environment density and three "city block" edge lengths: 200m, 450m, and 700m (roughly capturing urban, suburban, and rural densities). For each run, the search number, time to mission completion, and the fraction of the entire map searched were recorded. As before, precomputation time for all strategies except the offline simulation were less than several seconds, which may be considered realtime for practical implementation. Likewise, once again, all environments tested were deliberately chosen to form trees, so that comparison of performance after emplacing any necessary cycle-breaking guards might be evaluated.

The effect of pursuer-to-target speed ratio on the search number is shown in Figures 6.20 through 6.22. In addition to the total search number for all four compared strategies, also shown are the search number for the entire map and the number of agents assigned to the internal search (rather than as guards) for the latter two strategies. All exhibit roughly the identical trend that for lower speed ratios (here typically 8, though this may vary for differently-sized environments), live search requires more agents on average, while bound and search requires more for higher speed ratios. This is consistent with the intuition that live search is better suited to smaller teams of faster agents. Except in dense environments, the simple version of bound and search consistently requires the most agents, since agents must be used as guards in addition to searching an area that, for less dense environments, is comparable in complexity (i.e. its own search number) to that of a larger area. The latter "sweep-up" variation of bound and search strictly outperforms this as it directly reduces search complexity by eliminating leaf subtrees whose parent node may have required a guard otherwise. Its performance levels out for higher speed ratios.

Next, the effect of speed ratio on map clearance time is shown in Figures 6.23 through 6.25. Note that in these plots, the search is being performed by the smallest team possible for each individual scenario (see Figures 6.32 through 6.34 for consistently-sized teams), so larger teams may perform faster. These too exhibit the same trend overall in that a crossover point is present between the performance of live search and bound-and-search. It is interesting to note that for dense environments, the relative performance of simple bound-and-search and its sweep-up variation are inverted, likely owing to the impact of leaf subtree parallelization during actual execution given larger team availability.

Finally, the fraction of the map searched is shown in Figures 6.26 through 6.28. For the most part, needing to search the entire map may be interpreted as cases in which the target would have escaped in an unbounded environment (this is not strictly the case since a sufficiently large team may perform almost arbitrarily fast

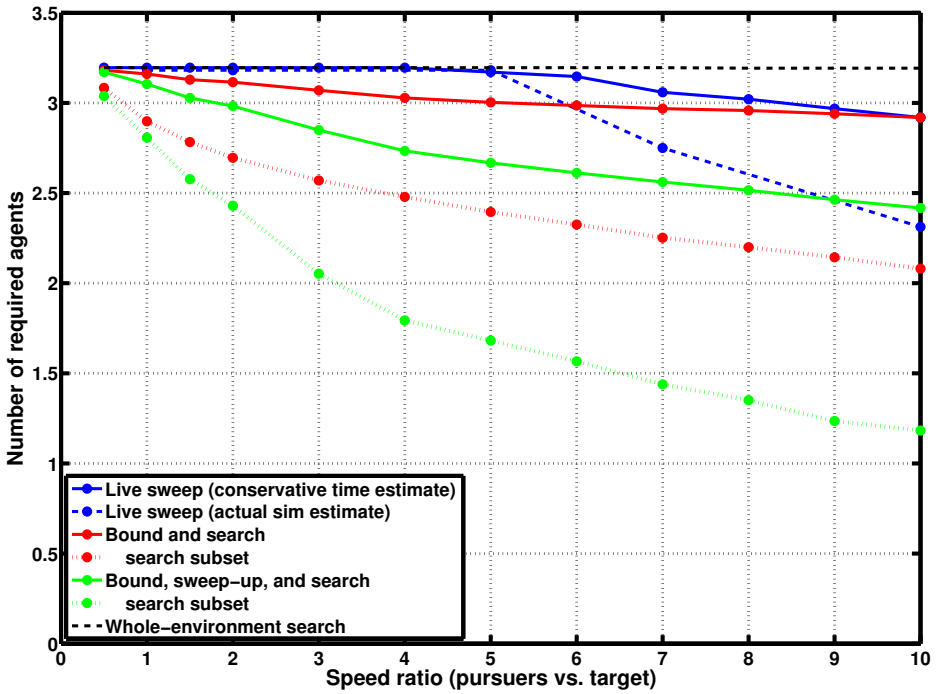


Figure 6.20: Required team size vs. searcher-to-target speed ratio used to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 200m (roughly, urban). The required team size is determined by iterating from a single agent until a guaranteed capture solution is found, of any mission duration. Because the environment is bounded, the largest required team size is the search number of the entire map, to which all of the strategies may fall back.

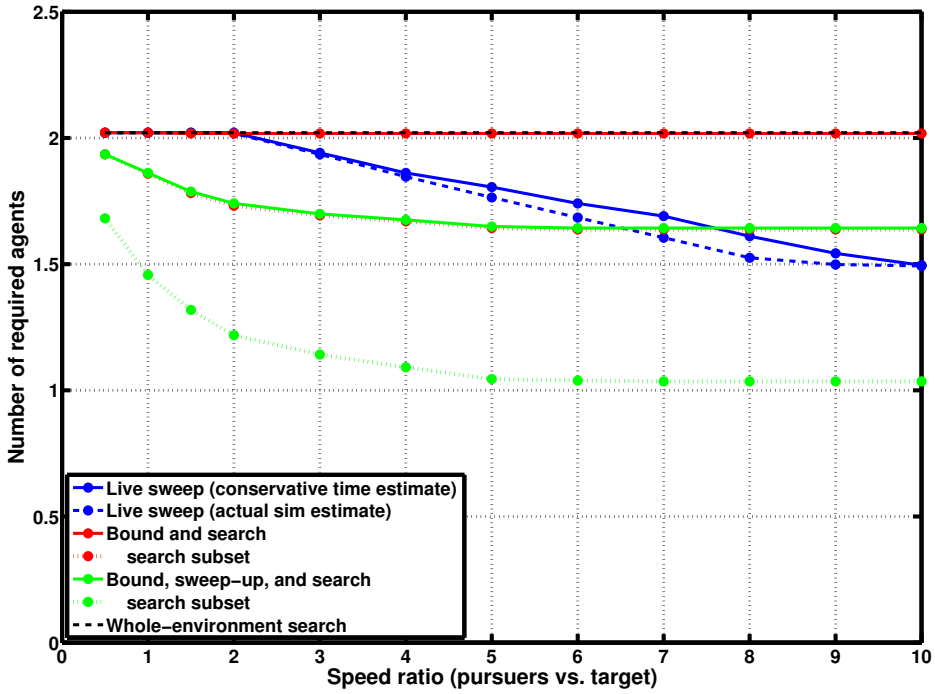


Figure 6.21: Required team size vs. searcher-to-target speed ratio used to perform a guaranteed search for a bounded-speed target in a 3km $\times$ 2km road network with 40% edge density and block (edge) lengths of 450m (roughly, suburban). The required team size is determined by iterating from a single agent until a guaranteed capture solution is found, of any mission duration. Because the environment is bounded, the largest required team size is the search number of the entire map, to which all of the strategies may fall back.

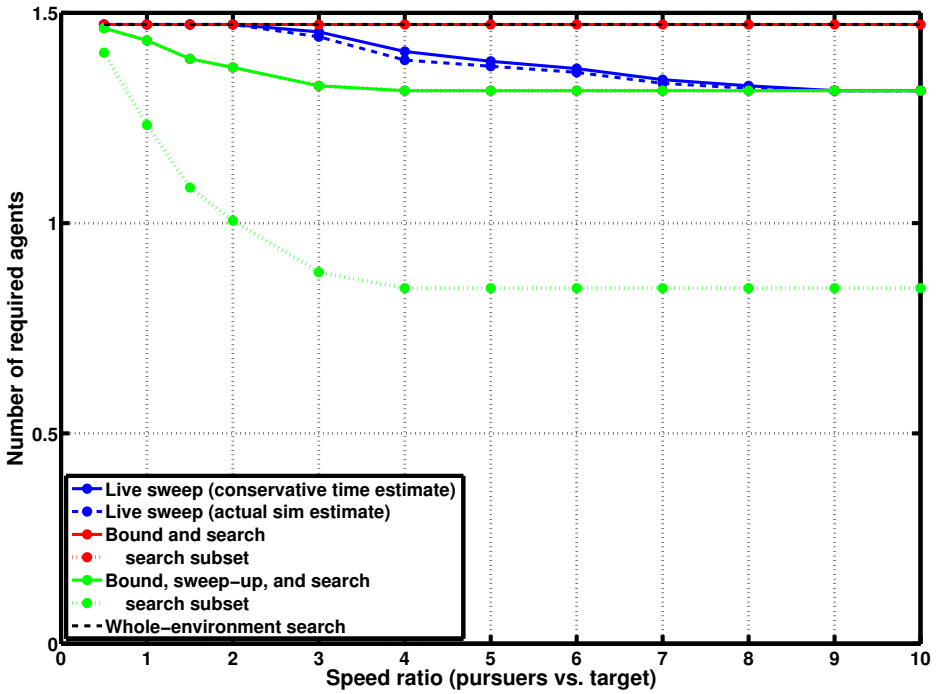


Figure 6.22: Required team size vs. searcher-to-target speed ratio used to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 700m (roughly, rural). The required team size is determined by iterating from a single agent until a guaranteed capture solution is found, of any mission duration. Because the environment is bounded, the largest required team size is the search number of the entire map, to which all of the strategies may fall back.

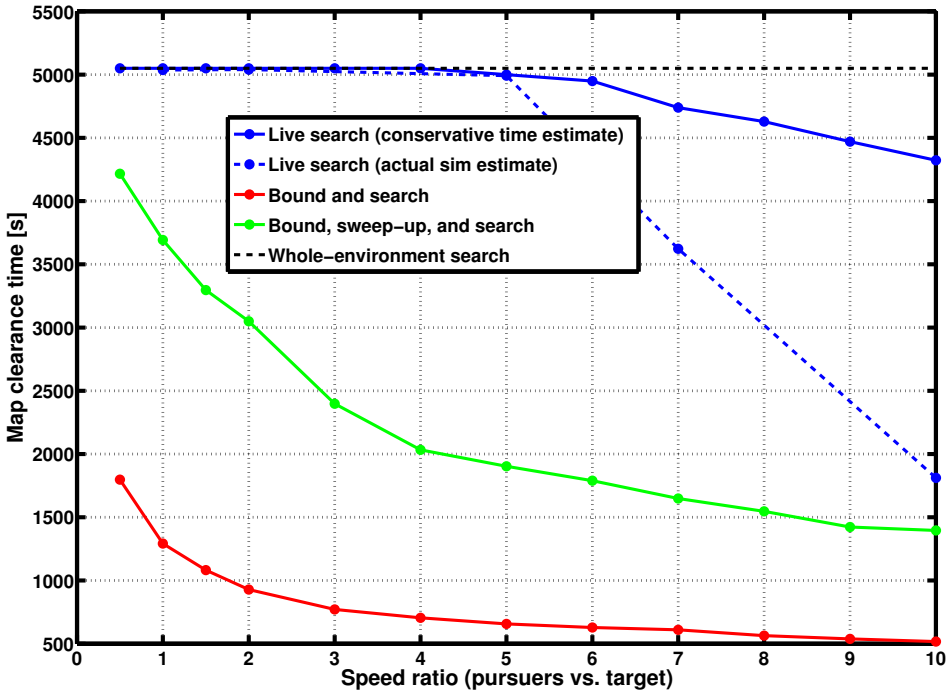


Figure 6.23: Comparison of mission duration with increasing searcher-to-target speed ratio to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 200m (roughly, urban). The number of agents available to each strategy corresponds to the matching datapoint in Figure 6.20, that is, the minimum team size providing a capture solution is used in each instance.

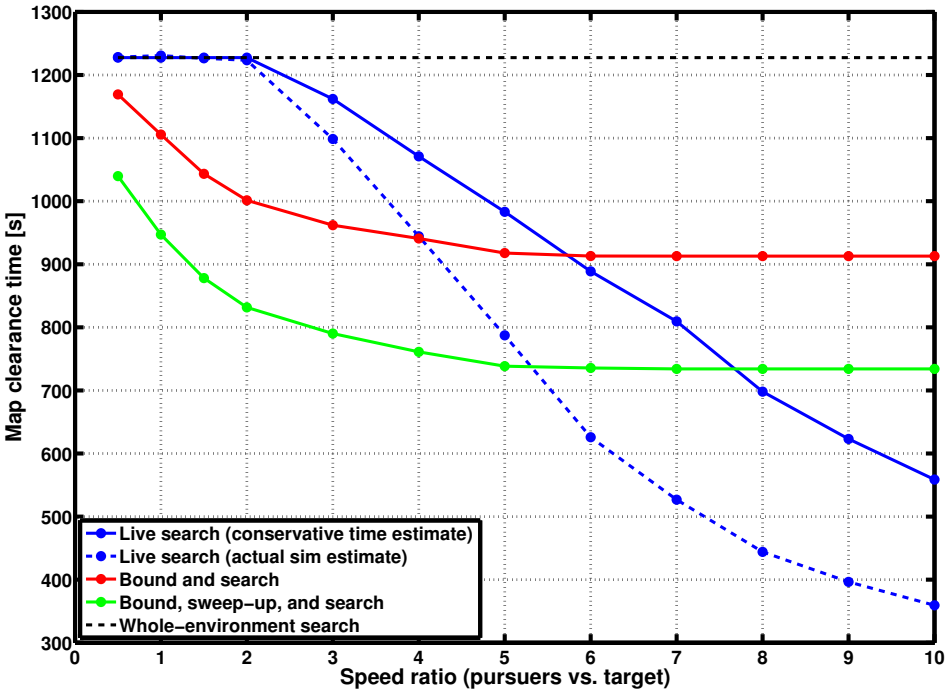


Figure 6.24: Comparison of mission duration with increasing searcher-to-target speed ratio to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 450m (roughly, suburban). The number of agents available to each strategy corresponds to the matching datapoint in Figure 6.21, that is, the minimum team size providing a capture solution is used in each instance.

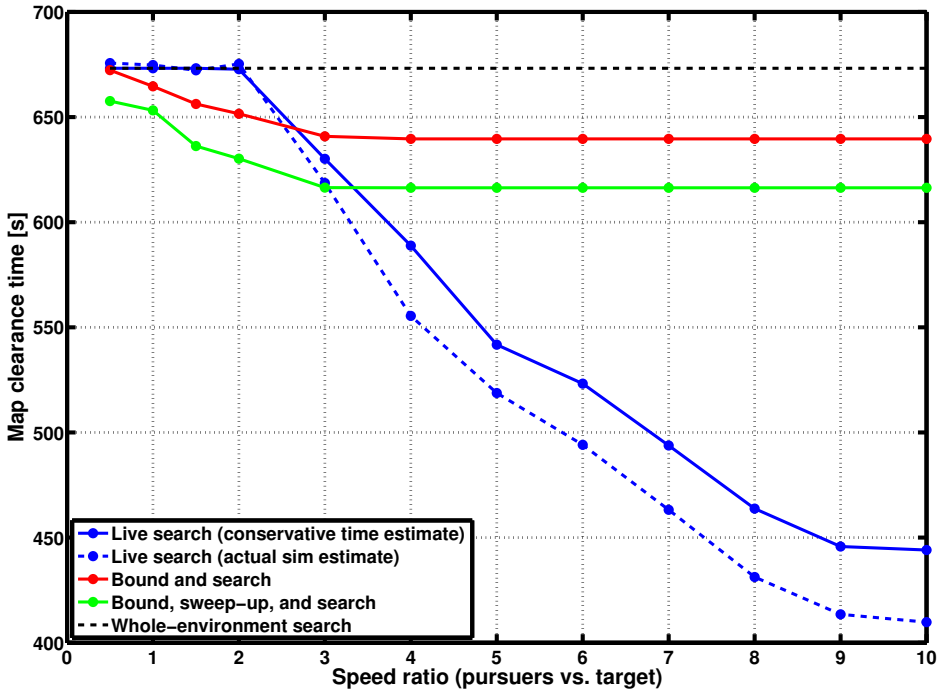


Figure 6.25: Comparison of mission duration with increasing searcher-to-target speed ratio to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 700m (roughly, rural). The number of agents available to each strategy corresponds to the matching datapoint in Figure 6.22, that is, the minimum team size providing a capture solution is used in each instance.

searches subject to initial positioning time if a completely parallelized search is used). As expected, the same crossover trend is exhibited, as is overall relative performance. This should come as no surprise, since map area covered is roughly proportional to the time required to do so, at constant vehicle speed. Note, however, the re-inversion of the two variations of bound-and-sweep given that only the area of the internally-searched region is shown, following the initial sweeps.

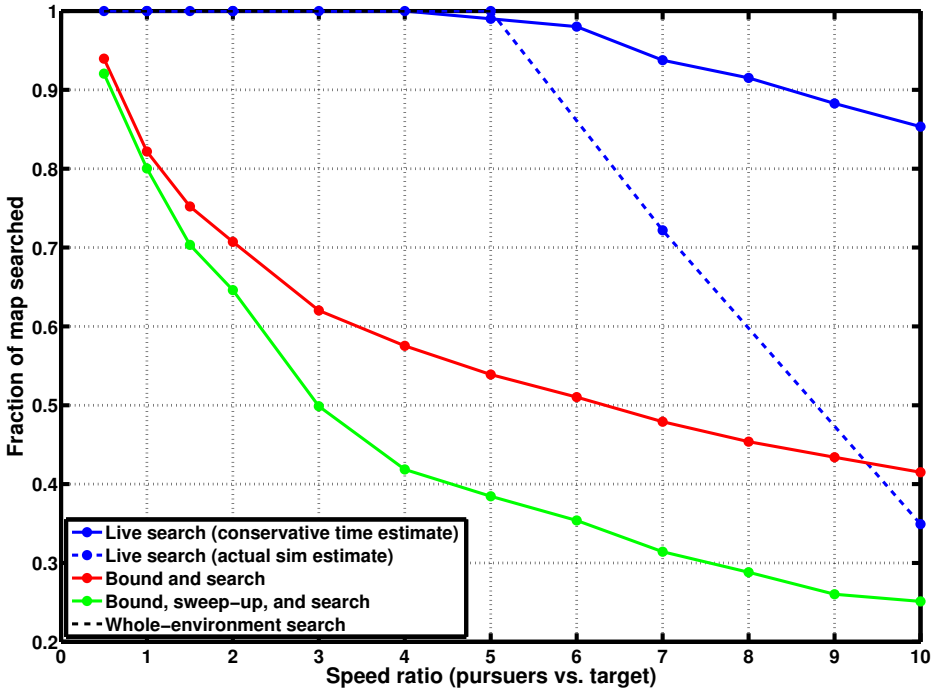


Figure 6.26: Fraction of the entire surrounding environment eventually searched vs. searcher-to-target speed ratio when performing a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 200m (roughly, urban). The number of agents available to each strategy corresponds to the matching datapoint in Figure 6.20, that is, the minimum team size providing a capture solution is used in each instance.

For comparison, the number of agents actually used (not including spare agents that might be used opportunistically in parallel search execution) and map clearance time for a hypothetical team with a fixed size of 10 agents are shown in Figures 6.29 through 6.31 and Figures 6.32 through 6.34, respectively. These depict the willingness of the strategies to choose search schedules (and guard assignments) using more than

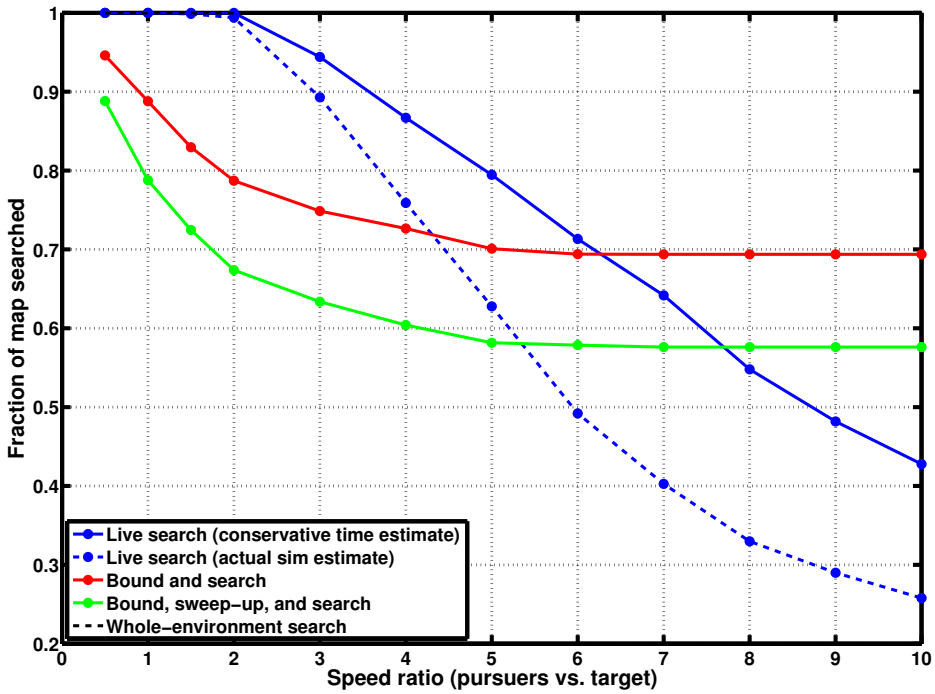


Figure 6.27: Fraction of the entire surrounding environment eventually searched vs. searcher-to-target speed ratio when performing a guaranteed search for a bounded-speed target in a 3km $\times$ 2km road network with 40% edge density and block (edge) lengths of 450m (roughly, suburban). The number of agents available to each strategy corresponds to the matching datapoint in Figure 6.21, that is, the minimum team size providing a capture solution is used in each instance.

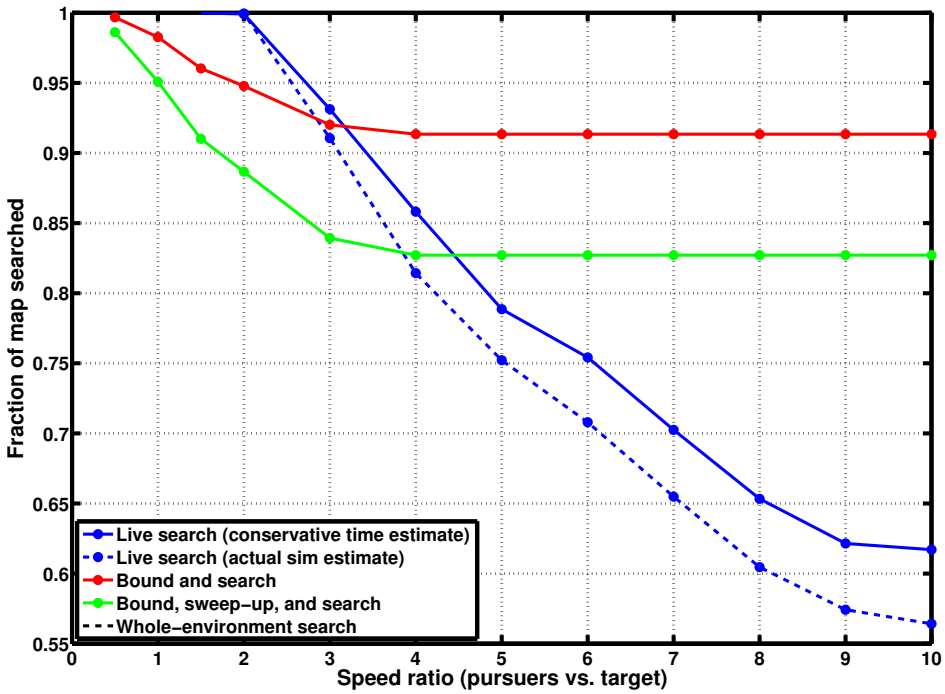


Figure 6.28: Fraction of the entire surrounding environment eventually searched vs. searcher-to-target speed ratio when performing a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 700m (roughly, rural). The number of agents available to each strategy corresponds to the matching datapoint in Figure 6.22, that is, the minimum team size providing a capture solution is used in each instance.

the minimum required number of agents if these still fit within the available team size—clearly so for bound-and-search that may choose guard-intensive boundaries, while live search only does so to the extent that the underlying guaranteed search can make use of additional agents via parallelism to reduce search time (of which the demonstrated algorithm does relatively little). This use of additional agents is particularly powerful at low speed ratios, at which bound-and-search strategies can consistently exhibit capture within durations far shorter than would be required to search the entire map as might otherwise be required with a relatively faster target.

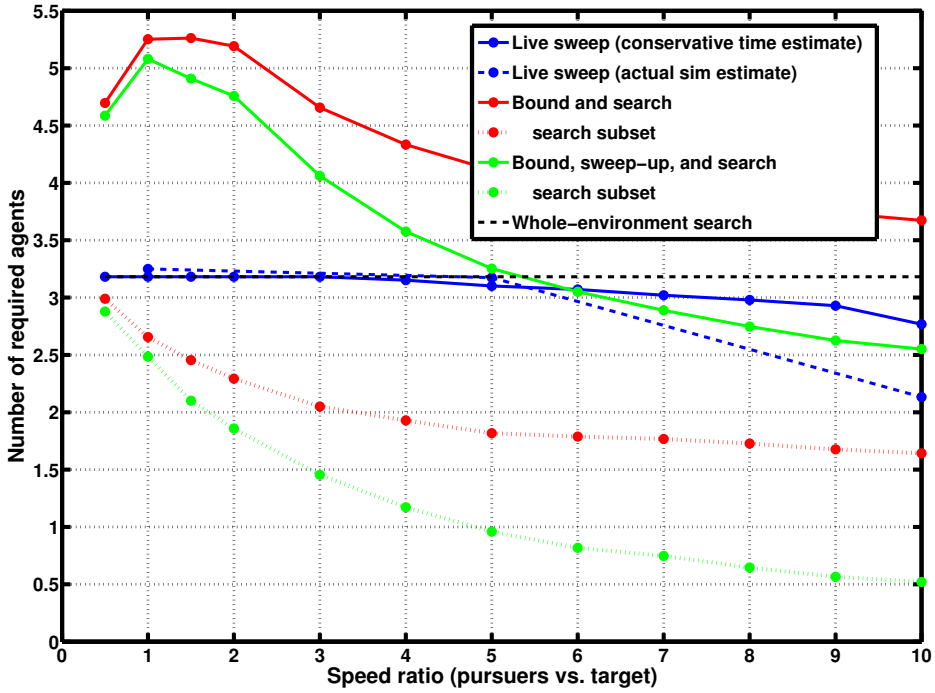


Figure 6.29: Required team size vs. searcher-to-target speed ratio used to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 200m (roughly, urban). The number of agents available to each strategy is fixed at 10, and more than strictly necessary may be used to reduce search time (area). Where obscured by legend, traces are roughly linear.

Overall, these experiments have validated the practicality of guaranteed recapture of adversarial bounded-speed targets in realistically sized environments with relatively small teams given speed ratios that may be feasible in practice—slow local-reconnaissance UAVs (at 20m/s) against humans or faster tactical ones (at 100m/s)

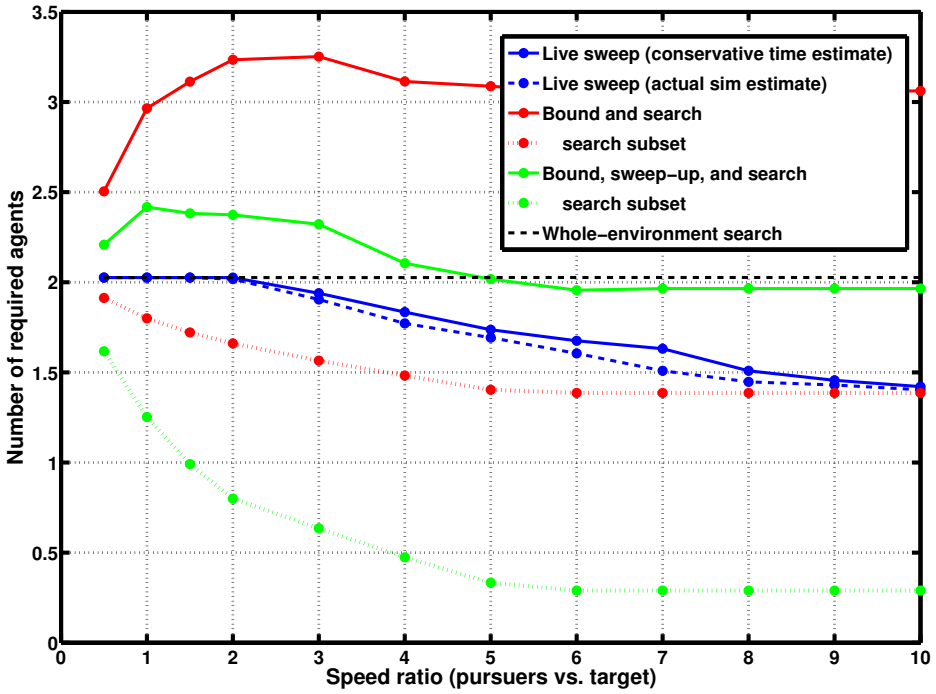


Figure 6.30: Required team size vs. searcher-to-target speed ratio used to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 450m (roughly, suburban). The number of agents available to each strategy is fixed at 10, and more than strictly necessary may be used to reduce search time (area). Where obscured by legend, traces are roughly linear.

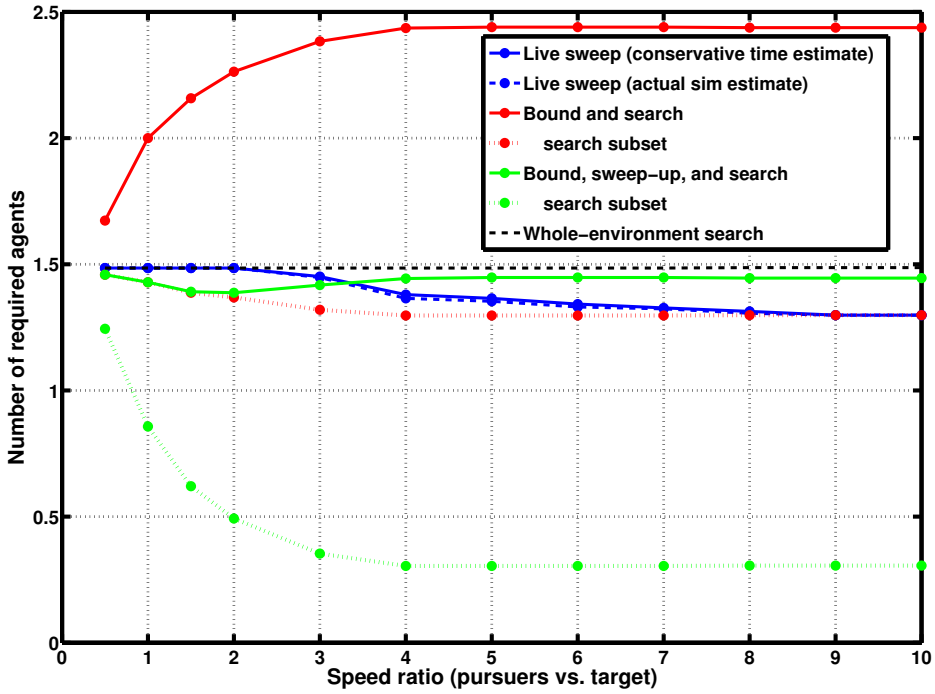


Figure 6.31: Required team size vs. searcher-to-target speed ratio used to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 700m (roughly, rural). The number of agents available to each strategy is fixed at 10, and more than strictly necessary may be used to reduce search time (area).

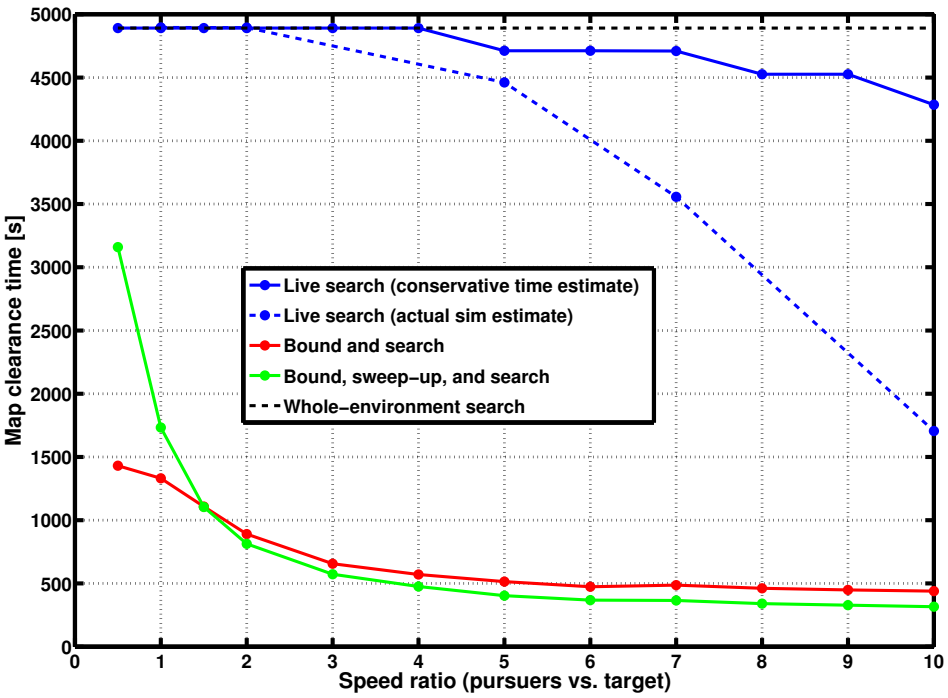


Figure 6.32: Comparison of mission duration with increasing searcher-to-target speed ratio to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 200m (roughly, urban). The number of agents available to each strategy is fixed at 10, and more than strictly necessary may be used to reduce search time (area). Note the impressive performance of bound-and-sweep methods given the chance to use many guard agents to surround the target.

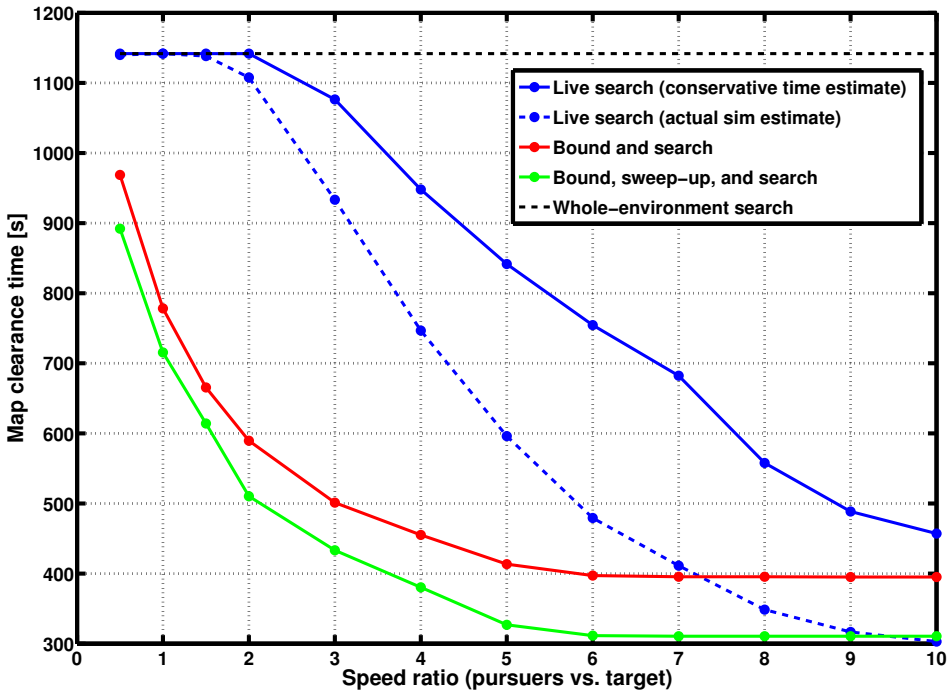


Figure 6.33: Comparison of mission duration with increasing searcher-to-target speed ratio to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 450m (roughly, suburban). The number of agents available to each strategy is fixed at 10, and more than strictly necessary may be used to reduce search time (area). Note again the impressive performance of bound-and-sweep methods given the chance to use many guard agents to surround the target.

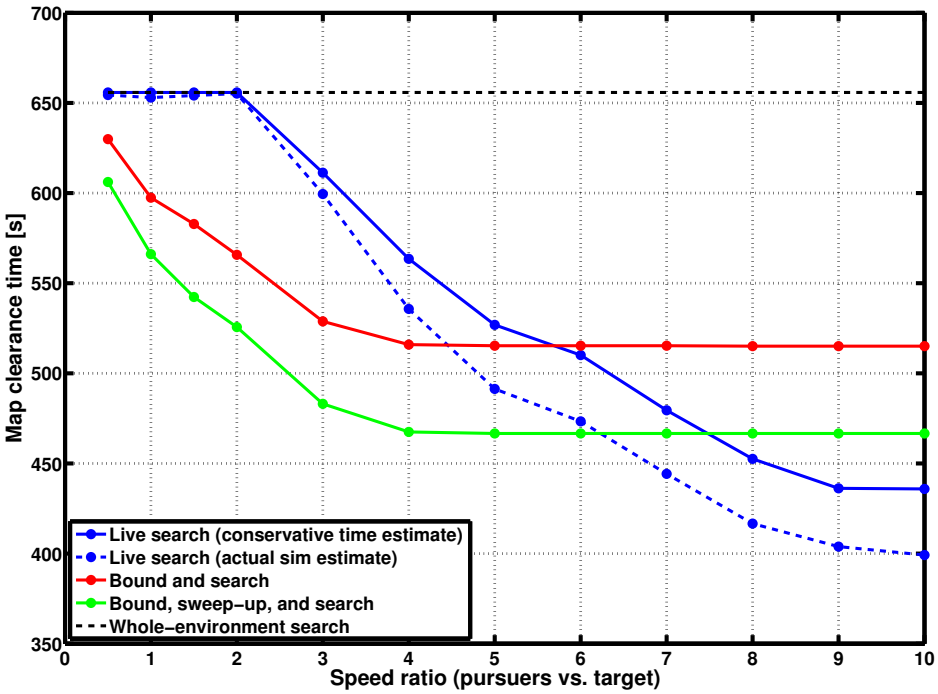


Figure 6.34: Comparison of mission duration with increasing searcher-to-target speed ratio to perform a guaranteed search for a bounded-speed target in a 3km × 2km road network with 40% edge density and block (edge) lengths of 700m (roughly, rural). The number of agents available to each strategy is fixed at 10, and more than strictly necessary may be used to reduce search time (area). Note here the effect of long traversal times dominating mission time, allowing live search to perform best at higher pursuer speeds.

for automobiles not seeking to stand-out by moving at very high speeds. The relative performance of the proposed strategies and their variations is moderately complex, leaving little opportunity to provide generalizations as to which should be applied under what scenarios. However, as the required team size and expected runtime for capture may be determined by rapid pre-mission computation, the best may easily be chosen on a per-situation basis. The use of a less-naive conservative estimate for the live sweep strategy (readily accomplished with additional implementation effort) might easily capture most of the remaining difference between its performance and that of the offline lower bound. Naturally, much better performance overall could be achieved by the use of an improved bounded-area guaranteed search strategy upon which these strategies are built. Additionally, highly-local recapture at higher speeds might be feasible by leveraging alternative pursuit-evasion abstractions, as suggested next.

6.4 Conclusions and Future Work

It can firmly be said that the work presented here barely scratches the surface of what may be possible in the realm of coordinated aerial search with capture guarantees. Both a number of direct extensions may be considered as well as conceptual alternatives, particularly through application of more complex pursuit-evasion abstractions that may provide better performance. Nevertheless, it is hoped that this work has demonstrated the power of even relatively simplistic abstractions in the hands of even highly motion-constrained UAVs to generate search (and in local or bounded-speed form, recapture) strategies providing performance guarantees.

Results and implementation experience suggest several avenues for direct extensions and additional possible interesting experiments. The proposed algorithm of leaf-subtree optimization demonstrated and evaluated here was chosen for implementation tractability, but it would be highly interesting to explore the relative improvement that may be gained by progressively deeper optimization. It is hereby conjectured that such improvement will be subject to very rapidly diminishing returns and that in practice such single-step optimization is sufficient. Another aspect not considered is the more active participation of either cycle-breaking or bounding guards in the search. Such integration may be done very incrementally, starting for instance with simply freeing the guard when no incident edge is any longer a source of contamination. Additional experimentation that would be worthwhile is further comparison against alternative approaches, such as those proposed for guaranteed search in open areas by either McGee and Hedrick [85] or Vincent and Rubin [134], to better evaluate when the resource requirements (as team size or search time) of

graph clearing exceed those of an open-area of the same area.

Generally, the capabilities of UAVs are still only partially leveraged as a trade-off against the availability of tractable, effective strategies. Though the aircraft are not artificially constrained to the graph beyond the act of sweeping edges, the presence of a continuous finite field of view is abstracted as a sweep or guard point, discarding the fact that multiple pieces of the environment may be visible at once. As an extreme example, two closely spaced parallel edges could be swept as one, but more commonly, portions of other edges are likely to be temporarily visible (and hence briefly guarded) during a sweep. Taking greater advantage of this, however, introduces tighter action coupling and worsened sensitivity to relative temporal trajectory disturbances that (except for sweeper prepositioning, which may be artificially made conservative) is presently almost entirely avoided.

In the same vein of a tradeoff in exchange for tractability, the strategies described are inevitably highly conservative, and an alternative abstraction, particularly one natively representing targets of bounded speed. Two suggestions are here made for such directions. The first would be to build upon a variation of the *IGNS* algorithm proposed by Kehagias et al. [70] that explicitly represents finite evader speed and multiple visibility, in addition to taking advantage of *non-monotonicity* or the allowance of recontamination in exchange for an overall better state globally. To do so, a discrete state set for the evader (e.g. a discretization of graph edges, coarser than but similar to that done in Section 5.2) and pursuers (e.g. by laying a lattice of "useful" viewing locations surrounding the graph) would be needed. Alternatively, a pursuit-evasion formalism known as cops-and-edge-robbers that is under active and recent development as a studied problem [35] may be applicable. A variation of the classical cops-and-robbers problem described in Section 3.1, this models robbers as lying on continuous edges rather than in discrete states (typically, vertices). A key question is how to fit the state and motion of UAV pursuers into this abstraction—most likely, by discretization of the graph into intermediate synthetic nodes lying along edges whose neighbors have feasible trajectories connecting them. An additional question with both this and the IGNS abstraction is the scalability and resulting tractability to environments with a width of multiple kilometers as considered here. IGNS relies upon a search in hypothetical configuration space, while cops-and-robbers problems are typically solved by dynamic programming.

Finally, several mission-level improvements are desirable. The first is that while the strategies proposed have a moderate amount of robustness against trajectory controller disturbances by placing guard and sweep points at the center of a large field of view and invariance to arrival time of agents at guard and sweep locations, larger disturbances such as failing to track the trajectory of a sweep and thereby losing view

resulting in recontamination are not handled. For the most part, such disturbances could be detected during execution, but optimal reaction to their occurrence has not been considered. Another desirable "online" aspect would be the handling of new information during the execution of a search. Certainly, if a new target location is reported, diffusion can be restarted from that location, but the current position of the team might be leveraged to reduce search time or required team size beyond simply recomputing a search.

As with efficient (probabilistic) search, handling of occlusion by terrain obstacles is also desirable. If such occlusion can be modeled a-priori via a directional visibility model, incorporating this may be straightforward. Consider again an assumed side-pointing sensor. Sweeps for a given edge can be constrained to only a single direction if occlusion is present when viewed from the other side. Similarly, intersections can be marked as requiring a minimum number of additional guards so that at least one has an unoccluded view of the area to be guarded at all times. Alternatively, if available, UAVs with specifically downward or forward-pointing sensors may be used exclusively for sweeping to always provide overhead views. Of course, where no unoccluded perspective for a given point is possible, an adversarial target may be assumed to hide in this location, which must perpetually be treated as a source of contamination (and its exits always always guarded). A canonical instance of this is a highway overpass, under which a target may always hide from aerial vehicles, strongly motivating cooperation with ground agents or the use of small UAVs capable of flying among such obstacles.

CHAPTER 7

Conclusion

This thesis considered the problem of continuous search and tracking with teams of unmanned aircraft with focus on road network environments, which has been noted as being most closely related to the previously defined cooperative search, acquisition, and tracking (CSAT) problem, disregarding certain elements of cooperation and acquisition for the purposes of scope. Decomposed in Chapter 2, a given portion of a mission may be classified as a search, recapture, or pursuit task, each of which may be extremely hard problems themselves.

A substantial body of existing work in aerial geolocation, pursuit, and search was covered in Chapter 3, yet practical performance frequently remains poor, especially when using small field-reconnaissance UAVs given their limited sensing accuracy, maneuverability, and ground-station or on-board computation. For any class of UAVs, vehicle or sensor-pointing trajectory planning tractability is a particular challenge given that a myriad of possible target actions, UAV or sensor motions, and erroneous observations given sensing uncertainty must be considered.

Chapter 4 demonstrated practical improvements in visual tracking to produce observations and of filtering these for geolocation under high-uncertainty, while exploring the performance benefits of exploiting road and road network constraints on target position and actions. Initial demonstration of the applicability of conventional recursive Bayesian estimation applied to cellular environment decompositions of road networks indicates viability for pursuit tasks, however limited resolution and pursuit trajectory intractability suggest a strong need for parametric continuous represen-

tations. Thus, a remaining avenue for exploration is the application of information-theoretic planning metrics to variations of existing multi-hypothesis estimators for high-performance pursuit trajectory generation. In the geolocation realm, new directions include continued evaluation of the proposed geolocation filters and other variants to best understand under what circumstances each should be used, as well as further development for differing target motion models.

Search and coverage for mapping or the discovery of non-adversarial targets was examined in Chapter 5, starting with a simple coverage search that, though more appropriate for open-areas than road networks, is highly applicable to the class of small UAVs considered here, represents a useful point of comparison, and has proved very useful in live field testing. The effectiveness and difficulties of performing probabilistic efficient search in road networks using recently studied camera observation modeling for classical Bayesian estimation were briefly explored, with a field-of-view approximation proposed that substantially reduces motion planning computation time without compromising task performance. Finally, an efficient road network coverage algorithm based on several variations of Traveling Salesman Problem (TSP) abstractions was presented, providing rapid search of road networks of varying density with single or multiple UAV teams. In the area of efficient search, the most gaping hole in remaining work is a compelling demonstration of probabilistic search in road networks using state-of-the-art trajectory generation—perhaps a form of randomized planning—and exploration of representations that may be more appropriate than road segment discretization, which may include particles or higher-level reasoning about edge probability distribution that may build upon parametric estimator representations proposed for pursuit. Possible extensions to work on TSP-based coverage include primarily improvements in multi-vehicle distribution of effort.

Finally, Chapter 6 studied search and, in local form, recapture of potentially-adversarial targets with guarantees subject to sensor detection certainty. Algorithms built on several simple concepts drawn from classical graph search were demonstrated for both infinite-speed and bounded-speed target motion models and evaluated extensively under different circumstances. This represents merely a first step in the application of powerful pursuit-evasion abstractions to aerial search, remarkably little of which has been seen to date. A number of different alternative abstractions worthy of future exploration were proposed, each providing the potential for higher-performance taking better advantage of the wide field of view and unobstructed motion of UAVs as compared to ground searchers, though each with substantial respective difficulties as well.

Throughout, each aspect was validated in either real-world field trials using widely-fielded air vehicles or in realistic simulation using similarly parametrized mod-

els, providing direct applicability to demonstration on board real aircraft. This work directly contributes to a strong practical need for greater automation of small UAV teams to reduce operator workload and enable rapid completion of urgent missions with minimal resources.

7.1 Possible Extensions and Future Work

A number of closely related issues or avenues for extension remain unaddressed by this thesis. Several of these include:

Operation beyond roads This thesis focused specifically on road networks, but real-world environments contain both large and small open areas as well. Most described geolocation methods of Chapter 4 operate in unconstrained areas without modification, albeit without the space reduction provided by a road constraint. Coverage and efficient search may be applied to more general areas by way of tessellation by cells, orbits, or small virtual road segments as appropriate, after which strategies described in Chapter 5 may still be applied. Finally, via decomposition such as that suggested in Section 2.6, complex environments may be macroscopically viewed as a sparsely connected collection of open areas, allowing graph-based search abstractions such as those of Chapter 6 to still be applied.

Target occlusion In practice, environments may have obstacles of sufficient height to shadow targets nearby. Reasoning about these effects can eliminate a large proportion of unexpected observation dropouts. Frequently, it may be the case that such obstacles are well known in advance, such as would be the case for most buildings, which are constructed and demolished very infrequently. Using such pre-existing data, directional visibility models may be generated, describing perspectives from which any location is either visible or occluded (particularly for a given sensor pointing mode, e.g. a side-facing camera). Throughout this thesis, opportunities for applying such an occlusion model have been suggested. These have included, for instance, use within a detection model for information-theoretic planning used for pursuit or search, removing occluded perspectives from generalized TSP clusters for road network coverage in Section 5.3, and enforcing certain sweep directions or guard patterns for guaranteed search (Chapter 6). Much existing work on the general issue of occlusion in local aerial surveillance may be drawn upon for further inspiration, some of which is referenced in Section 3.2. Overall, given the ease with which

evasive targets may leverage natural occlusion to their advantage, particularly so as to frustrate guaranteed capture strategies, the following suggestion for direct collaboration with ground agents is made with the hope that combined air and ground viewpoints provide total coverage.

Ground agents Described field work in air-ground collaboration has shown clear value in pairing air and ground vehicles given their complementary capabilities. Including ground agents in proposed strategies provides observations having a very complementary sensing model that potentially overcomes the aforementioned issue of occlusion, permits physical interaction with targets, and enables direct application of pursuit-evasion abstractions intended for planar (ground) vehicles.

Gimbaled sensor aiming As first stated in Section 2.7, a fixed side-pointing camera is generally assumed throughout this thesis for simplicity and maximal applicability across vehicle platforms. However, reasoned use of a gimbal could greatly improve task performance. Section 3.2 refers to several existing efforts to integrate a gimbal into missions. More generally, a gimbal could improve pursuit by effectively increasing the field of view, reducing loss frequency, increasing time in view, and by integration into information-theoretic path planning to generate more observations from preferable relative poses (jointly produced by vehicle pose and sensor pointing). In probabilistic search, a vehicle equipped with a rapidly-moving gimbal could scan it while moving the vehicle, sweeping a larger footprint. Smaller pockets of residual target probability could also often be covered by a gimbal re-orientation rather than (slowly, for a fixed-wing vehicle) turning the aircraft around. In a guaranteed search scenario, a gimbal could be used to transition between side- and front-pointing orientations for optimal sweeping in urban canyons while also providing persistent single-vehicle guarding by orbiting. It may also enable a single vehicle to perform a guard maneuver with alternative patterns such as a figure-8 rather than an orbit in case this should this place the vehicle at a more convenient terminal pose for the next task.

Pursuit of multiple targets Though the coverage and search strategies described naturally handle multiple targets (or additional hypotheses for locations), exploration of pursuit has not. In an ad-hoc fashion, teams might be divided explicitly between targets, and existing work in task allocation may be applicable to formalize this.

Decentralized operation In practice, failures of individual agents are inevitable,

and communication bandwidth is highly limited. Applying and extending existing strategies such as decentralized estimation and anonymous cooperation for reducing reliance on a central ground station and the amount of data that must be transmitted by each vehicle would improve both the robustness and practical applicability of these systems.

Mission robustness Small UAVs, especially in large numbers, are prone to inevitable failure—in the form of total agent failure if not simply substantial trajectory tracking failure—and it is vital to handle this in an effective physical system. Balancing reliability and redundancy is an open question, particularly in search tasks in which the failure of a single agent guarding a key location could result in recontamination of the entire environment. Similarly, the mission longevity of small UAVs may not be sufficient for long search schedules, and it is necessary to perform hand-off without loss of state.

Alternative form-factors Extension of these ideas to other vehicle types such as micro air-vehicles—presently a popular research area—would broaden the usefulness of this work. Smaller vehicles can operate in dense urban environments to provide closer inspection and viewing from angles potentially occluded for larger vehicles (e.g. in urban canyons). However, their use adds additional complexities such as concern about obstacle avoidance, reduced mission longevity, and a general inability to cover large areas. On the other end of the scale, operation on larger aircraft may also be desirable. Though better sensing is likely to be available on such vehicles, higher speed and altitude provide their own respective increases to target geolocation uncertainty. Further, as noted initially, sensing may take place through a sub-window of a larger sensor, which must be directed towards points of interest, with some delay and uncertainty as to its exact coordinates. This is directly analogous to moving a smaller vehicle between points of interest, and the enumerated search and coverage algorithms remain directly applicable. Another particularly interesting area of study stems from greater available onboard computation (which is still far below that available on the ground), raising questions such as how to process larger amounts of sensor data than can be downlinked in real time and which decisions to make autonomously when onboard sensor control (e.g. of a gimbal) can occur at much higher bandwidth than control via a remote link.

Appendices

APPENDIX A

Robust Automatic Visual Tracking from Small UAVs

Operating a UAV in a surveillance mission can be a heavy burden on a human user who is viewing the UAV's video stream sent via radio communications to a ground station. Any autonomy available in the process to assist the user is very valuable. We have developed a user-initiated visual tracking system that takes a cue from the operator of the location of an object of interest in an image and then autonomously maintains track of the object in the video stream. The UAV's restricted payload capabilities require their on-board sensors/computing/communications to be low weight and therefore low-grade and low performance, making visually tracking objects of interest from UAVs very difficult. Furthermore, these vehicles low weight and hence low inertia make them susceptible to buffeting from the wind, causing high camera motion and therefore even more difficulties in visual tracking. Our real-time visual tracking system has many aspects designed to overcome the difficulties of the specific application domain. The system operates with high levels of camera motion, performs out-of-view tracking recovery, tracking recovery from communications corruptions/loss and adaptively updates a template-model of the object being tracked.

The primary external sensor assumed to be present on small UAVs is one or more cameras, as others such as radars and laser rangefinders are generally of prohibitive size and cost. Therefore, the targeting procedure envisioned is that as a UAV covers an area of interest, targets visible in the video feed are designated by an operator click or an automatic detection algorithm, these designated targets are

then tracked through succeeding video frames without further operator intervention by one of several algorithms, and a recovery procedure is executed to attempt to re-acquire targets that have become occluded by terrain or that have left the view of the camera. The system developed handles high levels of camera motion caused by wind, performs out-of-view tracking recovery that is frequently required in practice, and handles tracking and recovery through video corruption or blackouts caused by communications glitches.

For the purposes of this work, it is assumed that the camera does not have variable zoom capability or a mechanical gimbal for pointing. The Raven has both electronic zoom and software panning abilities, however these are not used both for greater applicability to other UAVs and because these were not designed for automatic software control in the case of the Raven. The algorithms described here will operate perfectly on a vehicle with either and may be extended to take advantage of them.

Image Stabilization

Possibly the greatest challenge in visual tracking from a lightweight UAV is rapid unwanted camera motion caused by wind-induced attitude disturbances. This causes an objects location in one video frame might be significantly different in subsequent frames, creating failures in many traditional tracking schemes and also making it difficult for a human user to initially designate a target in the video stream.

The address this, the first stage of our tracking system performs image stabilization of the live video stream. This scheme operates with a planar ground assumption, which is not strictly true, but given the altitude of the UAV is much higher relative to the undulations on the ground, this assumption is approximately correct. Two neighboring image frames are analyzed to find the planar transform that minimizes some objective function. For us, the objective function is a global image difference. The typical alternative is to perform point-feature tracking of arbitrary ground features and minimize the reprojection error of the tracked points to an estimated geometric transform. We have implemented both, but find the point-feature approach fails when flying over relatively featureless terrain, such as open fields or dense forests. Global image alignment is more robust in such featureless scenes. One notable aspect of our alignment is how we find the optima in the objective function. Often the objective function is linearized at an initial estimate and the gradient of the function is found and the estimate updated accordingly and the process is repeated. The Lucas-Kanade alignment [10] is one of the formative approaches. However, with large amounts of image motion, often the initial estimate is too far away from a

global optimum and the gradient directs the algorithm into a local optima away from the correct answer. An approach to avoid this problem is to perform the alignment coarse-to-fine, starting at a low resolution first. However, the image motion experienced from a UAV camera can be so large that the gradients calculated from the linearization of the objective function are still not accurate and the algorithm will still descend into local optima. We avoid local optima all together by sampling a discrete set of estimates from the objective function and choosing the optima from a global the set. An example of this visual stabilization algorithm in action is given in Figure A.1.

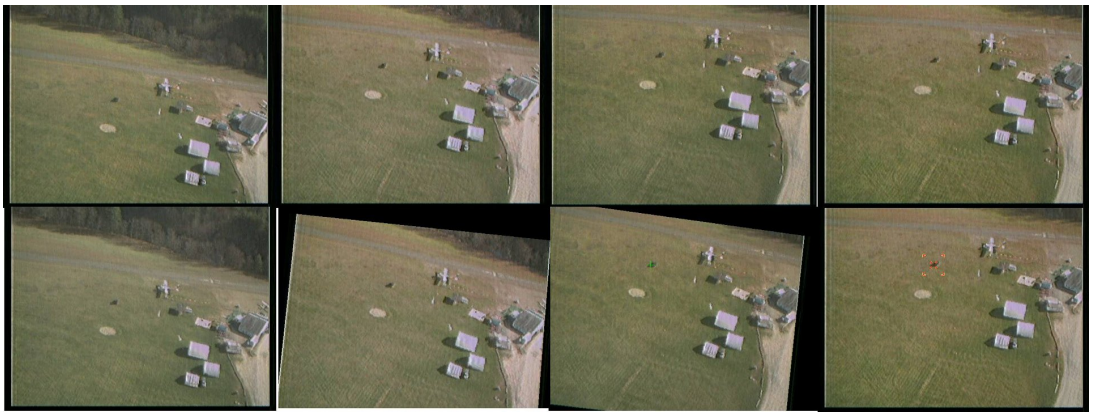


Figure A.1: An example sequence illustrating the image stabilization algorithm that is used to assist an operator designate a target. The top row shows the raw video, and the bottom row shows the stabilized output. The bottom right image shows the commencement of the autonomous target tracking after the designation at the highlighted pixel.

Target Designation

In contrast to typical existing systems that provide for target designation in the form of highlighting a target in previously captured imagery or by using a joystick to continuously aim a gimbal or cursor, the system developed here provides a simple means for operator-assisted target designation. First, the operator enters target designation mode, during which the live video stream is stabilized as described to produce a synthetic view as though the UAV were not moving. Next the operator may either draw a box around the target that is then automatically sized to best fit the identified target or click on the center of the target, causing an estimated box to be drawn around it that is then likewise resized. Resizing is performed by initiating one of the following tracking methods and sampling bounding box sizes to find the

one providing the best separation between area within it and the surrounding area for the tracker.

While in general target identification is an extremely difficult problem, especially considering the relatively low-resolution cameras assumed here, methods for automatic target designation have been explored. Among these is moving target detection, which compares a sequence of stabilized video frames, subtracts out consistent unchanging regions between them, and highlights portions that have moved as candidate targets. Similar to existing schemes that implement this idea [137, 6], we have evaluated an implementation of our own that attempts to address the issue of tall objects appearing to a low-flying aircraft as mis-identified motion due to parallax by considering only moving areas of a minimum size that correspond to a sequence of consistent straight-line motion, such as from moving vehicle. Initial experiments have proved to be promising, albeit requiring computational resources at the high-end of the scale. Other promising ideas for future work include taking a target appearance model from one robot's video stream in which tracking was just initiated and applying it to the video of any other vehicles looking at the same area so that the operator need not click in multiple streams, further reducing workload.

Mean-shift Color Tracking

Once a target is designated, an automatic visual tracking method is applied to identify the target's location in succeeding video frames without the need for the operator to continuously highlight it. We have evaluated two such methods, both relying on the use of a template of the target generated from the initial video frame.

The first of these builds a color model of the target in the form of a histogram of color values appearing in the region initially designated as the target and finds the nearby region in succeeding video frames whose color distribution most closely matches the template histogram. We use a direct implementation of this concept [27] as well as a variation of an alternative [26] that continuously reweights a set of features derived from functions of color values (encoding higher-level features such as intensity and chrominance) to explicitly maximize the template's discrimination between the target and the surrounding background. Both have been used with considerable success and use the well-known mean-shift algorithm to rapidly converge on the location of the target between frames with minimal computational effort, starting at an initial guess location lying where the target lay in the previous image.

Spatial Patch Tracking

An alternative approach to visual tracking is to construct a spatial appearance template of the target, treating it as a patch in the image that may move between video frames. One of the most well-known of these is the KLT algorithm [127]. We have developed an algorithm similar to KLT that provides greater robustness to large camera motions and addresses changing target appearance.

This approach initializes the template as the image patch around the target location that the user designates (see Figure A.2) and adapts it at each subsequent frame to account for the changing appearance of the target. Formally, the tracking algorithm can be expressed by the following equation:

$$\mathbf{p}_n = \underset{\mathbf{p}}{\operatorname{argmin}} \sum_{\mathbf{x} \in T} [I_n(\mathbf{W}(\mathbf{x}; \mathbf{p})) - T(\mathbf{x})]^2 \tag{A.1}$$

where $\mathbf{p}_n$ is the motion parameters that warp the template onto the current image frame, I_n. The warp function is $T(\mathbf{W}(\mathbf{x}; \mathbf{p}))$ that takes every pixel, $\mathbf{x}$, from the coordinate frame of the template T and maps it to the coordinate frame of the current video I_n. Equation A.1 can be summarized as finding the warp which minimizes the summation of the error between the warped template and the current image.

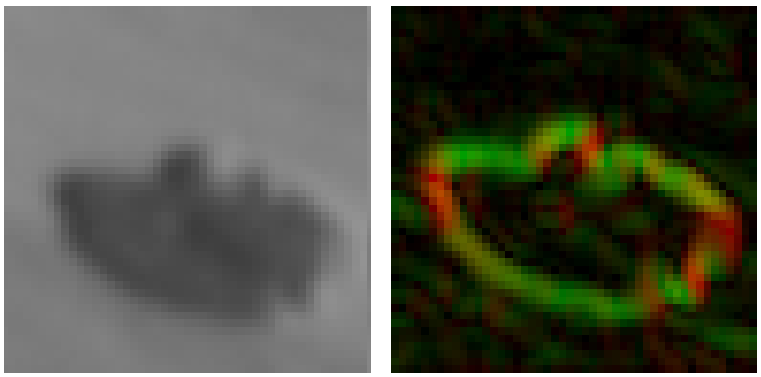
Typically, as in the canonical KLT framework, for video rate tracking the assumption is that within the period between two frames, the deformation of the object's appearance will be minimal and the warp can be simplified as a 2D translation. The KLT framework linearizes the objective function around initial motion/warp parameters, $\mathbf{p}$, to estimate a warp update, $\mathbf{\Delta p}$:

$$\mathbf{p} \leftarrow \mathbf{p} + \mathbf{\Delta p} \tag{A.2}$$

which hopefully will be descending towards the correct solution to Equation A.1. The estimation of warp updates is iterated until convergence. The typical test for convergence is when the norm of $\mathbf{\Delta p}$ is below some threshold ϵ. However, in our case the camera motion is unusually large, and often the image motion can be on the order of the size the template itself. In this case, the initial motion parameters will be so far away from the correct values of $\mathbf{p}_n$ that the linearization of Equation A.1 will be meaningless, causing the $\mathbf{\Delta p}$ estimates to be incorrect and likely causing tracking to fall into an incorrect local minimum.

Our alternative approach is to make a dense sampling of possible motion parameters $\mathbf{p}$, and for each calculate a sum of squared difference between the warped template and the current image, and take the maximal likelihood of these estimates

for our solution to $\mathbf{p}_n$. This approach avoids local minima at the cost of the computation of the dense sampling. However, because we track a single target per video stream whose template is typically small, the computational overhead is not significant.



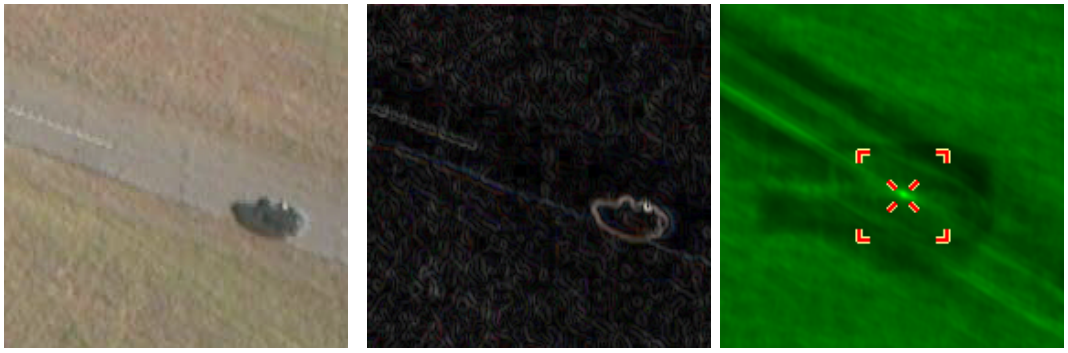
(a) Raw image (b) Sobel transform

Figure A.2: The image patch template that is tracked in the video stream.

Another slight modification of our approach is to use a Sobel transform version of the template image as input to our objective function, depicted in Figure A.2(b). The Sobel transform makes the approach more robust to fast illumination, fast exposure changes, and any compression noise in the video stream. Figure A.3(c) shows an example of the objective function output over the entire parameter space and the selected parameters are designated by a crosshair. There is a notable peak at the optimum but the peak is very narrow and there are other minima in the search window that gradient descent type algorithms are likely to get trapped inside of when the camera motion is high.

Tracker Selection

Clearly, there exist scenarios in which each type of tracker may perform better than the other. Bright, distinctly-colored targets are best tracked using a mean-shift color tracker, while the patch tracker performs well with indistinctly colored but consistently-shaped targets. More subtle strengths and weaknesses are present, however. For instance, as shown in Figure A.4, the patch tracker requires continuous tuning of the template to handle changes in perspective that non-spatial mean-shift algorithms are largely unaffected by and performs poorly if too much background that will not move with the target lies within the template, while environments with



(a) Raw image (b) Sobel image (c) Tracking likelihood

Figure A.3: Examples showing the search for the patch within a region.

similar objects confound non-spatial mean-shift. On average, the patch tracker performs best and is selected by default. At any time during a mission, the operator may choose an alternate tracker if appropriate.

(a) Changing perspective (b) Similar objects (c) Background in template

Figure A.4: Examples showing two cases, (a) and (c), for which non-spatial tracking dominates, and another, (b), for which a spatial template is more appropriate. This figure is duplicated for viewing convenience from Figure 4.3.

Tracking Template Adaptation

As the UAV flies around a target, the perspective change causes a change in the targets appearance in the image. The template update problem is a well documented problem, requiring a compromise between adapting to the current appearance of the object and unwanted model drift. Matthews et al. [83] discuss three options, first *no update*, where the template at initialization never changes. This they say will undoubtedly will become out-of-date quickly as the viewpoint with respect to

the object changes. The second approach is a *naive update*, where the template is updated to exactly the new tracked patch in the current image. This approach is explained by Matthews et al. as suffering from drift from accumulated small tracking errors. A third update approach is suggested by Matthews et al., *template update with drift correction*, where the template is updated completely to the new tracked location only when the estimated motion is below a certain threshold. Their new approach again updates the template by replacing it with the current tracked patch, which again will actually be susceptible to model drift if the motion threshold does not capture the small tracking errors. We avoid the drifting problem by adjusting the template by mixing it with the current patch, instead of a direct replacement. This helps adjust the template's appearance over time to the current appearance of the target, while also doing the merging at a slow enough rate to avoid drift, thereby allowing the tracker to continue uninterrupted. An example of the adaptive template operating as the vehicle flies around a target is shown in Figure A.5.

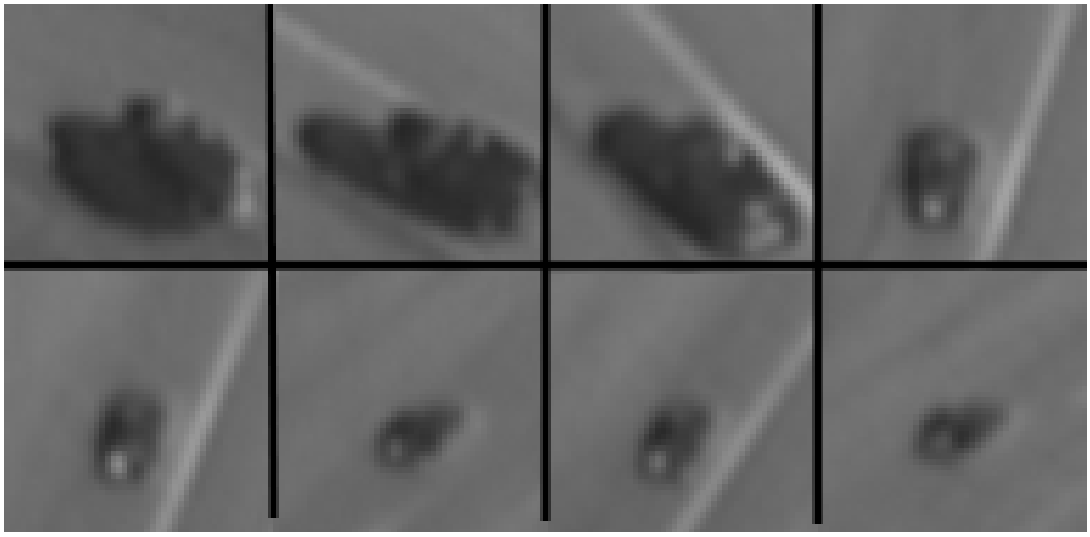


Figure A.5: An example sequence illustrating the template of a target being updated as the UAV flies around it. The target appears quite different from different positions on the UAV orbit.

Tracking Recovery

Inevitably, a tracked target will often leave the view of a UAV's camera, especially when lacking a gimbal or if zoomed in, reducing the field of view. We recover from

out-of-view tracking losses by estimating the camera motion and predicting the new location of the target, and when the predicted location re-enters the view, we attempt to search for the target inside a sizeable search window. In addition to out-of-view events, other factors may induce tracker failure. For instance, analog and digital radio communications loss causes corruption in the video stream. We detect failures in the tracker when the template no longer matches well with the search window and then attempt to recover the target once the period of corruption is over. An example of this procedure applied to both video corruption and an out-of-frame event is shown in Figure A.6.

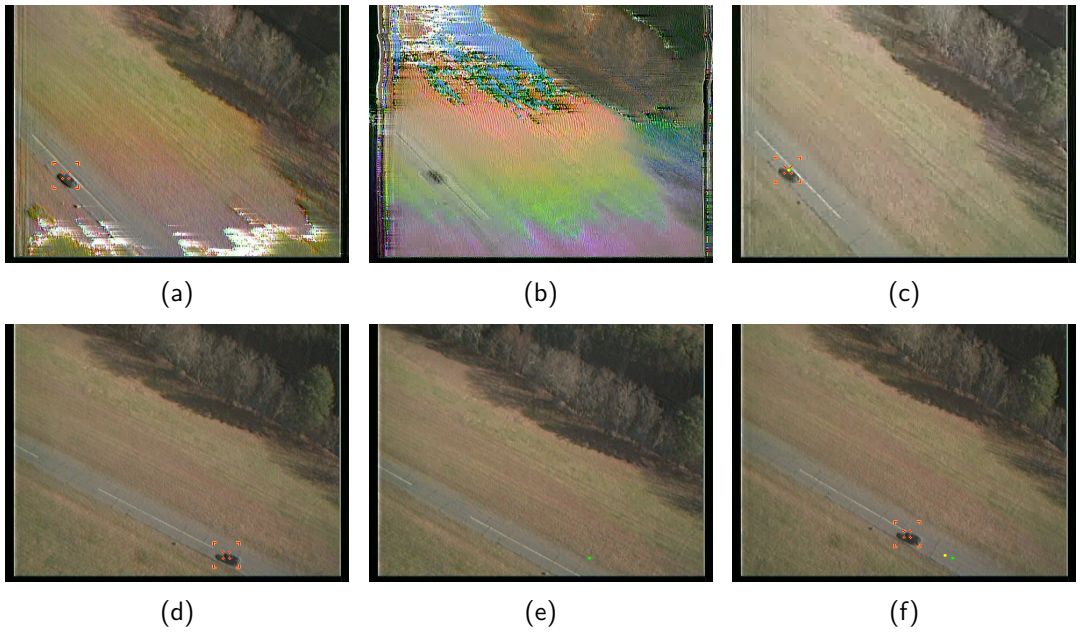


Figure A.6: Different types of tracking recovery. The top row shows the tracking recovery after a period of communications corruption. The bottom row shows the tracking recovery after the target leaves the field of view. This figure is duplicated for viewing convenience from Figure 4.4.

Bibliography

[1] Vitaly Ablavsky and Magnús Snorrason. Optimal search for a moving target: A geometric approach. In *AIAA Conference on Guidance, Navigation, and Control*, 2000.

[2] Micah Adler, Harald Räcke, Naveen Sivadasan, Christian Sohler, and Berthold Vöcking. Randomized pursuit-evasion in graphs. *Combinatorics, Probability and Computing*, 12:225–244, May 2003.

[3] AeroVironment Inc. Raven Product Data Sheet. `http://www.avinc.com/downloads/Raven_Domestic_1210.pdf`, December 2010.

[4] AeroVironment, Inc. AeroVironment develops worlds first fully operational life-size hummingbird-like unmanned aircraft for DARPA. Press Release, 2011.

[5] Ali Ahmadzadeh, James Keller, George J. Pappas, Ali Jadbabaie, and Vijay Kumar. Multi-UAV cooperative surveillance with spatio-temporal specifications. In *IEEE Conference on Decision and Control*, 2006.

[6] Saad Ali and Mubarak Shah. COCOA - tracking in aerial imagery. In *SPIE Airborne Intelligence, Surveillance, Reconnaissance Systems and Applications*, 2006.

[7] Evan D. Andersen and Clark N. Taylor. Improving MAV pose estimation using visual information. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3745–3750, November 2007.

[8] David Applegate, Robert Bixby, Vasek Chvatal, and William Cook. Concorde TSP Solver. `http://www.tsp.gatech.edu/concorde.html`, December 2011.

[9] M. Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.

[10] Simon Baker and Iain Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004.

[11] Yaakov Bar-Shalom and X. Rong Li. *Estimation and Tracking: Principles, Techniques, and Software*. Artech House, 1993.

[12] D. Blake Barber, Joshua D. Redding, Timothy W. Mclain, Randal W. Beard, and Clark N. Taylor. Vision-based target geo-location using a fixed-wing miniature air vehicle. *J. Intell. Robotics Syst.*, 47(4):361–382, 2006.

[13] Lali Barrière, Pierre Fraigniaud, Nicola Santoro, and Dimitrios M. Thilikos. Searching is not jumping. In *Workshop on Graph Theoretic Concepts in Computer Science*, pages 34–45, 2003.

[14] Randal W. Beard and Timothy W. McLain. Multiple UAV cooperative search under collision avoidance and limited range communication constraints. In *IEEE Conference on Decision and Control*, 2003.

[15] Stanley J. Benkoski, Michael G. Monticino, and James R. Weisinger. A survey of the search theory literature. *Naval Research Logistics*, 38(4):469–494, August 1991.

[16] Deepak Bhadauria and Volkan Isler. Capturing an evader in a polygonal environment with obstacles. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[17] Frederic Bourgault, Tomonari Furukawa, and Hugh Durrant-Whyte. Coordinated decentralized search for a lost target in a Bayesian world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2003.

[18] Harry Q. Bovik. On the inevitability of manic-depressive behaviour in artificially intelligent programs. *Journal of Personality and Social Psychology*, 21, 1984.

[19] Kai Briechle and Uwe Hanebeck. Localization of a mobile robot using relative bearing measurements. In IEEE *Transactions on Robotics and Automation*, volume 20(1), pages 36–44, 2004.

[20] Mitch Bryson and Salah Sukkarieh. Bearing-only SLAM for an airborne vehicle. In *Australasian Conf. on Robotics and Automation*, 2005.

[21] Fernando Caballero, Luis Merino, Joaquin Ferruz, and Anibal Ollero. Vision-based odometry and SLAM for medium and high altitude flying UAVs. *J. Intell. Robotics Syst.*, 54(1-3):137–161, 2009.

[22] Mark E. Campbell and William W. Whitacre. Cooperative tracking using vision measurements on SeaScan UAVs. *IEEE Transactions on Control Systems Technology*, 15(4):613–626, 2007.

[23] David Casbeer, Pengcheng Zhan, and A. Lee Swindlehurst. A non-search optimal control solution for a team of MUAVs in a reconnaissance mission. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2006.

[24] Howie Choset and Philippe Pignon. Coverage path planning: The boustrophedron decomposition. In *Proceedings of Field and Service Robotics (FSR)*, Canberra, Australia, December 1997.

[25] Timothy H. Chung, Geoffrey A. Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics. *Autonomous Robots*, 31(4):299–316, 2011.

[26] Robert T. Collins and Yanxi Liu. On-line selection of discriminative tracking features. In *IEEE Conf. on Computer Vision (ICCV)*, pages 346–352, 2003.

[27] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 142–149, 2000.

[28] Gianpaolo Conte and Patrick Doherty. Vision-based unmanned aerial vehicle navigation using geo-referenced information. *EURASIP J. Adv. Signal Process*, 2009:1–18, 2009.

[29] Gianpaolo Conte, Maria Hempel, Piotr Rudol, David Lundström, Simone Duranti, Mariusz Wzorek, and Patrick Doherty. High accuracy ground target geo-location using autonomous micro aerial vehicle platforms. In *AIAA Conference on Guidance, Navigation, and Control*, August 2008.

[30] Jeffrey B. Corbets and Jack W. Langelaan. Parameterized trajectories for target localization using small and micro unmanned aerial vehicles. In *American Control Conference*, 2008.

[31] Michael Dille, Ben Grocholsky, and Stephen Nuske. Persistent visual tracking and accurate geo-location of moving ground targets by small air vehicles. In *AIAA Infotech@Aerospace Conference*, March 2011.

[32] Michael Dille, Ben Grocholsky, Stephen Nuske, Mark Moseley, and Sanjiv Singh. Air-ground collaborative surveillance with human portable hardware. In *AUVSI Unmanned Systems North America*, August 2011.

[33] Joseph Djugash, Sanjiv Singh, and Benjamin P. Grocholsky. Modeling mobile robot motion with polar representations. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, October 2009.

[34] Lester Dubins. On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents. *American Journal of Mathematics*, 79:497–516, 1957.

[35] Andrzej Dudek, Przemyslaw Gordinowicz, and Pawel Pralat. Cops and robbers playing on edges. *Journal of Combinatorics*, submitted, 2014.

[36] Jack Edmonds and Ellis L. Johnson. Matching, euler tours, and the chinese postman. *Mathematical Programming*, 5(1):88–124, 1973.

[37] Mats Ekman and Egils Sviestins. Multiple model algorithm based on particle filters for ground target tracking. In *International Conference on Information Fusion*, 2007.

[38] Dale Enns, Dan Bugajski, and Steve Pratt. Guidance and control for cooperative search. In *American Control Conference*, 2002.

[39] Mariam Faied, Ahmed Mostafa, and Anouck Girard. Vehicle routing problem instances: Application to multi-uav mission planning. In *AIAA Conference on Guidance, Navigation, and Control*, 2010.

[40] Matthew D. Flint. *Cooperative Unmanned Aerial Vehicle (UAV) Search in Dynamic Environments Using Stochastic Methods*. PhD thesis, University of Cincinnati, 2004.

[41] Fedor V. Fomin and Dimitrios M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science*, 399(3):236–245, June 2008.

[42] Eric W. Frew. Cooperative standoff tracking of uncertain moving targets using active robot networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[43] Eric W. Frew and Jack Elston. Target assignment for integrated search and tracking by active robot networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[44] Eric W. Frew, Jack Langelaan, and Maciej Stachura. Adaptive planning horizon on information velocity for vision-based navigation. In *AIAA Conference on Guidance, Navigation, and Control*, 2007.

[45] Eric W. Frew and Dale Lawrence. Cooperative stand-off tracking of moving targets by a team of autonomous aircraft. In *AIAA Guidance, Navigation, and Control Conference*, 2005.

[46] Eric W. Frew and Stephen M. Rock. Trajectory generation for constant velocity target motion estimation using monocular vision. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2003.

[47] Brian R. Geiger, Joseph F. Horn, Anthony M. DeLullo, Lyle N. Long, and Albert F. Niessner. Optimal path planning of UAVs using direct collocation with nonlinear programming. In *AIAA Conference on Guidance, Navigation, and Control*, 2006.

[48] Brian P. Gerkey, Sebastian Thrun, and Geoff Gordon. Visibility-based pursuit-evasion with limited field of view. *International Journal of Robotics Research*, 25(4):299–315, April 2006.

[49] Christopher Geyer. Active target search from UAVs in urban environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[50] Danny Gibbins, Philip Roberts., and Leszek Swierkowski. A video geo-location and image enhancement tool for small unmanned air vehicles (UAVs). In *Intelligent Sensors, Sensor Networks and Information Processing Conference*, pages 469–473, Dec. 2004.

[51] Matthew Gibson. *Clusters and covers: geometric set cover algorithms*. PhD thesis, University of Iowa, 2010.

[52] Anouck R. Girard, Adam S. Howell, and J. Karl Hedrick. Border patrol and surveillance missions using multiple unmanned air vehicles. In *IEEE Conference on Decision and Control*, 2004.

[53] Arthur S. Goldstein and Edward M. Reingold. The complexity of pursuit on a graph. *Theoretical Computer Science*, 143(1):93 – 112, 1995.

[54] Michael A. Goodrich, Bryan S. Morse, Damon Gerhardt, Joseph L. Cooper, Morgan Quigley, Julie A. Adams, , and Curtis Humphrey. Supporting wilderness search and rescue using a camera-equipped mini UAV. *Journal of Field Robotics*, 25(1):89–110, 2008.

[55] Ben Grocholsky, Michael Dille, and Stephen Nuske. Efficient target geolocation by highly uncertain small air vehicles. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, September 2011.

[56] Leonidas Guibas, Jean-Claude Latombe, Steven Lavalle, David Lin, and Rajeev Motwani. Visibility-based pursuit-evasion in a polygonal environment. In *Workshop on Algorithms and Data Structures*, pages 17–30, 1997.

[57] Ruijie He, Abraham Bachrach, and Nicholas Roy. Efficient planning under uncertainty for a target-tracking micro-aerial vehicle. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2010.

[58] Tamir A. Hegazy. *A Distributed Approach to Dynamic Autonomous Agent Placement for Tracking Moving Targets with Application to Monitoring Urban Environments*. PhD thesis, Georgia Institute of Technology, 2004.

[59] Marcel L. Hernandez. Optimal sensor trajectories in bearings-only tracking. In *International Conference on Information Fusion*, 2004.

[60] Gabriel M. Hoffmann, Steven L. Waslander, and Claire J. Tomlin. Distributed cooperative search using information-theoretic costs for particle filters, with quadrotor applications. In *AIAA Conference on Guidance, Navigation, and Control*, 2006.

[61] Geoffrey A. Hollinger. *Search in the Physical World*. PhD thesis, Carnegie Mellon University, 2010.

[62] Mong–ying A. Hsieh, Luis Chaimowicz, A. Cowley, Ben Grocholsky, Jim Keller, Vijay Kumar, C. J. Taylor, Y. Endo, Ronald Arkin, Boyoon Jung, Denis F. Wolf, Gaurav S. Sukhatme, and Doug MacKenzie. Adaptive teams of autonomous aerial and ground robots for situational awareness. *Journal of Field Robotics*, 24(11):991–1014, 2007.

[63] Jason T. Isaacs, Daniel J. Klein, and João P. Hespanha. Algorithms for the traveling salesman problem with neighborhoods involving a dubins vehicle. In *American Control Conference*, 2011.

[64] Rufus Isaacs. Games of pursuit. Technical Report P-257, RAND Corporation, Santa Monica, CA, November 1951.

[65] Rufus Isaacs. *Differential Games*. John Wiley, 1965.

[66] Volkan Isler, Sampath Kannan, and Sanjeev Khanna. Randomized pursuit-evasion in a polygonal environment. *IEEE Transactions on Robotics*, 21(5):875–884, October 2005.

[67] Gregory F. Ivey and Eric N. Johnson. Investigation of methods for target state estimation using vision sensors. In *AIAA Conference on Guidance, Navigation, and Control*, 2005.

[68] Phillip J. Jones. *Cooperative Area Surveillance Strategies Using Multiple Unmanned Systems*. PhD thesis, Georgia Institute of Technology, 2009.

[69] Roy Jonker and Ton Volgenant. Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters*, 2(4):161–163, November 1983.

[70] Athanasios Kehagias, Geoffrey Hollinger, and Sanjiv Singh. A graph search algorithm for indoor pursuit/evasion. *Mathematical and Computer Modeling*, 50:1305–1317, 2009.

[71] Jongrae Kim and Yoonsoo Kim. *Optimal circular flight of multiple UAVs for target tracking in urban areas*. IN-TECH, 2009.

[72] Thiagalingam Kirubarajan, Yaakov Bar-Shalom, Krishna R. Pattipati, and Ivan Kadar. Ground target tracking with variable structure IMM estimator. *IEEE Transactions on Aerospace and Electronic Systems*, 36(1):26–46, January 2000.

[73] Andreas Kolling and Stefano Carpin. Pursuit-evasion on trees by robot teams. *IEEE Transactions on Robotics*, 26(1):32–47, 2010.

[74] Bernard O. Koopman. The theory of search, part III: The optimum distribution of searching effort. *Operations Research*, 5(5):613–626, October 1957.

[75] Swastik Kopparty and Chinya V. Ravishankar. A framework for pursuit evasion games in Rn. *Information Processing Letters*, 96(3):114–122, November 2005.

[76] Kalyanam Krishnamoorthy, Swaroop Darbha, Pramod Khargonekar, David Casbeer, Phil Chandler, and Meir Pachter. Optimal minimax pursuit evasion on a Manhattan grid. In *American Control Conference*, pages 3421–3428, 2013.

[77] Ratnesh Kumar and Haomin Li. On asymmetric tsp: Transformation to symmetric tsp and performance bound. Technical report, Iowa State University, Ames, IA, 1995.

[78] Andrea S. LaPaugh. Recontamination does not help to search a graph. *Journal of the Association for Computing Machinery*, 40(2):224–245, 1993.

[79] Dale A. Lawrence, Eric W. Frew, and William J. Pisano. Lyapunov vector fields for autonomous UAV flight control. *AIAA Journal of Guidance, Control, and Dynamics*, 31(5):1220–1229, September 2008.

[80] Jerome Le Ny. *Performance Optimization for Unmanned Vehicle Systems*. PhD thesis, Massachusetts Institute of Technology, 2008.

[81] Jusuk Lee, Rosemary Huang, Andrew Vaughn, Xiao Xiao, J. Karl Hedrick, Marco Zennaro, and Raja Sengupta. Strategies of path-planning for a UAV to track a ground vehicle. In *Autonomous Intelligent Networks and Systems Conference*, June 2003.

[82] Richard Madison, Paul DeBitetto, A. Rocco Olean, and Mac Peebles. Target geolocation from a small unmanned aircraft system. In *IEEE Aerospace Conference*, pages 1–19, March 2008.

[83] Iain Matthews, Takahiro Ishikawa, and Simon Baker. The template update problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:810–815, 2004.

[84] Ivan Maza and Anibal Ollero. Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *International Symposium on Distributed Autonomous Robotic Systems*, June 2004.

[85] Timothy G. McGee and J. Karl Hedrick. Guaranteed strategies to search for mobile evaders in the plane. In *American Control Conference*, 2006.

[86] Timothy G. McGee, Stephen Spry, and J. Karl Hedrick. Optimal path planning in a constant wind with a bounded turning rate. In *AIAA Conference on Guidance, Navigation, and Control*, 2005.

[87] Charles E. Noon and James C. Bean. An efficient transformation of the generalized traveling salesman problem. Technical Report 91-26, University of Michigan, Ann Arbor, MI, October 1991.

[88] Stephen Nuske, Michael Dille, Ben Grocholsky, and Sanjiv Singh. Representing substantial heading uncertainty for accurate target geolocation by small UAVs. In *AIAA Conference on Guidance, Navigation, and Control*, August 2010.

[89] Office of the U.S. Secretary of Defense. Unmanned aerial vehicles roadmap 2005-2030, 2005.

[90] Hyondong Oh, Seungkeun Kim, Antonios Tsourdos, and Brian White. Cooperative road-network search planning of multiple uavs using dubins paths. In *AIAA Conference on Guidance, Navigation, and Control*, 2011.

[91] Hyondong Oh, H.S. Shin, A. Tsourdos, B.A. White, and P. Silson. Coordinated road network search for multiple uavs using dubins path. *Advances in Aerospace Guidance, Navigation and Control*, pages 55–65, 2011.

[92] Torrence Parsons. Pursuit-evasion in a graph. In *Theory and applications of graphs*, pages 426–441. Springer, 1976.

[93] Valerii Semenovich Patsko and Varvara Leonidovna Turova. Homicidal chauffeur game: history and modern studies. Technical report, Russian Academy of Sciences, Ural Branch, Institute of Mathematics and Mechanics, Ekaterinburg, Russia, February 2009.

[94] N. N. Petrov. A problem of pursuit in the absence of information on the pursued. *Differentsial'nye Uravneniya*, 18:1345–1352, 1982.

[95] Marios M. Polycarpou, Yanli Yang, and Kevin M. Passino. A cooperative search framework for distributed agents. In *IEEE International Symposium on Intelligent Control (ISIC)*, 2001.

[96] Sameera S. Ponda. Trajectory optimization for target localization using small unmanned aerial vehicles. Master's thesis, Massachusetts Institute of Technology, 2008.

[97] Morgan Quigley, Michael A. Goodrich, Stephen Griffiths, Andrew Eldredge, and Randal W. Beard. Target acquisition, localization, and surveillance using a fixed-wing mini-UAV and gimbaled camera. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2600–2605, April 2005.

[98] A. Quilliot. *Problémes de jeux, de point fixe, de connectivité et de représentation sur des graphes, des ensembles ordonnés et des hypergraphes*. These d´Etat, Université de Paris VI, 1983.

[99] Steven A. P. Quintero, Francesco Papi, Daniel J. Klein, Luigi Chisci, and J. P. Hespanha. Optimal UAV coordination for target tracking using dynamic programming. In *IEEE Conference on Decision and Control*, 2010.

[100] Fahd Rafi, Saad M. Khan, Khurram H. Shafiq, and Mubarak Shah. Autonomous target following by unmanned aerial vehicles. In *SPIE Defense and Security Symposium*, Orlando, USA, 2006.

[101] Joshua Redding, Timothy W. Mclain, Randal W. Beard, and Clark Taylor. Vision-based target localization from a fixed-wing miniature air vehicle. In *American Control Conference*, 2006.

[102] Johan Reimann and George Vachtsevanos. UAVs in urban operations: Target interception and containment. *Journal of Intelligent Robotics Systems*, 47(4):383–396, 2006.

[103] Johan M. Reimann. *Using Multiplayer Differential Game Theory to Derive Efficient Pursuit-Evasion Strategies for Unmanned Aerial Vehicles*. PhD thesis, Georgia Institute of Technology, 2007.

[104] James A. Ross, Brian R. Geiger, Gregory L. Sinsley, Joseph F. Horn, Lyle N. Long, and Albert F. Niessner. Vision-based target geolocation and optimal surveillance on an unmanned aerial vehicle. In *AIAA Conference on Guidance, Navigation, and Control*, August 2008.

[105] Allison Ryan. Information-theoretic tracking control based on particle filter estimate. In *AIAA Conference on Guidance, Navigation, and Control*, 2008.

[106] Allison Ryan, Hugh Durrant-Whyte, and J. Karl Hedrick. Information-theoretic sensor motion control for distributed estimation. In *ASME International Mechanical Engineering Congress and Exposition*, November 2007.

[107] Rolf Rysdyk. UAV path following for constant line-of-sight. In *AIAA Conference on Guidance, Navigation, and Control*, 2003.

[108] Jeffrey Saunders and Randal W. Beard. Reactive vision based obstacle avoidance with camera field of view constraints. In *AIAA Conference on Guidance, Navigation, and Control*, 2008.

[109] Andrey V. Savkin and Hamid Teimoori. Bearings-only guidance of a unicycle-like vehicle following a moving target with a smaller minimum turning radius. *IEEE Transactions on Automatic Control*, 55(10):2390–2395, 2010.

[110] Ketan Savla, Emilio Frazzoli, and Francesco Bullo. Traveling salesperson problems for the dubins vehicle. *IEEE Transactions on Automatic Control*, 53(6):1378–1391, 2008.

[111] Fred C. Schweppe. Recursive state estimation: unknown but bounded errors and system inputs. In IEEE *Transactions on Automatic Control*, volume AC(13)-1, pages 22–28, 1968.

[112] David W. Scott and Stephan R. Sain. *Multi-Dimensional Density Estimation*, pages 229–263. Elsevier, Amsterdam, 2004.

[113] Eduard Semsch, Michal Jakob, Dusan Pavlicek, Michal Pechoucek, and David Sislak. Autonomous uav surveillance in complex urban environments. In *IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technologies (WI-IAT)*, volume 2, pages 82–85, 2009.

[114] Dong Gyu Sim, Rae Hong Park, Rin Chul Kim, Sang Uk Lee, and Ihn Cheol Kim. Integrated position estimation using aerial image sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):1–18, 2002.

[115] Per Skoglar. A planning algorithm of a gimballed EO/IR sensor for multi target tracking. Technical Report LiTH-ISY-R-2887, Linköping University, Linköping, Sweden, March 2009.

[116] Per Skoglar. *Planning Methods for Aerial Exploration and Ground Target Tracking*. Licentiate thesis, Linköping University, 2009.

[117] Stephen Spry, Andy Vaughn, Xiao Xiao, and J. Karl Hedrick. A vehicle following methodology for UAV formations. In *International Conference on Cooperative Control and Optimization*, 2003.

[118] Anthony Stentz, Alonzo Kelly, Herman Herman, Peter Rander, Omead Amidi, and Robert Mandelbaum. Integrated air/ground vehicle system for semi-autonomous off-road navigation. In *AUVSI Unmanned Systems Symposium*, July 2002.

[119] Lawrence D. Stone. What's happened in search theory since the 1975 Lanchester prize? *Operations Research*, 37(3):501–506, June 1989.

[120] Ethan Stump, Ben Grocholsky, and Vijay Kumar. Extensive representations and algorithms for nonlinear filtering and estimation. In *Workshop on Algorithmic Foundations in Robotics (WAFR)*, 2006.

[121] P.B. Sujit, Derek Kingston, and Randy Beard. Cooperative forest fire monitoring using multiple UAVs. In *Conference on Decision and Control*, December 2007.

[122] Panagiotis Theodorakopoulos and Simon Lacroix. A strategy for tracking a ground target with a UAV. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[123] Harold Thimbleby. The directed chinese postman problem. *Software: Practice and Experience*, 33(11):1081–1096, 2003.

[124] John Tisdale, Hugh Durrant-Whyte, and J. Karl Hedrick. Path planning for cooperative sensing using unmanned vehicles. In *ASME International Mechanical Engineering Congress and Exposition*, November 2007.

[125] John Tisdale, Zu Kim, and J. Karl Hedrick. An autonomous system for cooperative search and localization using unmanned vehicles. In *AIAA Conference on Guidance, Navigation, and Control*, 2008.

[126] John Tisdale, Allison Ryan, Zu Kim, David Törnqvist, and J. Karl Hedrick. A multiple UAV system for vision-based search and localization. In *American Control Conference*, 2008.

[127] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. Technical Report CMU-CS-91-132, Carnegie Mellon Robotics Institute, Pittsburgh, PA, 1991.

[128] Benjamin Tovar and Steven M. LaValle. Visibility-based pursuit-evasion with bounded speed. *International Journal of Robotics Research*, 27(11-12):1350–1360, November/December 2008.

[129] Emanuele Trucco and Alessandro Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall, 2002.

[130] Martin Ulmke and Wolfgang Koch. Road-map assisted ground moving target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, 42(4):1264–1274, October 2006.

[131] United Nations Population Fund. UN state of the world population. `http://web.unfpa.org/swp/2007/english/chapter_1/urbanization.html`, 2007.

[132] United States Air Force. RQ-4 Global Hawk printable fact sheet. `http://www.af.mil/information/factsheets/factsheet.asp?id=13225`, 2009.

[133] Rene Vidal, Omid Shakernia, H. Jin Kim, David Hyunchul Shim, and Shankar Sastry. Probabilistic pursuit-evasion games: Theory, implementation and experimental evaluation. *IEEE Transactions on Robotics and Automation*, 18(5):662–669, 2002.

[134] Patrick Vincent and Izhak Rubin. A framework and analysis for cooperative search using UAV swarms. In *ACM Symposium on Applied Computing*, 2004.

[135] Yoko Watanabe, Patrick Fabiani, and Guy Le Besnerais. Air-to-ground target tracking in a gps-denied environment using optical flow estimation. In *AIAA Conference on Guidance, Navigation, and Control*, 2009.

[136] Richard A. Wise and Rolf T. Rysdyk. UAV coordination for autonomous target tracking. In *AIAA Conference on Guidance, Navigation, and Control*, 2006.

[137] Hulya Yalcin, Robert Collins, Michael Black, and Martial Hebert. A flow-based approach to vehicle detection and background mosaicking in airborne video. Technical Report CMU-RI-TR-05-11, Carnegie Mellon Robotics Institute, Pittsburgh, PA, March 2005.