See but Don't Be Seen: Towards Stealthy Active Search in Heterogeneous Multi-Robot Systems

Nikhil Angad Bakshi

CMU-RI-TR-22-69

December 21, 2022

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Jeff Schneider, *Chair*
Zachary Manchester
Michael Kaess
Rishi Veerapaneni

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

For all the Gods out there and my mum, Bina, who, all on her own, has made me who I am.

Abstract

Robotic solutions for quick disaster response are essential to ensure minimal loss of life, especially when the search area is too dangerous or too vast for human rescuers. We model this problem as an asynchronous multi-agent active-search task where each robot aims to efficiently seek an unknown number of sparsely located targets in an unknown environment. This formulation addresses the requirement that search missions should focus on quick recovery of targets rather than full coverage of the search region. Previous approaches fail to accurately model sensing uncertainty, account for occlusions due to foliage or terrain when estimating the next optimal action, meet the requirement for heterogeneous search teams or robustness to hardware and communication failures. We present the Generalized Uncertainty-aware Thompson Sampling (GUTS) algorithm, which addresses these issues and is suitable for deployment on heterogeneous multi-robot systems for active search in large unstructured environments.

Some search problems have an additional dimension to seek targets while attempting to conceal the search agents' location from the targets. This is applicable to reconnaissance missions wherein the safety of the search agents can be compromised. Prior work usually focuses on adversarial search settings where the evaders (targets) are actively trying to evade the pursuers (search agents), however, most approaches assume unrealistic parameters such as complete knowledge, infinite travel speed, unlimited compute, and/or perfect observation models. We model the problem as a multi-objective optimisation over the potential information gain of taking an action and the risk of information leakage to an unknown number of targets with unknown locations. We present the Stealthy Topography-Aware Reconnaissance (STAR) algorithm, a multi-objective parallelized Thompson sampling-based algorithm that relies on a strong topographical prior to reason over changing visibility risk over the course of the search.

In both sub-problems defined above, we show through simulation experiments that GUTS and STAR consistently outperform existing methods in their respective problems. We conduct field tests using our multi-robot system in an unstructured environment with a search area of varying scale (0.075 km^2 - 2.6 km^2). Our system demonstrates robustness to various failure modes, achieving full recovery of targets (where feasible) in every field test.

Acknowledgments

First and foremost, I'd like to thank my mum, Bina. Her love and enduring patience for me has underpinned my life. She deserves much more than my thanks, she deserves credit that I have completed this dissertation.

I have many to thank in Auton Lab. Conor and Tejus, our discussions were particularly fruitful, even when they had nothing to do with work! Arundhati and Ramina who helped me translate Greek into English.

This project was in collaboration with National Robotics Engineering Center (NREC). The hard work and expertise of the people at NREC made running large-scale field tests with full-size robots possible. There are so many wonderful people in NREC that I'm sure I'll miss some people, but I would be remiss to not mention Herman, whose foresight and deep knowledge led us to successful tests; the autonomy team, Luis, Prasanna and Jacob, who took extra time out of their already busy schedules to indulge the musings of a grad student; the software team, Vamsi, Dave and Felix, who saved me the trouble of many a debugging session; the drone team, Matt, Wennie and Kris; and several others like Jesse, Trenton and Jon.

I want to thank all my comrades, you made my time at CMU truly memorable and will continue to do so in the years to come. Suffice it to say, I got by with a little help from my friends.

Thanks to my committee members Rishi, Zac and Michael who were instrumental in moulding this dissertation and its defence into its final form.

It goes without saying that my advisor, Jeff, played a pivotal role in guiding me and shaping this body of work. It was a childhood dream of mine to work on field robotics and I will be forever grateful to Jeff for providing me the opportunity make a meaningful contribution to this domain.

Funding

This material is based upon work supported by the U.S. Army Research Office and the U.S. Army Futures Command under Contract No. W911NF-20-D-0002.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Search for Disaster Response

The frequency of disaster events is rapidly increasing, world over, due to climate change [20, 69]. Usually, disaster response operations such as search and rescue (SAR) are a human endeavour however several factors can impede effective search by human teams. The search region could be too vast to mobilise enough human resources to effectively search as is the case during wildfires, floods, landslides, or avalanches (See Fig. 1.1 top). The search region could also pose a safety risk for searchers as in the case of nuclear containment or earthquakes (See Fig. 1.1 bottom-left). Over the years, the deployment of robots for disaster response has become more common [6, 15, 47, 48, 49] however teleoperation of robots is the most common use. While this reduces risk to the human search teams, it does not scale as coordination between agents and communication of information gleaned over the course of the search is rate limited by the human operators [25] (See Fig. 1.1 bottom-right). Emergency operations are also time-critical, hence multi-robot teams with high search efficiency are crucial. This motivates the requirement for decentralized multi-robot systems capable of efficient active search in large unstructured terrestrial environments.

Active search is the problem of making efficient sequential data-collection decisions to identify sparsely located targets, while adapting to new sensing information [21, 22]. The basic framework for active search consists of maintaining a posterior probability distribution over locations of targets and optimizing for information-seeking actions.

Figure 1.1: Top-left: Wildfire in a massive forested region (Image Source: NASA). Top-right: Massive flood scenario in Pakistan (Image Source: Fida Hussain/AFP). Bottom-left: Fukushima nuclear reactor meltdown (Image source: Telangana Today). Bottom-right: Representative picture of an intractable number of tele-operated drones in the air that human operators would not be able to effectively coordinate (Image Source: Andy Dean).

These actions must balance surveying unseen areas (exploration) and confirming suspected targets (exploitation) [80, 81, 85].

Even though this basic framework is straightforward, several additional challenges need to be addressed for deploying these algorithms on a multi-robot system in realistic search and rescue scenarios. Targets may be occluded to overhead or lateral views (See Fig. 1.2); hence, a combination of ground and aerial search vehicles is essential to canvass such a search region thoroughly. However, this requires the search algorithm to reason through occlusions, viewshed constraints, as well as each robot's sensing model and navigational constraints to plan their search actions. Similarly, though multi-robot teams can theoretically partition the search region for efficient exploration [62], generating such a partitioning is challenging in a decentralized system

Figure 1.2: Testing area for our search system in Pittsburgh, PA.

as robots may fall in and out of communication or may get disabled over the course of the search. Though we would like the robots to share as much sensing information as possible for effective coordination, we cannot rely on persistent communication. Therefore, we require our system to be asynchronous and be able to make intelligent decisions with partial information. Finally, the number of targets in the search region is usually unknown, and sensing actions are noisy [32, 43, 53]. Previous search methods make strong simplifying assumptions [19, 60] and don't deal with these challenges fully, making it intractable to deploy them in realistic search settings. We propose a novel search algorithm, Generalized Uncertainty-aware Thompson Sampling (GUTS) that follows the parallelized Thompson Sampling framework [21, 22, 35]. GUTS has an improved reward function that prioritizes the rate of recovery of targets, realistically models sensing noise in UAVs and UGVs, and is computationally optimized it to run in large environments on real hardware. GUTS is applicable to any unstructured search region, without assumptions on the number of targets, and is robust to communication and hardware failures.

We demonstrate our search system consisting of UGVs and UAVs on field tests in a large unstructured natural environment in Pittsburgh, PA, shown in Fig. 1.3 and on

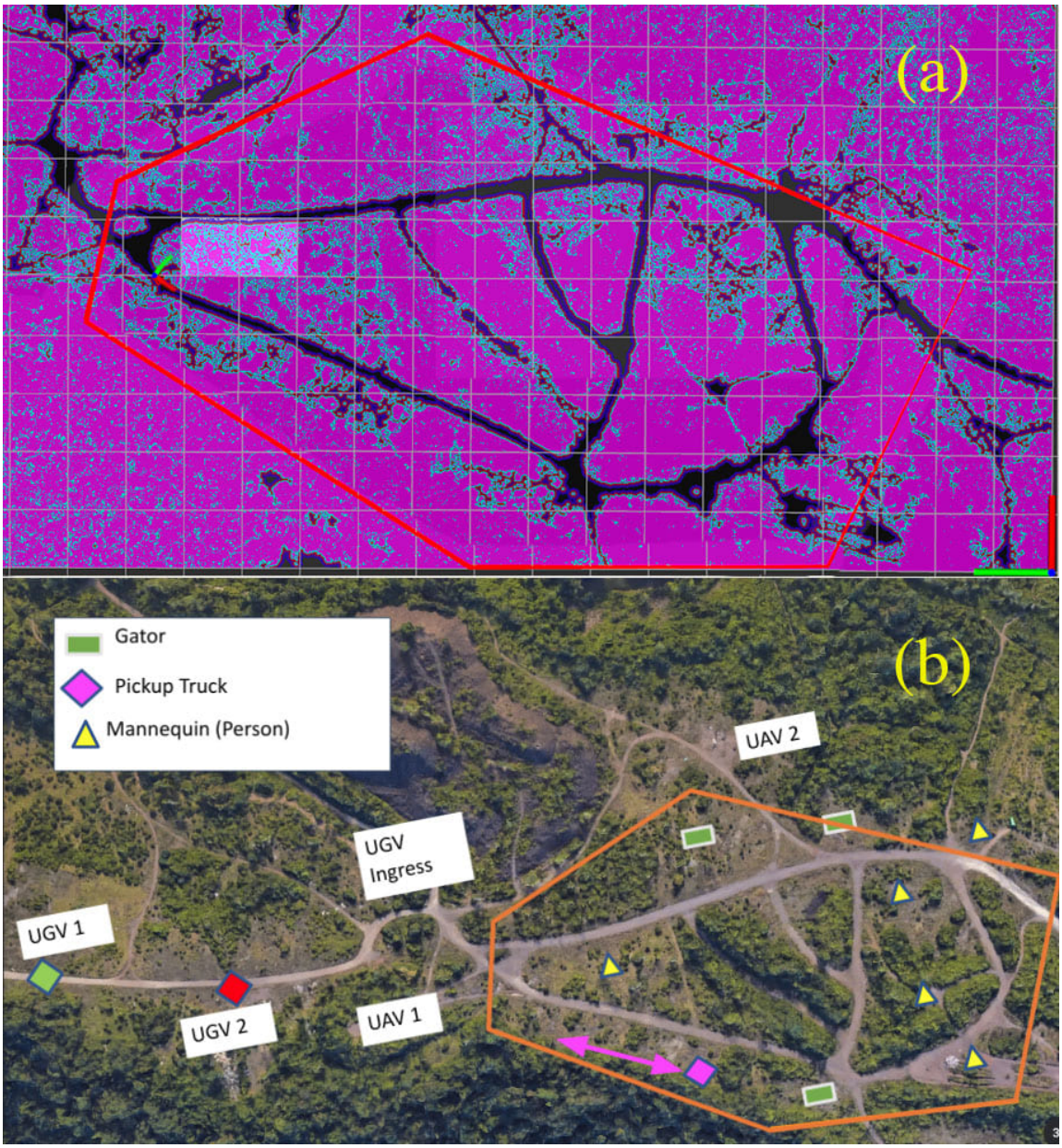


Figure 1.3: Test-field in Pittsburgh, PA. (a) shows the traversal costmap and the search polygon (purple is high cost and black is low cost), and (b) shows an overhead image with the most common target locations and launching areas of the robots. The area inside the polygon is roughly 75,000 sq meters.

simulated tests. Our results show the robustness of our system, along with superior search performance compared to existing approaches. To the best of our knowledge, this is the first heterogeneous multi-robot system for asynchronous multi-agent active

Figure 1.4: Left: Military vehicles with military personnel whose safety is at risk during reconnaissance missions (Image Source: Reuters). Right: Desert mountainous landscape, the topography can inform search and evasion strategies (Image Source: Ed Ledford).

search built at this scale that is robust to communication and robot failures and operates without any human direction or manual subdivision of the search region.

1.2 Search for Reconnaissance

In reconnaissance missions, a team of agents search for an unknown number of hostile targets in a given region. In contrast with search for disaster response, the targets in reconnaissance are adversarial in nature: they do not wish to be found and search agents attempt to conceal their own locations from the targets. As pictured in Fig. 1.4-right we consider landscapes with an undulating terrain devoid of foliage.

Similar to disaster response solutions, as it stands today, reconnaissance missions are a largely human endeavour, time-critical, pose great risk for the human search agents, and teleoperation of robots is not a scalable solution.

In prior work, the problem of adversarial search is often referred to as the pursuer-evader problem [1, 50]. Several approaches seek to maximise the worst-case performance of the pursuer and imbibe the evader with extraordinary abilities like complete knowledge, infinite travel speed, and infinite compute. However, these solutions are often prohibitively expensive to compute [23] on real hardware or too conservative to be applicable in real-world settings even when using more realistic constraints like limited visibility for the pursuer and evader [31].

Target tracking [83, 84] and target detection [11] are distinct problem formulations

in the adversarial setting where, in the former case, the targets move in an adversarial or non-adversarial way and the team's role is to maintain a strong belief over the location of the target over time, whereas in the latter case, the team's role is to identify its location with high certainty as quickly as possible. In the general case, the former problem is NP-complete [9], however for the resource-constrained single-pursuer case an optimal solution exists taking into account visibility risk [61]. We model the problem as the latter case, with multiple pursuers and immobile evaders that may be placed on the map adversarially. Our metrics are the rate of recovery of the sparsely located targets and the stealth penalty incurred by the team of search agents over the course of the search.

The core idea of our approach is simple, during reconnaissance missions a strong prior on the number of targets or their locations is usually not available, however, satellite imagery and by extension topographical information of the search region may be available. Refer to Fig. 1.5 for an example average visibility of a terrain similar to that pictured in Fig. 1.4, if the agent were to search for sparsely located targets in the map, the intuition is that it would be beneficial to search in better hiding places as the adversarial targets don't want to be found. Additionally, the topography also informs the path planning as taking routes with lower visibility risk should be preferred.

Topography-aware path planning with adversarial targets is well-studied in the context of military operations [46, 67, 68] and in the context of stealth-based video games [2, 30]. However, these approaches focus on path planning but not on a competing search objective, i.e. they assume that the adversary locations are known or are unknown but don't need to be located, they simply need to be avoided en route to the goal location if encountered.

We present the Stealthy Topography-Aware Reconnaissance (STAR) algorithm, a multi-objective optimisation algorithm that follows the parallelized Thompson Sampling framework [21, 22, 34] which has been shown to be near optimal for adaptive goal-oriented tasks [35] such as in the multi-agent asynchronous active search problem we have here. We combine the information-seeking reward-based policy from GUTS with a topography-aware stealth penalty that informs search and path planning using the OMPL planner [66]. We show through our simulation experiments that the STAR algorithm outperforms GUTS, an information-greedy

region sensing policy [44], a myopic coverage-based policy, and a random policy on the recovery rate of targets as well as on the stealth metric we define. Finally, we verify that the STAR algorithm runs on upto four ground-based search agents in a search region that is $2.5km^2$ in size, though the field tests were conducted at a restricted site and we do not present the results here.

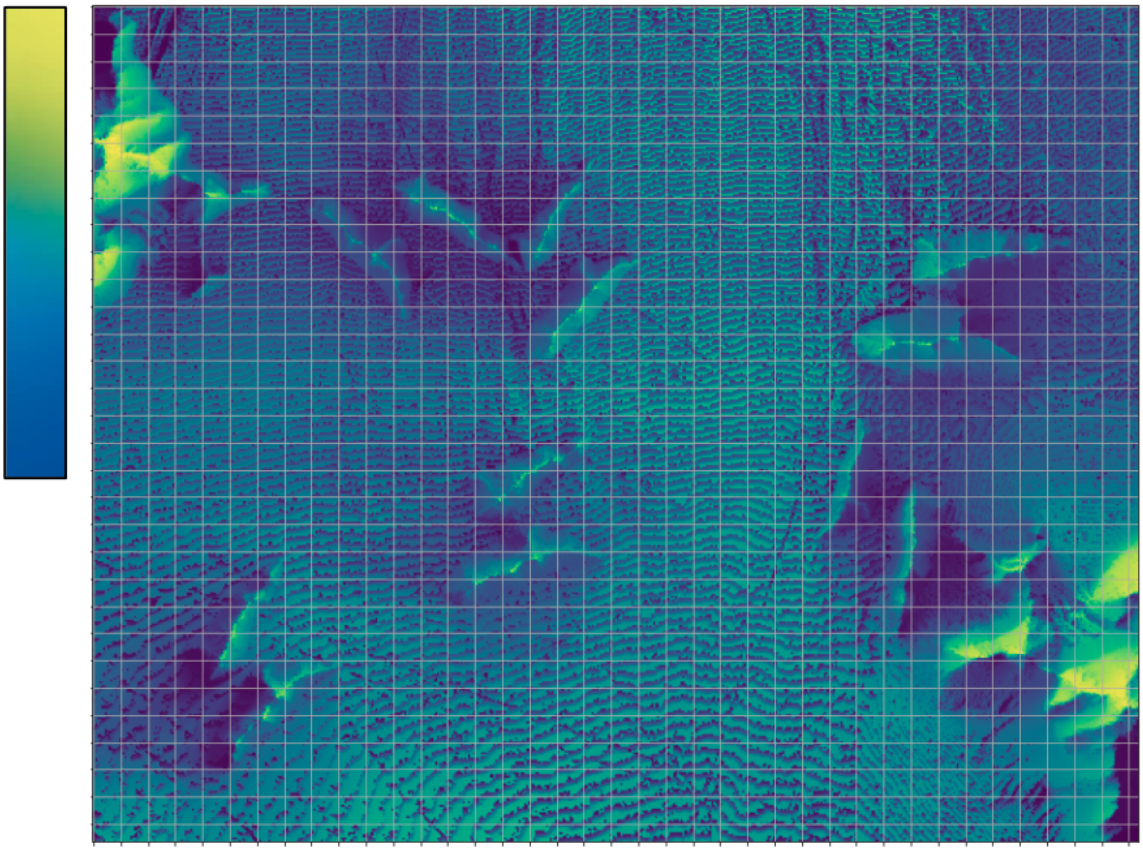


Figure 1.5: Top-down view of an example terrain showing average visibility as a colour gradient from deep blue to bright yellow. As we can see the peaks of the mountains are visible from anywhere in the map and are bright yellow while the valleys between closely spaced mountains are less visible and hence deep blue. The grid size here is $60m \times 60m$. This region's dimensions are $2305.5m \times 1837.5m$ representing a total area of $4.236km^2$. That's nearly 750 American Football fields. As is evident there are plenty of locations that the targets can be hidden and simultaneously plenty of opportunity for the search agent(s) to search while remaining concealed.

1.3 Research Contributions

- We propose the Generalised Uncertainty-aware Thompson Sampling (GUTS) algorithm, that uses an improved reward design, more accurately models sensing noise, and is computationally optimized over prior parallelized TS methods [21, 22] and optimised for quick recovery during search and rescue missions.
- We experimentally show superior search success rate and search efficiency of GUTS compared to existing search methods, namely NATS [21] and coverage-based search, in field and simulated experiments.
- We propose the Stealthy Topography Aware Reconnaissance (STAR) algorithm, a multi-objective search algorithm that combines with reward function of GUTS with a stealth objective that uses topographical information to encourage concealment of the search agent as well as search in locations with greater likelihood of recovery.
- We show good search performance of the STAR algorithm, locating all targets within the time budget while performing better than GUTS and other baseline methods on both the stealth metric and rate of recovery of targets in simulation. We validate the STAR algorithm on real robots.
- We present a working system of heterogeneous robots that is capable of asynchronous multi-agent active search in large unstructured natural environments without a central planner. We design our multi-agent search algorithm to be robust to loss of robots mid-search, noisy observations and communication breakdown without any human intervention.

1.4 Thesis Outline

In Sec. 2 we cover key concepts that constitute background knowledge in this thesis and will underpin the rest of the work. In Sec. 3 we describe an overview of the physical test systems employed in this work. In Sec. 4 we describe the prior work in the area of multi-agent active search and present the GUTS algorithm that tackles the shortcomings of prior approaches. We show superior results in simulation and in field experiments. In Sec. 5, we define the problem of stealthy multi-agent active

search before we go on present the STAR algorithm that shows promising results by outperforming existing methods on a realistic formulation of the problem that is not tackled in literature.

Chapter 2

Background

2.1 Active Search

Active search refers to the problem of finding a target by making a sequence of queries, each of which reveals some information about the target's location. Active search problems can be formulated in a variety of ways, depending on the specific context and the constraints on the searcher's actions. For example, in a multi-armed bandit problem, the searcher is trying to locate a target that is hidden in one of a number of 'arms,' each of which has a different probability of yielding a reward. The searcher must balance the trade-off between exploration (trying out different arms to gather more information) and exploitation (choosing the arm that has the highest expected reward based on the information gathered so far).

In the multi-agent search context, active search is the problem of making efficient sequential data-collection decisions to identify sparsely located targets, while adapting to new sensing information [21, 22]. The basic framework for active search consists of maintaining a posterior probability distribution over locations of targets and optimizing for information-seeking actions. These actions must balance surveying unseen areas (exploration) and confirming suspected targets (exploitation) [80, 81, 85].

Information gain is commonly used as a metric to inform search actions. Information gain refers to the amount of additional information that is obtained by making a particular query. It can be measured using Shannon's entropy [63], which is a measure of the uncertainty or randomness in a system. The entropy of the target's location is

high initially, and it decreases as the search agent receives more information through the queries. The information gain of a particular query is the difference in the entropy of the target's location before and after the query is made. Information gain can be formalised as follows:

$$IG = \mathcal{H}(p_t^{pursuer}) - \mathcal{H}(p_{t+1}^{pursuer})$$

where, $p_t^{pursuer}$ is the belief (probability mass function) over the location of the **single** evader at timestep t with respect the the **single** pursuer. $H(p_t^{pursuer})$ is the Shannon entropy of this probability mass function at timestep t. Hence the information gain measures the reduction in entropy (and therefore the increase in certainty) in the belief of the location of the evader with respect to the pursuer between timesteps. The assumption is that the pursuer makes a query at every timestep causing its belief $p^{pursuer}$ to change.

In adversarial active search, the search agent has a competing objective to limit Information Leakage to the targets being searched for. Information leakage refers to the amount of information that is revealed about the search agent's location to the adversarial targets. Information leakage can occur when the search agent makes queries to locate the targets.

Information leakage can be formalised as follows:

$$IL = \mathcal{H}(p_t^{evader}) - \mathcal{H}(p_{t+1}^{evader})$$

where, p_t^{evader} is the belief (probability mass function) over the location of the **single** pursuer at timestep t with respect the the **single** evader. $H(p_t^{evader})$ is the Shannon entropy of this probability mass function at timestep t. Hence the information leakage measures the reduction in entropy (and therefore the increase in certainty) in the belief of the location of the pursuer with respect to the evader between timesteps. The assumption is that the pursuer makes a query at every timestep causing it to leak some information about its own location to the evader and hence the belief p^{evader} changes.

Then in our formulation of a **single** pursuer and evader the optimal policy would be the result of:

$$\pi_*^{\text{pursuer}} = \arg\max_{\pi^{\text{pursuer}}} \mathbb{E}\left[\overbrace{\sum_t \gamma^t \big(\mathcal{H}(p_t^{\text{pursuer}}) - \mathcal{H}(p_{t+1}^{\text{pursuer}})\big)}^{\gamma \text{ controls the horizon}}; \pi^{\text{pursuer}} \right]$$

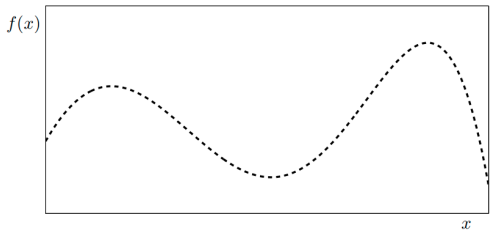
$$s.t. \quad H(\pi^{\text{pursuer}}) = \mathbb{E}\left[\sum_t \gamma^t \big(\mathcal{H}(p_t^{\text{evader}}) - \mathcal{H}(p_{t+1}^{\text{evader}})\big); \pi^{\text{pursuer}}, \pi^{\text{evader}} = f(\pi^{\text{pursuer}}) \right] < \bar{\mathcal{H}}$$

where, π_*^{pursuer} is the optimal policy of the pursuer that maximises the expected information gain over the entire episode with γ controlling the extent of the receding horizon. The second equation is a constraint to limit the information leakage to the evader below some threshold $\bar{\mathcal{H}}$ over the entire episode. π^{evader} is the policy of the evader which is a function of the pursuer policy π^{pursuer}, this makes sense in our case as the evader is immobile and it can only make an observation of the pursuer given the action of the pursuer, for example if the pursuer chooses to come close enough to it that the evader can detect it.

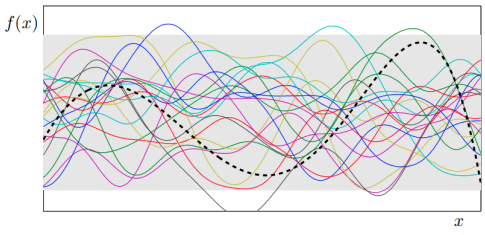
2.2 Thompson Sampling

The emphasis on mentioning that there is only a single evader and single pursuer in the above formulation is that, usually, Bayesian search or information greedy approaches to active search are inapplicable to multi-agent search situations without a central planner [32, 43, 53]. In order to overcome this limitation of needing a central planner for multiple agents we explore Thompson Sampling.

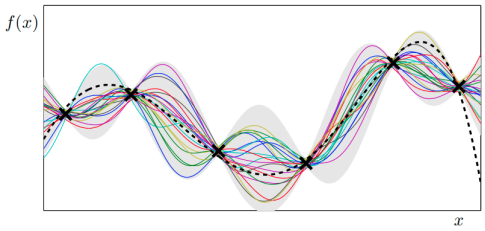
Thompson Sampling (also known as Myopic Posterior Sampling) is an online Bayesian learning algorithm that is used to make decisions in a sequential manner. Thompson Sampling provides a framework to solve problems which involve balancing the trade-off between exploration (trying out different options to gather more information) and exploitation (choosing the option that has the highest expected reward based on the information gathered so far) [57]. In Thompson Sampling, the decision-maker maintains a probability distribution (called the posterior distribution) over the expected rewards of each option at each step. At each step, the decision-maker samples a reward from the posterior distribution for each option and selects



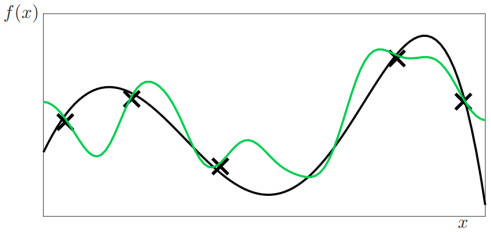
(a) Suppose we want to determine the maximum of some unknown function (dotted line). We only have access to a black box that when queried with an input x returns an observation y.



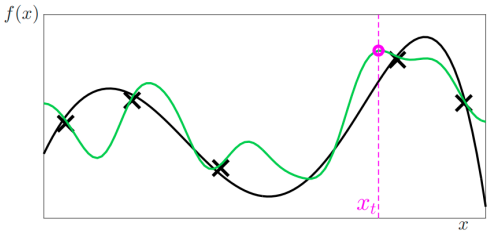
(b) Our prior over the function is extremely uncertain (coloured lines).



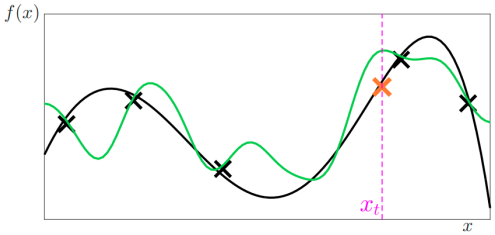
(c) As we get more data ($\{x, y\}$ pairs) we refine our posterior belief (coloured lines) and bring it closer to reality.



(d) Thompson Sampling provides us with a sampling paradigm that is near optimal: we start by taking a sample from the posterior (green line) and assume that is the true function.



(e) We find the maximum of our Thompson Sample (green line) at x_t and query the black box at that value.



(f) The red cross marks the observation y_t returned by the black box when queried with x_t.

Figure 2.1: Illustrative example of Thompson Sampling in 1D. In this example we attempt to find the maximum of an unknown function, in the process we refine our estimate of the actual underlying function. Steps (d), (e) and (f) are repeated and over time the we expect to converge to the optimal decision-making strategy [35]. Diagram credit: Kirthevasan Kandasamy

the option with the highest sampled reward. This process is repeated at each step until the problem is solved. Fig. 2.1 illustrates the working of Thomspon Sampling for a 1-D continuous example.

Thompson Sampling has several attractive properties, including convergence to the optimal decision-making strategy [35] as the number of steps increases and robustness to misspecification of the prior distribution [34]. It has been applied to a wide range of problems, including online advertising, clinical trials, and recommendation systems.

Thompson Sampling is an excellent candidate for an asynchronous multi-agent online algorithm without a central planner. By using a posterior sample in its reward function, it allows a calculated randomness in the reward that enables multiple agents to independently solve for different values that all contribute to the overall goal of locating targets even if all agents have the exact same information but are not able to centrally plan.

2.3 Sparse Bayesian Learning

Thompson Sampling allows us to scale to multiple agents without a central planner whilst being robust to communication loss or robots becoming disabled mid-run, however, when the number of targets is unknown, there is no closed form of equations that can describe the optimal policy in terms of the information gain and information leakage as both quantities need to be integrated over a meta-belief over how many targets are actually out there.

Before we examine the solution to having an unknown number of targets, let us consider that we had k targets, the approach here would be to have k distributions over the space for each of the targets and then information gain and information leakage could be summed for each agents over all targets when doing the Thompson Sampling step.

We follow [21] and utilize Sparse Bayesian Learning as explained in Tipping [71] to deal with an uncertain k. The idea of Sparse Bayesian Learning is simple, instead of k being known apriori it is learned through observations. We assume a separate gaussian distribution for each location in the discretised space. Each of these distributions has a zero-mean prior while the variance is sampled from an inverse-gamma distribution (See Fig. 2.2). As is evident from the figure, the inverse-gamma distribution will

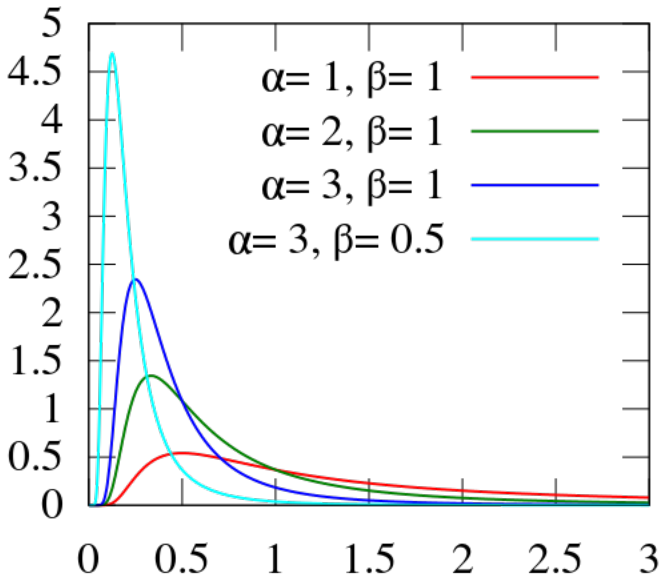


Figure 2.2: Representative plot of inverse-gamma distributions with various parameters

encourage only a few variances values to be significantly greater than zero while most will be close to zero. The parameters of the inverse-gamma distribution will be learned through incoming observations and hence this encodes some flexibility in the number of targets being modelled k.

Now we have all the building blocks in place that we may construct a robust multi-agent active search system [21] capable of searching for an unknown number of sparsely located targets [51, 71] without strong communication/coordination guarantees [34] whose search performance is expected to be near optimal [35].

Chapter 3

System Description

This section describes our multi-robot system on which we evaluate our search algorithm. For the UGV, we use the RecBot, a John Deere E-Gator utility vehicle that has been retro-fitted with a drive-by-wire autonomy kit (Fig. 3.1). For the UAV, we use the Lil Hexy [1] air-frame paired with a PixHawk flight controller [2] as our base platform (Fig. 3.1). We added a custom camera payload and additional onboard compute in the form of the Nvidia Jetson AGX Xavier to this platform.

We summarize the components of our autonomy stack necessary to understand the work in this dissertation below.

3.1 Camera FOV

As depicted in Fig. 3.2 the sensor pod on the UGV has a 37° vertical and 192° horizontal FOV. The sensor module consists of five RGB cameras, each of 12 MP, to allow for accurate target detection up to several hundred meters away. A sensor pod with one RGB camera of 5 MP resolution is mounted on the UAV pointed downward with a 5° angle with the vertical, tilted in the direction of the travel of the UAV. The 19°× 16° FOV on this camera would, for instance, cover an area of $27.8m \times 22.5m$ from a flight height of 80m.

[1] https://vscl.tamu.edu/vehicles/little-hexy/
[2] https://pixhawk.org/

(a) (b)

Figure 3.1: Left: RecBot UGV. Right: Lil Hexy UAV.

3.2 Object Detector

For the UGV, we use a customized version of the widely used YOLOv3 neural network architecture [54] to support fast inference on the Nvidia RTX 6000 GPU given very high-resolution input images. For the UAV detector, the computing power is limited; so we use a less computationally intensive model [36] and run it on the Nvidia Jetson AGX Xavier.

3.3 Object Tracking

The perception module uses a Kalman filter [33] to track possible targets. We describe our sensing model in detail in section 4.2 and 5.3. For example, the estimated target locations detected by the ground-vehicles have more uncertainty along the range to the object compared to the bearing direction. This is represented by the covariance associated with a track returned by the Kalman filter. Moreover, this uncertainty increases with object distance from the observer.

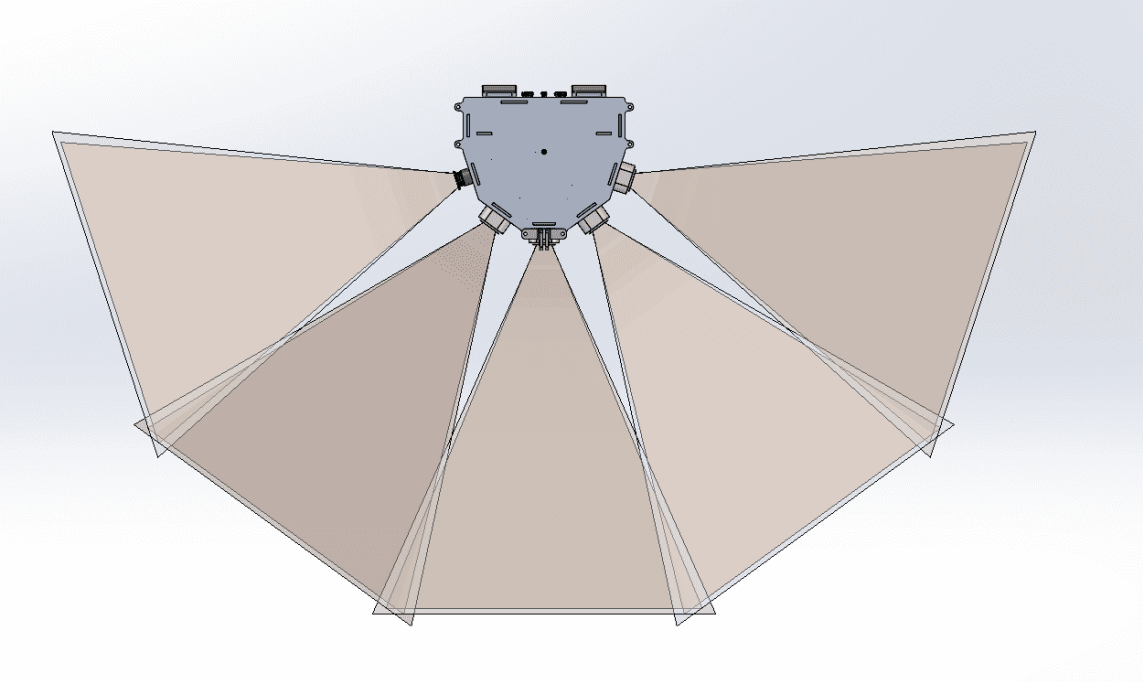


Figure 3.2: Top-down view of UGV camera field-of-view. Span is 192° horizontally and 37° vertically.

3.4 Traversal Costmap for UGV

We represent the environment using a global and local costmap. The global costmap is initialized from prior information based on overhead imagery. The local costmap is generated from sensor data as the robot traverses the environment. These costmaps are used extensively by the low-level planner, Search Based Planning Library (SBPL)[3] or Open Motion Planning Library (OMPL) [66], to chart safe paths to waypoints provided by the planning module being presented in this paper.

3.5 Targets

The targets are people (mannequins), a John Deere E-Gator, or a pickup truck. Fig. 3.4 shows example frames from the UGV and UAV camera systems with successful detections.

[3]https://github.com/sbpl/sbpl

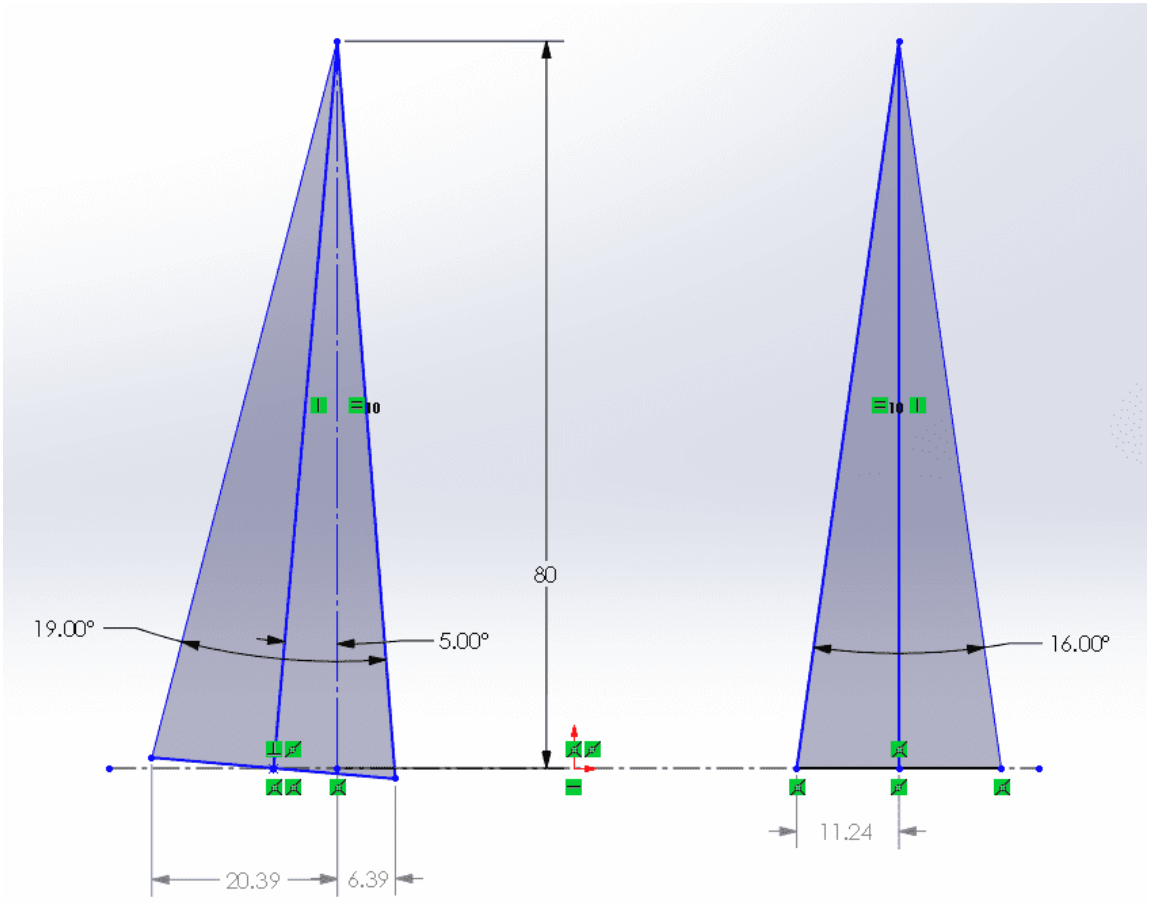
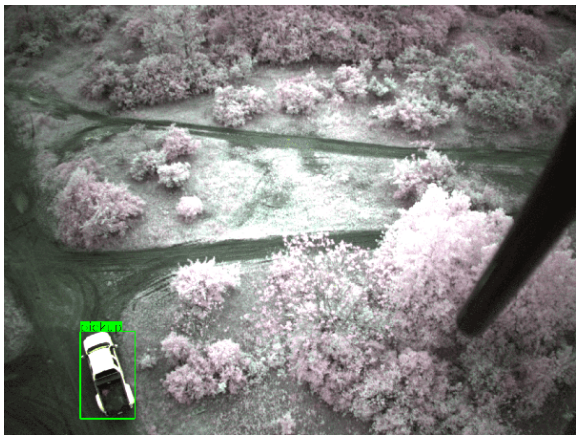


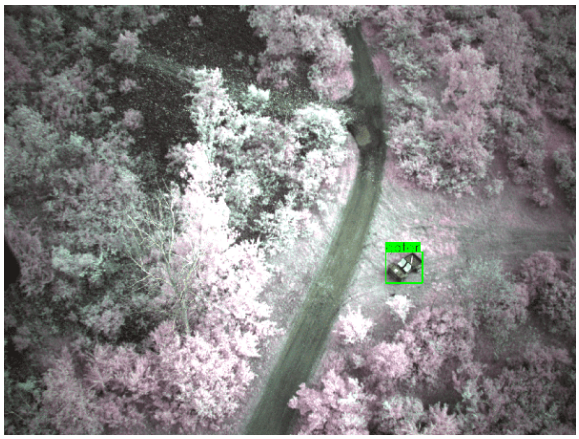
Figure 3.3: UAV camera field-of-view. Span is 19° × 16°. At a flight height of 80m, this leads to approximately $27.8m \times 22.5m$ rectangle of visibility on the ground.

(a) Pickup Truck



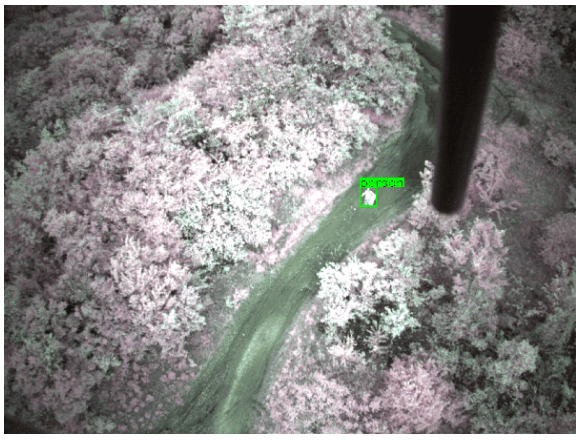
(b) Pickup Truck

(c) E-Gator



(d) E-Gator

(e) Person



(f) Person

Figure 3.4: Object detector: qualitative results. We show images and target detections for the UGV (left) and the UAV (right)

Chapter 4

GUTS: Generalized Uncertainty-Aware Thompson Sampling

In this section, we will formalise the decentralised multi-agent active search problem and present the GUTS algorithm.

4.1 Related Work

Previous works on multi-robot search systems work in more specialized settings (e.g., homogeneous teams, single agent search) and make strong assumptions (perfect communication, homoscedastic noise model). For instance, Coverage-based search methods [13, 42, 58, 59] exhaustively navigate the search space. They assume that the search area is mapped and do not adapt to new information online. A similar criticism exists for frontier based methods like [52].

Bayesian active-search methods [32, 43, 53] can accurately model observation uncertainty when searching for sparse signals with a single agent. However, the deterministic nature of their search policies makes extending these methods to the multi-agent setting difficult without centralization. Recent work has modelled the active-search problem as a POMDP and trained search policies in simulation using reinforcement learning [29, 64, 80, 81, 85]. It is usually difficult to deploy these

methods on robotic systems because of their high sample complexity and unrealistic simulations.

Ghassemi et al. [19] employ an information-theoretic approach to asynchronous multi-agent active search, but work with a homogenous swarm assuming perfect communication and a homoscedastic noise model. Salman et al. [60] generate trajectories such that the time spent in an area corresponds to the likelihood of the presence of targets. However, their method does not incorporate new observations and relies on perfect communication and a strong unified prior on locations of targets. Ayvali et. al. [4] tries to improve upon the previous approach without relying on a prior but is confined to a single agent case.

Dec-MCTS [8] is a multi-agent active-perception algorithm that has shown promising simulated results. Dec-MCTS is a decentralized version of MCTS where all robots optimize their trajectories asynchronously but achieve coordination by sharing their partial search trees and do not model uncertainties in realistic perception pipelines.

Noise-Aware Thompson Sampling (NATS) [21] is an asynchronous multi-agent active-search algorithm that can deal with heteroscedastic noise models and has shown promising simulated results. NATS uses myopic posterior sampling to ensure that the agents take diverse sensing actions without centralization. Thompson Sampling [70] is an online optimization method that balances exploration and exploitation by maximizing the expected reward assuming that a sample from the posterior is the true state of the world [57]. This stochastic nature is the key attribute that makes it an excellent candidate for an asynchronous multi-agent setting with unreliable communication as it promotes some diversity in action selection.

4.2 Problem Formulation

We model the search region as a grid with a cell size of 30m x 30m. We have access to the costmap for ground robot traversability. Fig 1.3 shows an overhead view and the costmap for the field testing site in Pittsburgh, PA. This costmap was generated from a drone flyover several months before field testing. Some parts of the map have changed since then, but our on-robot sensing and mapping system can recognize changes like blocked paths. Due to the ubiquity of satellite imagery, it is reasonable to assume such approximate map information of the search region to be available.

As is evident from Fig. 1.3, several areas cannot be accessed or viewed by ground vehicles. In addition, large portions of the map are covered with dense foliage, which may obscure any targets hidden under it from the view of the air vehicles. This necessitates heterogenous coordination to explore the search region completely.

We model the active search problem as follows:

- We give the same search region to all robots. (see the red polygon in Fig. 1.3 for an example).
- The targets are sparsely placed and static. They need to be located as fast as possible with high certainty.
- Each robot must plan its next data collection action on-board, i.e., no central planner exists.
- The robots may communicate their locations and observations with each other; however, the algorithm should function anywhere in the spectrum of total absence of communication to perfect communication.

We note that the objective is *not to cover* the entire search space. Rather it is specifically to locate all targets as fast as possible.

More formally, we represent the locations of targets in our 2D grid representation using a sparse matrix $\mathbf{B} \in \mathbb{R}^{M_1 \times M_2}$. Let $\boldsymbol{\beta} \in \mathbb{R}^M$ be the flattened version of matrix $\mathbf{B}$, where $M = M_1 M_2$. This vector is sparsely populated, with 1 at locations corresponding to targets and 0 elsewhere. We model the uncertainty in our observations by

$$\mathbf{y}_t = clip(\mathbf{X}_t \boldsymbol{\beta} \pm \mathbf{n}_t, 0, 1) \text{ with } \mathbf{n}_t \sim \mathcal{N}^+(0, \Sigma_t) \tag{4.1}$$

where $\mathbf{X}_t \in \mathbb{R}^{Q \times M}$ describes the sensing matrix such that each row in $\mathbf{X}_t$ is a one-hot vector indicating one of the grid cells in view of the robot at timestep t, and Q is the total number of grid cells the robot can view. $\mathbf{y}_t \in \mathbb{R}^{Q \times 1}$ is the resultant observation including the additive depth-aware noise vector $\mathbf{n}_t \in \mathbb{R}^{Q \times 1}$. This depth-aware noise encodes the intuition that observation uncertainty increases with distance to the robots, and we model it by having the diagonal elements of the noise covariance matrix Σ_t monotonically increase with the distance of the observed cell from the robot. The noise is sampled from a positive half-Gaussian distribution $\mathcal{N}^+(0, \Sigma_t)$ and is added for cells without targets and subtracted for cells with targets. The observation y_t is clipped to be within 0 and 1.

Algorithm 1 GUTS Algorithm

Assume: Sensing model (4.1), sparse signal $\boldsymbol{\beta}$, J agents
Set: $\mathbf{D}_0^j \leftarrow \emptyset \ \forall j \in \{1, ..., J\}, \ \ \gamma_m = 1 \ \forall m \in \{1, ..., M\}$
for $t = 1, ..., T$ **do**
  Wait for an agent to finish; for the free agent j:
  Sample $\tilde{\boldsymbol{\beta}} \sim p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\Gamma}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{V})$ from (4.2)
  $\mathbf{X}_t = argmax_{\tilde{\mathbf{X}}} \mathcal{R}(\tilde{\boldsymbol{\beta}}, \mathbf{D}_t^j, \tilde{\mathbf{X}})$ from (4.5)
  Observe $\mathbf{y}_t$ given action $\mathbf{X}_t$
  Update $\mathbf{D}_{t+1}^j = \mathbf{D}_t^j \cup (\mathbf{X}_t, \mathbf{y}_t)$
  Share $(\mathbf{X}_t, \mathbf{y}_t)$ Estimate $\boldsymbol{\Gamma} = diag([\gamma_1, ..., \gamma_M])$ using (4.3)
end for

Let $\mathbf{D}_t^j$ be the set of observations available to robot j at timestep t. $\mathbf{D}_t^j$ comprises of $(\mathbf{X}_t, \mathbf{y}_t)$ pairs collected by robot j as well as those communicated to robot j by other robots. Let the total number of sensing actions by all agents be T. Our main objective is to sequentially optimize the next sensing action $\mathbf{X}_{t+1}$ based on $\mathbf{D}_t^j$ at each timestep t to recover the sparse signal $\boldsymbol{\beta}$ with as few measurements T as possible. Each robot optimizes this objective based on its own partial dataset $\mathbf{D}_t^j$ in a decentralized manner.

4.3 Algorithm

This section presents the GUTS algorithm. The framework for decision-making follows [21]. Each robot j asynchronously estimates the posterior distribution over target locations based on its partial dataset $\mathbf{D}_t^j$. During the action selection stage, each robot generates a sample from this posterior and optimizes a reward function for this sampled set of target locations. Sec. 4.3.1 and Sec. 5.4.2 describe both these steps for the Noise-Aware Thompson Sampling (NATS) algorithm presented in [21].

We utilize the posterior computation step in Sec. 4.3.1 for GUTS . In Sec. 4.3.3 and onward, we describe the new reward function to improve search performance, the depth-aware noise models for our UGVs and UAVs, and speed-up techniques to optimise the code to run on large environments.

4.3.1 Calculating Posterior

Each robot assumes a zero-mean gaussian prior per entry of the vector $\boldsymbol{\beta}$ s.t. $p_0(\beta_m) = \mathcal{N}(0, \gamma_m)$. The variances $\boldsymbol{\Gamma} = diag([\gamma_1 ... \gamma_M])$ are hidden variables which are estimated using data. We follow [71, 78] and use a conjugate inverse gamma prior on γ_m to enforce sparsity s.t. $p(\gamma_m) = IG(a_m, b_m) = \frac{b_m^{a_m}}{\Gamma(a_m)} \gamma_m^{(-a_m-1)} e^{-(b_m/\gamma_m)} \; \forall \; m \in \{1...M\}$. We estimate the posterior distribution on $\boldsymbol{\beta}$ given data $\mathbf{D}_t^j$ for robot j using EM.

We can write analytic expressions for the E-step (estimating $p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\Gamma}) = N(\boldsymbol{\mu}, \mathbf{V})$) and M-step (computing $\max_{\boldsymbol{\Gamma}} p(\mathbf{D}_t^j|\boldsymbol{\beta}, \boldsymbol{\Gamma})$) respectively:

$$V = (\Gamma^{-1} + X^T \Sigma X)^{-1}; \mu = V X^T \Sigma y \tag{4.2}$$

$$\gamma_m = ([\mathbf{V}]_{mm} + [\boldsymbol{\mu}]_m^2 + 2b_m)/(1 + 2a_m) \tag{4.3}$$

where $\mathbf{X}$ and $\mathbf{y}$ are created by vertically stacking all measurements $(\mathbf{X}_t, \mathbf{y}_t)$ in $\mathbf{D}_t^j$, and $\boldsymbol{\Sigma}$ is a diagonal matrix composed of their corresponding depth-aware noise variances.

Each robot estimates $N(\boldsymbol{\beta}|\mathbf{D}_t^j) = N(\boldsymbol{\mu}, \mathbf{V})$ on-board using it's partial dataset D_t^j. We drop the index j from equations (4.2) and (4.3) for clarity. We set the values of $a_m = 0.1$ and $b_m = 1$ as these were found to be effective in [21]. Finally, agent j samples from the posterior $\tilde{\boldsymbol{\beta}} \sim p(\boldsymbol{\beta}|\mathbf{D}_t^j)$.

4.3.2 Choosing Next Sensing Action

Each robot chooses the next sensing action $\mathbf{X}_{t+1}$ by assuming that the sampled set of target locations $\tilde{\boldsymbol{\beta}}$ is correct. Specifically, let $\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{X}_{t+1}, \mathbf{y}_{t+1}))$ be our expected estimate of the parameter β using all available measurements $\mathbf{D}_t^j$ and the next candidate measurement $(\mathbf{X}_{t+1}, \mathbf{y}_{t+1})$. Then the reward function to evaluate each possible sensing action is defined as:

$$\mathcal{R}(\tilde{\boldsymbol{\beta}}, \mathbf{D}_t^j, \mathbf{X}_{t+1}) = -\mathbb{E}_{\mathbf{y}|\mathbf{X}_{t+1}, \tilde{\boldsymbol{\beta}}}[||\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{X}_{t+1}, \mathbf{y}_{t+1}))||_2^2] \tag{4.4}$$

The reward function is stochastic due to the sampling of $\tilde{\beta}$ and this ensures that the search actions selected by the robots are diverse.

4.3.3 GUTS Reward function

During action-selection, each robot optimizes a sensing location by maximizing the reward function in equation (4.4). We observed that this reward function leads to poor search performance at the start of the run. At the start of episode, the posterior belief over target locations is very uncertain and all possible next waypoints have almost-zero probability of finding a target. Hence, this reward function leads to excessively explorative behavior.

We modify the reward function to

$$\mathcal{R}(\tilde{\boldsymbol{\beta}}, \mathbf{D}_t^j, \mathbf{X}_t) = -||\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{X}_t, \mathbf{y}_t))||_2^2 - \lambda \times I(\tilde{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}) \tag{4.5}$$

Where $\lambda(= 0.01)$ is a hyperparameter that reduces the reward for a search location if the estimated $\hat{\boldsymbol{\beta}}$ does not have high likelihood entries in common with the belief of the ground truth at the current step $\tilde{\boldsymbol{\beta}}$. This change improves the discriminatory power of the reward and encourages exploitative actions early in a search run. Formally, let $\hat{k}$ and $\tilde{k}$ be the number of non-zero entries in $\hat{\boldsymbol{\beta}}$ and $\tilde{\boldsymbol{\beta}}$, then the indicator function $I(.)$ is defined as:

$$I(\tilde{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}) = \begin{cases} 0, \text{ if any matches between top } \frac{\hat{k}}{2} \text{ entries in } \hat{\boldsymbol{\beta}} \text{ and top } \frac{\tilde{k}}{2} \text{ entries in } \tilde{\boldsymbol{\beta}} \\ 1, \text{ otherwise} \end{cases}$$

Our experiments show that this modified reward function achieves better search efficiency.

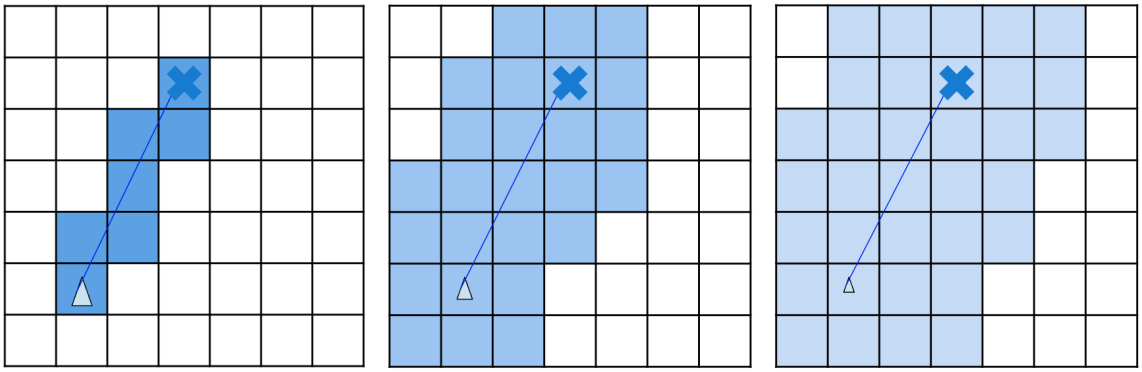
4.3.4 UAV Sensing Action Model

In the general case, Fig. 4.1 shows an exaggerated example of increasing FOV and corresponding increase in depth-aware noise with increasing flight height of the UAV.

For the experiments in this work, the flight height of each UAV is fixed between $70m - 90m$, and have an approximate FOV of $30m \times 30m$ on the ground below it. Additionally, we consider flight paths that are straight lines and hence we include cells that will be flown over en-route to the final waypoint in the reward computation.

Note that NATS [21] does not consider information gain en-route to the waypoint

when planning, it only accounts for the reward at the final waypoint. We correct this modelling error in estimating $\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{X}_{t+1}, \mathbf{y}_{t+1}))$ by integrating over observations on the path to sensing action $\mathbf{X}_t$.



(a) (b) (c)

Figure 4.1: Exaggerated example of UAV sensing action model with different flight heights for the same flight path. The light blue triangle is the UAV, the blue cross is the goal location and the blue line connecting the two is the projected flight path. The shaded region is the expected sensing action given the flight path and flight height. The lighter the shade, the higher is the noise value associated with the observation in that cell. From (left) to (right) we see increasing field-of-view and increasing depth-aware noise.

4.3.5 UGV Sensing Action model

We use a set of high-resolution cameras with total FOV 192° for the ground vehicles. This allows the perception system to pick up detections several hundred metres out. In the general case we would model the FOV as shown in Fig. 4.2, however this approach was prone to error without dynamic updation of the extent of the FOV subject to occulsions. Therefore, we model the FOV of the UGV as a $30m \times 60m$ space directly in front of it. In the GUTS grid representation, this would be the two cells along the bearing of the UGV. We approximate the FOV conservatively. Overestimating the FOV would be disastrous since GUTS would then erroneously assume some parts of search region have been searched and never find targets in that region. Note that the perception module still reports positive detections even beyond the modelled FOV, hence there is no loss of information.

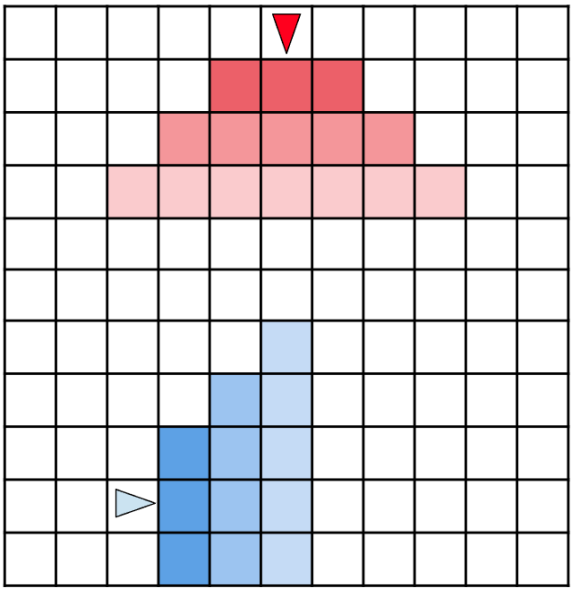
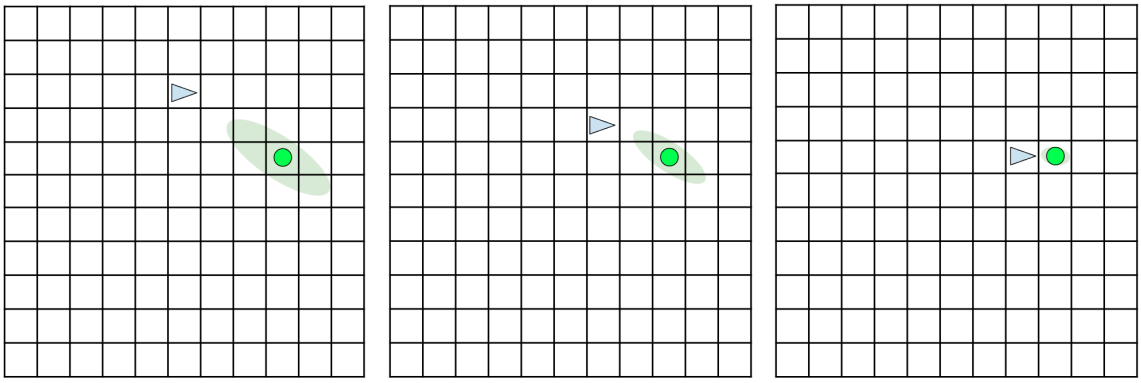


Figure 4.2: A simple illustration of the maximum extent of the robot viewable region given its coordinates and and direction of facing. The robots cannot see beyond the limits of the map or through obstructions as shown by the viewable region of the blue robot. Since this viewshed does not take into account occulsions, we employ a conservative FOV for UGVs as described in Sec. 4.3.5.

We do not consider possible targets en route to a candidate sensing action $\mathbf{X}_t$ for computing the reward function for UGVs because computing and integrating over trajectories for each possible sensing action $\mathbf{X}_t$ is expensive. Each robot still logs and shares observations along the way to a waypoint even though they weren't explicitly accounted for. The problem of integrating trajectory-level information gain for UGVs will be tackled in future work.

4.3.6 Modified Depth-Aware Noise Modelling

We describe our heteroscedastic noise-model for target observations in this section. NATS [21] models this noise using a diagonal covariance matrix where the diagonal elements are proportional to the distance between the observed grid cell and the robot. Intuitively, this represents our confidence in existence of a detected target at the grid cell in consideration. However, this is not a quantity that is actually available for learning-based perception systems.



(a) (b) (c)

Figure 4.3: Representative example of modified depth-aware noise modelling using location uncertainty associated with target tracks over time. The blue triangle is the agent, the green circle is the target and the light-green shaded elliptical region is the location uncertainty of the target as calculated by the agent. From (left) to (right) we see decreasing location uncertainty of the target from the agent's perspective with reducing distance between the agent and the target.

We combine the detection confidence of our object detector with the location uncertainty obtained through Kalman filter to estimate this confidence. A representative example of this is depicted in Fig. 4.3. We employ the following heuristic to approximate the existence uncertainty for positive observations:

$$\sigma^2 = min\left(0.5, \frac{\text{Vol. of target loc uncertainty ellipsoid } (m^3)}{\text{target detection confidence} \times 1000}\right)$$

This formula is based on the intuition that the location uncertainty obtained by the Kalman filter implicitly encodes existence uncertainty as well. We found that this heuristic worked well in our experiments. We re-use the original depth-aware noise-model used in [21] for negative observations.

4.3.7 Accelerating GUTS

We describe a simple example to make the scale of the speedup clear. For an area of 0.7 sq.km, the original NATS implementation [21] took over 60 minutes to compute the first waypoint. After the following three changes, we cut the waypoint selection time to under two minutes:

Table 4.1: Field Testing Results: We report the search parameters (team size, total search area, total targets, algorithm used) with evaluation metrics (runtime, targets found, and search efficiency T/C) for each run. We observe that GUTS outperforms the coverage-based planner in terms of search efficiency (T/C). We also note runs with communication breakdown (CB), hardware failures (HF), and issues with detectors (ND = noisy detector).

Run	Team Size (J)	Search Area (m^2)	Runtime (s)	Targets found (C)	Total targets (M)	T	T/C	Algorithm	Notes
1	1 UAV	$\sim 75,000$	1600	4	7	27	6.75	Coverage	-
2	1 UAV	$\sim 75,000$	1500	5	8	28	5.6	Coverage	-
3	1 UAV	$\sim 75,000$	1000	2	8	8	4	5% GUTS	-
4	1 UAV	$\sim 75,000$	1600	4	8	16	4	5% GUTS	-
5	1 UAV	$\sim 75,000$	500	3	4	5	1.67	GUTS	-
6	2 UGVs	$\sim 75,000$	1000 (avg)	3	3	3	1	GUTS	ND
7	2 UGVs	$\sim 75,000$	600 (avg)	6	6	6	1	GUTS	CB, ND
8	2 UGVs 1 UAV	$\sim 75,000$	1200 (avg)	6	6	13	2.167	GUTS	CB, HF, ND

- GUTS needs to compute $\mathbf{V}$ in (4.2) repeatedly for candidate sensing action $\mathbf{X}_{t+1}$. The matrix inversion step is expensive. We leverage the diagonal nature of $(\mathbf{\Gamma}^{-1} + \mathbf{X}^T\mathbf{\Sigma}\mathbf{X})^{-1}$ to speed up inversion by simply inverting the diagonal elements.
- Additionally, the $\mathbf{X}^T\Sigma\mathbf{X}$ term in (4.2) can be calculated additively as follows:

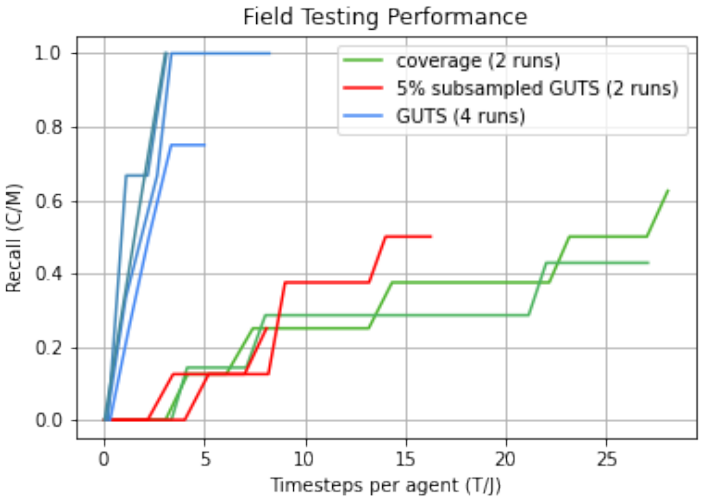
$$\begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}^T \begin{bmatrix}\Sigma_1 & 0\\ 0 & \Sigma_2\end{bmatrix} \begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix} = \mathbf{X}_1^T\Sigma_1\mathbf{X}_1 + \mathbf{X}_2^T\Sigma_2\mathbf{X}_2$$

  This leverages the independence of sensing noise in different grid cells and results in significant-speedup later in the run when $\mathbf{X}$ is high-dimensional.
- For $\boldsymbol{\mu}$ in eqn. 4.2, we exploit the sparse nature of the sensing matrix X to speed up computation [76]

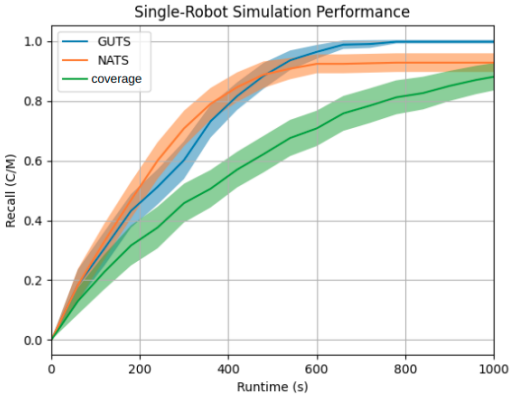
Compute is extremely limited on the UAVs. For the drones we had an additional sampling parameter that subsampled the possible sensing actions the drone could take to dial in the planning time as needed. Typically we tested within the 1% to 10% subsampling range.

4.4 Experiments and Results

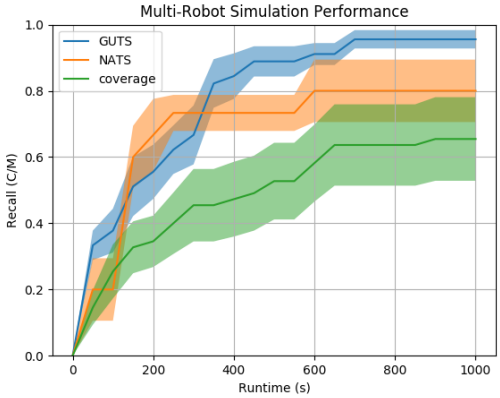
Our experiments aim to demonstrate the improved search efficiency of our proposed algorithm GUTS compared to existing search methods: NATS and coverage-based



(a)



(b)



(c)

Figure 4.4: Comparison of Search Efficiency. (a) We visualize the search efficiency for each run in Table I. We plot the recall versus the number of sensing actions taken per robot. This graph shows that GUTS outperforms the coverage baseline on our system. We also observe that subsampling the total set of waypoints achieves a good balance of computational efficiency and performance, and slighly outperforms the coverage baselines on UAVs. (b and c) We compare the recall rate of different search algorithms vs runtime in a single-robot and multi-robot simulation. We again observe that GUTS outperforms NATS and coverage-based search.

search. We run a computationally optimised (similar to Sec. 4.3.7) version of NATS for our experiments since the original implementation [21] is too slow for the scale of

our experiments. The coverage-based baseline myopically chooses the next waypoint in an unvisited part of the search region. We evaluate these methods on a realistic multi-robot Gazebo [40] simulation. Our simulation uses a pre-mapped costmap of our field test site and accurately simulates the robots' sensing, trajectory planning, and traversal.

We also seek to validate that GUTS can run in real-time on a heterogenous multi-robot team and can function robustly in the face of communication breakdown, hardware failures, and noisy detections. We conduct field tests with a combination of ground and aerial vehicles in a natural unstructured environment with a search area of $\sim 75,000m^2$.

4.4.1 Testing Setup

Each search run is initialized by specifying the search region for each robot (see Fig. 1.3b). Fig 1.3 also shows the launch locations for the UGVs and UAVs, along with the most common target locations. We start evaluating each robot's search performance once it enters the search polygon.

4.4.2 Evaluation Metric

Our primary evaluation metric is the search success rate, which is defined as the fraction of runs in which the search method locates all targets within the prescribed time budget.

For our simulated experiments, we measure search efficiency by plotting recall versus runtime for each algorithm. For our field tests, we compute the number of sensing actions per target found. We prefer to isolate each search algorithm's performance from physical factors such as terrain conditions in field tests and measure the recall in terms of the number of decisions rather than the total time taken. That said, we also report wall clock time for each run, and our total search time per run is always under 30 mins.

4.4.3 Simulated Results

We compare the search efficiency of GUTS with NATS and coverage-based search in Fig. (4.4). We plot results for a single UGV (center plot) and two UGVs (right plot) with 5 targets and a 1000 seconds ($\approx$ 17mins) runtime budget averaged across 20 runs and 10 runs, respectively.

We can see that GUTS clearly outperforms our baselines: NATS and coverage-based search, and is the only search algorithm to consistently recover all the targets within our time budget. GUTS has a success rate of 80% in the multi-robot setting, compared to 40% and 30% success rate for NATS and coverage-based search respectively. GUTS can outperform NATS due to the modification in the reward function described in 4.3.3. Coverage-based search performs poorly because it seeks to navigate the search region exhaustively rather than optimizing its sensing actions for target identification.

GUTS uses a reward function that prioritises rate of recovery of targets, realistically models sensing noise, and is computationally optimized to be run on large environments. We show that GUTS outperforms other state-of-the-art parallelized Thompson sampling-based approaches and coverage-based search to more quickly and consistently recover all objects on our simulated experiments.

4.4.4 Field-Testing Results

We report detailed run parameters and evaluation metrics for our field tests in Table 4.1. We report results for 6 runs of GUTS and 2 runs of our coverage-based baseline. We plot the number of sensing actions required per agent (T/J) against the recall for each run in Fig 4.4 (left). We observe that the GUTS algorithm shows markedly more efficient search behaviour compared to our coverage based baseline.

We also observe that 5% GUTS, which is run on a 5% random subsample of all possible sensing actions, slightly outperforms random coverage as well. We note the runs where we observed communication breakdown or hardware failure show no degradation in performance. The search team sometimes fails to recover all the targets in runs with only UAVs, because some targets are only recoverable from the ground-vehicles. This demonstrates the need to use both UAVs and UGVs to thoroughly canvass the search region.

GUTS handles realistic constraints on robot traversal in heterogeneous teams, is robust to robot hardware and communication failures, and has been experimentally validated in field tests in a large unstructured environment.

Chapter 5

STAR: Stealthy Topography-Aware Reconnaissance

Thus far, the approach to multi-agent active search has been to maximise the rate of recovery while being robust to various failures. In this section, we will consider the case that the search agents must minimise exposure to the targets, thus turning our problem into an adversarial multi-agent active search problem. Our motivation is reconnaissance missions, where the safety of the search agents is at stake and the search region is massive (see Fig. 1.4).

5.1 Related Work

This problem can be broadly classified as adversarial search, however the parameters of the problem can vary significantly. There is a class of work devoted to pursuer-evader scenarios where the primary objective of the search agent(s) is to trap or track the evader with theoretical guarantees for a worst-case evader [27, 28, 37, 39]. However, these often have unrealistic assumptions like complete knowledge or infinite traversal speed of the evader, while this makes sense in a theoretical setting it does not consider the inverse adversarial problem of minimizing information leakage to the adversary (as it already has complete knowledge). Some approaches attempt to use approximate algorithms and relax the requirement for guarantees [45], however none of these approaches can be extended to have an unknown number of evaders.

A natural approach to multiplayer minimax games would be game theory [7, 82], however game-theoretic approaches in dynamic non-cooperative settings require a strong prior on the number of players in order to identify the optimal action. Therefore, in cases when the number of evaders is unknown, game theoretic approaches will yield suboptimal results.

Probabilistic search methods seek to improve expected or average case performance. Bayesian search provides an effective way to model the probabilistic world, inculcate prior information and adapt to information over the course of the search [3, 10]. However, these approaches often rely on perfect observation models (no noise, no false positives) and their examination of optimal behaviour is usually confined to single pursuer or single evader cases [38, 79]. Interestingly, in these highly specified cases using Bayesian search, myopic policies have been shown to be optimal and applicable to cases with noisy observations [41]. In a similar vein, prior work has attempted to model the world using Partially Observable Markov Decision Processes (POMDPs) [56] in an effort to identify optimal policies, however beyond one pursuer, the optimal solution becomes intractable. An excellent survey on adversarial search by Chung et. al. [14] covers an overview of the field and open research questions.

Decentralised adversarial multi-agent active search that is robust to communication failure is an actively researched field. Given the success of the POMDP formulation mentioned earlier in the single-agent case, researchers have attempted to apply reinforcement learning to the problem [5, 11]; however, these approaches are extremely sample inefficient and prone to overfit to the environment they are trained on. In our formulation with known topography, it is not clear if these RL methods can generalize to different topographies.

Topography-aware path planning with adversarial targets is well-studied in the context of military operations [46, 67, 68] and in the context of stealth-based video games [2, 30]. However, these approaches focus on path planning but not on a competing search objective, i.e. they assume that the adversary locations are known or are unknown but don't need to be located, they simply need to be avoided en route to the goal location if encountered.

As opposed to multi-agent search, multi-agent adversarial search has fewer implementations on real-hardware [18, 72, 74, 75], though there are approaches that attempt to validate their results in simulation [12, 26, 79]. Our approach is primarily

validated in a realistic simulation of a field testing site similar in scale and terrain to our actual field testing site. Additionally, our algorithm is efficient enough to run onboard full-size search vehicles and we have performed reconnaissance missions on very large scale search regions. However, we do not share statistics from our real robot experiments as the field testing site is restricted.

5.2 Topographical Visibility Prior

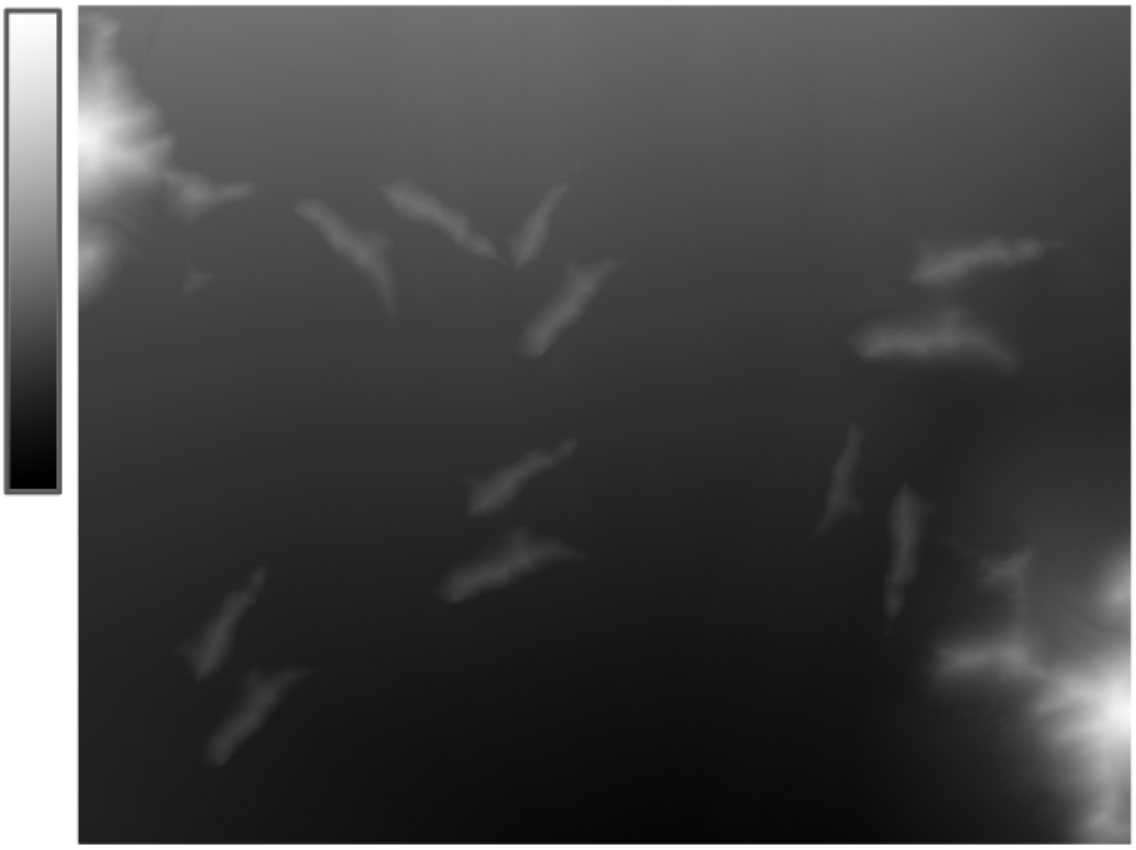


Figure 5.1: Height map of an example search region. The brighter the region, the greater the elevation. This is also known as a Depth Elevation Map (DEM).

Consider the case of a reconnaissance mission in a large unstructured environment such as pictured in Fig. 1.5. We assume that the topography of the region is known. One important aspect of the stealthy reconnaissance problem is 3D spatial reasoning

given the topographical prior of the region, that is, given a location and direction of facing in the map, what would be visible to the agent and conversely if an target was present at this coordinate, what could it potentially see.

Given the topographical prior, we leverage this information to inform our search agents' behaviour. We represent this prior as a digital elevation map (DEM) for a more efficient representation over voxels, given that our use case is in wide open space and our reasoning space is 2D. The DEM of our search region is shown in Fig. 5.1. Ray casting [55] is a common algorithm used to reason about free space in a 3D occupancy grid, however, it is usually an expensive algorithm with varying computation time depending on the actual topography. When doing this in a topographical setting, as opposed to general 3D space, this problem is called viewshed determination in Geographic Information Systems (GIS). Several approaches are optimised to use DEMs to represent 3D space [16, 17, 24, 73]. However, these approaches use sightlines to reason about 3D space similar to ray casting and hence have varying computation time.

In this work we follow [77] and use a reference plane-based approach for viewshed computation, which computes the viewable region from any point in the map in constant time. We assume that the strong topographical prior remains unchanged over the course of the run; however, our physical systems are capable of dynamically updating the topography using point clouds generated by stereo cameras. Hence, having a constant time algorithm for viewshed computation allows for efficient onboard updating if there are differences between the topographical prior and the dynamic observations made by the robot on the ground.

We cover the utilisation of the topography in viewshed computation further in Sec. 5.4.3 and a representative example can be seen in Fig. 5.4.

5.3 Problem Formulation

We model the search region as a grid with a cell size of 60m x 60m. We have access to the costmap for ground robot traversability. Fig 5.6 shows an overhead view and its costmap for an example testing site. During our field tests on real hardware, this costmap was generated using a LiDAR scan several months prior. Some parts of the map may have changed since then, but our on-robot sensing and mapping system

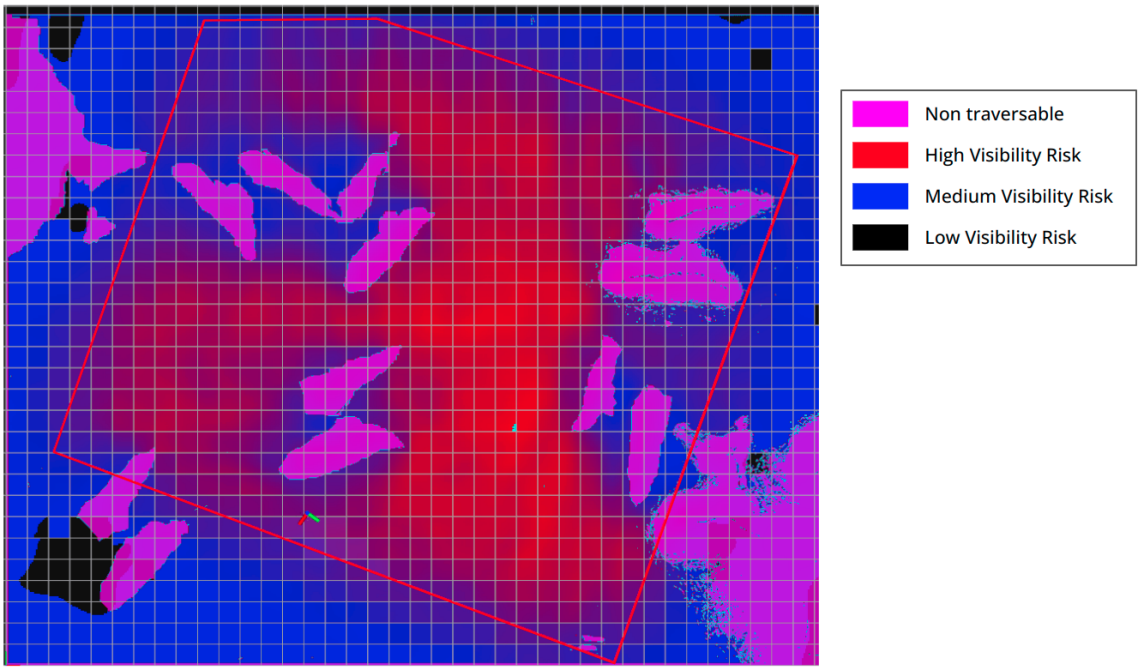


Figure 5.2: Costmap of search region. The grid size here is $60m \times 60m$. The area inside the red polygon is $2.48km^2$. Magenta indicates occupied space where the robot cannot traverse. Black to blue to red indicate increasing topography-aware visibility risk. As intuitively expected, there is greater cover and hence less risk in the vicinity of the mountains. For this particular image we assume that the targets can only be present inside the polygon and hence the visibility risk is lower outside it.

can recognize changes like blocked paths. Due to the ubiquity of satellite imagery, it is reasonable to assume such approximate map information of the search region to be available.

There are several non-traversable areas that cannot be accessed by ground vehicles. These are indicated in magenta. While increasing visibility risk is indicated with the colours black, blue and red.

We model the stealthy active search problem as follows:

- We give the same search region to all robots. (see the red polygon in Fig. 5.6 for an example).
- The targets are sparsely placed and static. They need to be located as fast as possible with high certainty.
- The robots must minimise their exposure to the targets which are considered

hostile.

- Each robot must plan its next data collection action on-board, i.e., no central planner exists.
- The robots may communicate their locations and observations with each other; however, the algorithm's performance should improve with increasing number of search agents anywhere in the spectrum of total absence of communication to perfect communication.

Formally, we represent the locations of targets in our 2D grid representation using a sparse matrix $\mathbf{B} \in \mathbb{R}^{M_1 \times M_2}$. Let $\boldsymbol{\beta} \in \mathbb{R}^M$ be the flattened version of matrix $\mathbf{B}$, where $M = M_1 M_2$. This vector is sparsely populated, with 1 at locations corresponding to targets and 0 elsewhere. We model the uncertainty in our observations by

$$\mathbf{y}_t^j = clip(\mathbf{X}_t^j \boldsymbol{\beta} \pm \mathbf{b}_t^j, 0, 1) \tag{5.1}$$

where, $\mathbf{X}_t^j \in \mathbb{R}^{Q \times M}$ describes the sensing matrix such that each row in $\mathbf{X}_t^j$ is a one-hot vector indicating one of the grid cells in view of the robot j at timestep t, and Q is the total number of grid cells the robot can view. $\mathbf{y}_t^j \in \mathbb{R}^{Q \times 1}$ is the resultant observation including the additive topography-aware noise vector $\mathbf{b}_t^j \in \mathbb{R}^{Q \times 1}$. Going forward, we assume that these quantities are defined on a per-robot basis and drop the superscript j for ease of notation. The topography-aware noise $\mathbf{b}_t$ has two components; firstly, it encodes the intuition that observation uncertainty increases with distance to the robots; secondly, it encodes the intuition that observation uncertainty increases with occlusions in the line of sight from the robot. We model it as follows:

$$\mathbf{b}_t = \mathbf{n}_t / \mathbf{v}_t \tag{5.2}$$

here, / denotes element wise division, $\mathbf{n}_t \sim \mathcal{N}^+(0, \Sigma_t)$ with diagonal elements of the noise covariance matrix Σ_t monotonically increasing with the distance of the observed cell from the robot. $\mathbf{v}_t \in \mathbb{R}^{Q \times 1}$ where each entry represents the square of the fractional visibility (accounting for occlusions) of each of the Q cells visible in $\mathbf{X}_t$. The noise is sampled from a positive half-Gaussian distribution $\mathcal{N}^+(0, \Sigma_t)$ and is added for cells without targets and subtracted for cells with targets. The observation $\mathbf{y}_t$ is clipped to be within 0 and 1. Similarly, we have an observation model for the

targets albeit with some relaxations, namely, only accounting for occlusions but no depth-aware noise.

Let the robot trajectory for robot j until timestep t be denoted $\mathbf{L}_t^j \in \mathbb{R}^M$ such that each entry m is the integer count of the number of times robot j has visited that cell in the whole space M. We define a penalty function $\mathcal{P}(\mathbf{L}_t^j)$, which penalizes the robot for showing itself to any of the targets.

$$\mathcal{P}(\mathbf{L}_t^j) = \sum_{q}^{Q'} \sum_{k} \mathbf{X}_k \mathbf{L}_t^j \tag{5.3}$$

Similar to above, $\mathbf{X}_k \in \mathbb{R}^{Q' \times M}$ is the sensing matrix for the k^{th} target, note that it is not dependent on the timestep t as targets are static. The second summation is to reduce the value to a single real number.

Let $\mathbf{D}_t^j$ be the set of observations available to robot j at timestep t. $\mathbf{D}_t^j$ comprises of $(\mathbf{X}_t, \mathbf{y}_t)$ pairs collected by robot j as well as those communicated to robot j by other robots. Let the total number of sensing actions by all agents be T. Our main objective is to sequentially optimize the next sensing action $\mathbf{X}_{t+1}$ based on $\mathbf{D}_t^j$ at each timestep t to recover the sparse signal $\boldsymbol{\beta}$ with as few measurements T as possible while minimizing the stealth penalty $\sum_j \mathcal{P}(\mathbf{L}_t^j)$. Each robot optimizes this objective based on its own partial dataset $\mathbf{D}_t^j$ in a decentralized manner.

5.4 Algorithm

This section presents the STAR algorithm. Each robot j asynchronously estimates the posterior distribution over target locations based on its partial dataset $\mathbf{D}_t^j$. During the action selection stage, each robot generates a sample from this posterior and simultaneously optimizes a reward function and a stealth penalty for this sampled set of target locations. The reward function represents potential information gain for the sensing action in consideration, while the stealth penalty represents the potential information leakage based on the partial dataset $\mathbf{D}_t^j$ available. The algorithm has been summarized in Alg. 2

The reward computation and posterior sampling mirrors the GUTS algorithm presented in Sec. 4.3.1 and Sec. 4.3.3. In Sec. 4.3.3 and onward, we describe the new

Algorithm 2 STAR Algorithm

Assume: Sensing model (5.1), sparse signal $\boldsymbol{\beta}$, J agents
Set: $\mathbf{D}_0^j \leftarrow \emptyset$, $\mathbf{L}_0^j \leftarrow \{x_j, y_j\}\ \forall\ j \in \{1, ..., J\}$, $\gamma_m = 1\ \forall m \in \{1, ..., M\}$
for $t = 1, ..., T$ **do**
  Wait for an agent to finish; for the free agent j:
  Sample $\tilde{\boldsymbol{\beta}} \sim p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\Gamma}) = \mathcal{N}(\boldsymbol{\mu}, \mathbf{V})$ from (5.4)
  $\mathbf{X}_t, \mathbf{l}_t = \arg\max_{\tilde{\mathbf{X}}, \tilde{\mathbf{l}}} \left(\mathcal{R}(\tilde{\boldsymbol{\beta}}, \mathbf{D}_t^j, \tilde{\mathbf{X}}) - \gamma \mathcal{P}(\tilde{\mathbf{l}}) \right)$ from (5.6) and (5.7)
  Observe $\mathbf{y}_t$ given action $\mathbf{X}_t$
  Update $\mathbf{D}_{t+1}^j = \mathbf{D}_t^j \cup (\mathbf{X}_t, \mathbf{y}_t)$ (robot observations)
  Update $\mathbf{L}_{t+1}^j = \mathbf{L}_t^j \cup (\mathbf{l}_t)$ (robot path)
  Share $(\mathbf{X}_t, \mathbf{y}_t)$ Estimate $\boldsymbol{\Gamma} = diag([\gamma_1, ..., \gamma_M])$ using (5.5)
end for

reward function to improve search performance, the depth-aware noise models for our UGVs and UAVs, and speed-up techniques to optimise the code to run on large environments.

5.4.1 Calculating Posterior

Each robot assumes a zero-mean gaussian prior per entry of the vector '$\boldsymbol{\beta}$ s.t. $p_0(\beta_m) = \mathcal{N}(0, \gamma_m)$. The variances $\boldsymbol{\Gamma} = diag([\gamma_1 ... \gamma_M])$ are hidden variables which are estimated using data. We follow [71], [78] and use a conjugate inverse gamma prior on γ_m to enforce sparsity s.t. $p(\gamma_m) = IG(a_m, b_m) = \frac{b_m^{a_m}}{\Gamma(a_m)} \gamma_m^{(-a_m - 1)} e^{-(b_m/\gamma_m)}\ \forall\ m \in \{1 ... M\}$. We estimate the posterior distribution on $\boldsymbol{\beta}$ given data $\mathbf{D}_t^j$ for robot j using EM.

We can write analytic expressions for the E-step (estimating $\hat{\boldsymbol{\beta}} = p(\boldsymbol{\beta}|\mathbf{D}_t^j, \boldsymbol{\Gamma}) = N(\boldsymbol{\mu}, \mathbf{V})$) and M-step (computing $\max_{\boldsymbol{\Gamma}} p(\mathbf{D}_t^j|\boldsymbol{\beta}, \boldsymbol{\Gamma})$) respectively:

$$\mathbf{V} = (\boldsymbol{\Gamma}^{-1} + \mathbf{X}^T \boldsymbol{\Sigma} \mathbf{X})^{-1}; \boldsymbol{\mu} = \mathbf{V} \mathbf{X}^T \boldsymbol{\Sigma} \mathbf{y} \tag{5.4}$$

$$\gamma_m = ([\mathbf{V}]_{mm} + [\boldsymbol{\mu}]_m^2 + 2b_m)/(1 + 2a_m) \tag{5.5}$$

where $\mathbf{X}$ and $\mathbf{y}$ are created by vertically stacking all measurements $(\mathbf{X}_t, \mathbf{y}_t)$ in $\mathbf{D}_t^j$, and $\boldsymbol{\Sigma}$ is a diagonal matrix composed of their corresponding depth-aware noise variances.

Each robot estimates $p(\boldsymbol{\beta}|\mathbf{D}_t^j) = N(\boldsymbol{\mu}, \mathbf{V})$ on-board using it's partial dataset D_t^j. We drop the index j from equations (4.2) and (4.3) for clarity. We set the values

of $a_m = 0.1$ and $b_m = 1$ as these were found to be effective in [21]. Finally, agent j samples from the posterior $\tilde{\boldsymbol{\beta}} \sim p(\boldsymbol{\beta}|\mathbf{D}_t^j)$.

5.4.2 Choosing Next Sensing Action

Each robot chooses the next sensing action $\mathbf{X}_{t+1}$ by assuming that the sampled set of target locations $\tilde{\boldsymbol{\beta}}$ is correct. Specifically, let $\hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{X}_t, \mathbf{y}_t))$ be our expected estimate of the parameter β using all available measurements $\mathbf{D}_t^j$ and the next candidate measurement $(\mathbf{X}_{t+1}, \mathbf{y}_{t+1})$. Then, as presented in Sec. 4.3.3 the GUTS reward function to evaluate each possible sensing action is defined as:

$$\mathcal{R}(\tilde{\boldsymbol{\beta}}, \mathbf{D}_t^j, \mathbf{X}_t) = -||\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}(\mathbf{D}_t^j \cup (\mathbf{X}_t, \mathbf{y}_t))||_2^2 - \lambda \times I(\tilde{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}) \tag{5.6}$$

Where $\lambda(= 0.01)$ is a hyperparameter that reduces the reward for a search location if the estimated $\hat{\boldsymbol{\beta}}$ does not have high likelihood entries in common with the belief of the ground truth at the current step $\tilde{\boldsymbol{\beta}}$. Let $\hat{k}$ and $\tilde{k}$ be the number of non-zero entries in $\hat{\boldsymbol{\beta}}$ and $\tilde{\boldsymbol{\beta}}$, then the indicator function $I(.)$ is defined as:

$$I(\tilde{\boldsymbol{\beta}}, \hat{\boldsymbol{\beta}}) = \begin{cases} 0, \text{ if any matches between top } \frac{\hat{k}}{2} \text{ entries in } \hat{\boldsymbol{\beta}} \text{ and top } \frac{\tilde{k}}{2} \text{ entries in } \tilde{\boldsymbol{\beta}} \\ 1, \text{ otherwise} \end{cases}$$

The reward function is stochastic due to the sampling of $\tilde{\beta}$ and this ensures that the search actions selected by the robots are diverse.

This reward function needs to be balanced against the stealthy penalty term that was defined in Eqn. 5.3 except that instead of the robot trajectory till timestep t we calculate the penalty for the next possible goal location of the robot j which we denote as $\mathbf{l}_t^j$. $\mathbf{l}_t^j \in \mathbb{R}^M$ is a one hot vector indicating the location of the robot corresponding to the potential measurement $\mathbf{X}_t^j$ in the reward defined in Eqn. 5.6. This penalty function can be mathematically represented as follows:

$$\mathcal{P}(\mathbf{l}_t^j) = \sum_{q}^{Q'} \sum_{k} \mathbf{X}_k \mathbf{l}_t^j \tag{5.7}$$

The overall optimisation objective can then be thought of as two competing

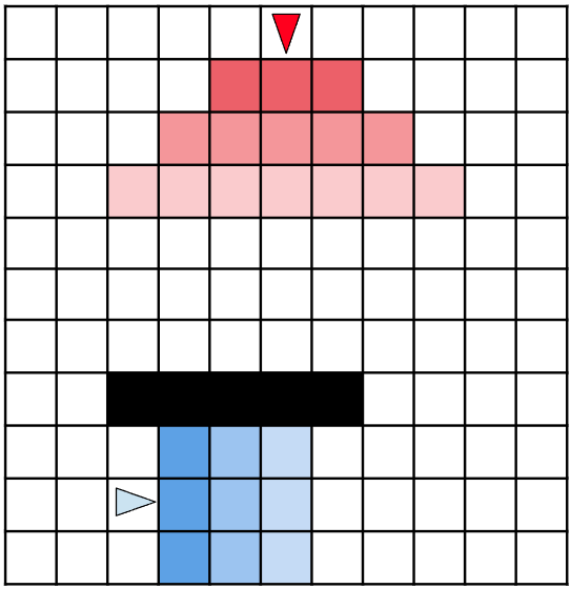


Figure 5.3: A simple illustration of the maximum extent of the robot viewable region given its coordinates and and direction of facing. The black cells represent obstructed cells. The robots cannot see beyond the limits of the map or through obstructions as shown by the viewable region of the blue robot.

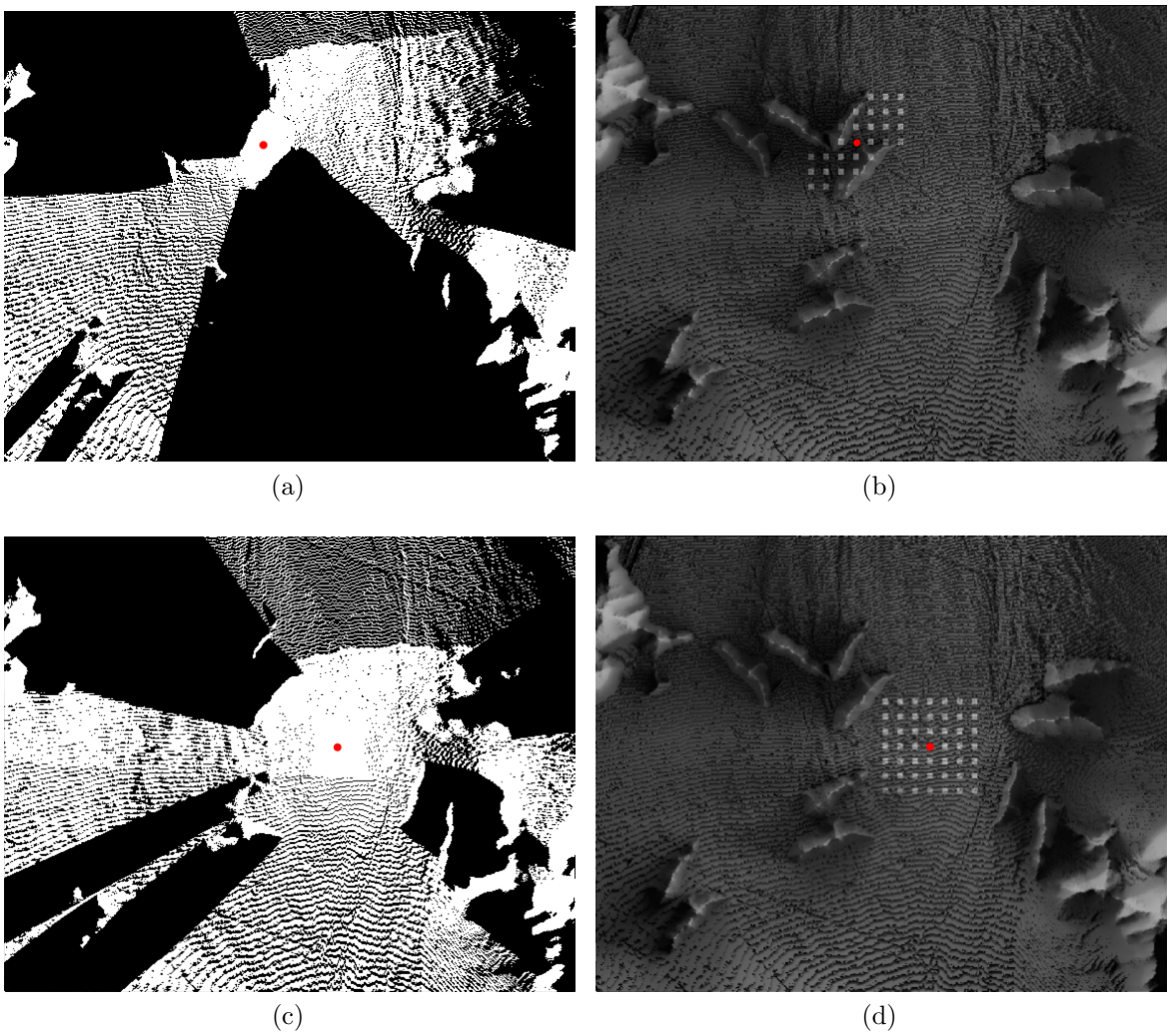
objectives as follows:

$$\mathbf{X}_t, \mathbf{l}_t = \arg\max_{\tilde{\mathbf{X}}, \tilde{\mathbf{l}}} \left(\mathcal{R}(\boldsymbol{\beta}^*, \mathbf{D}_t^j, \tilde{\mathbf{X}}) - \gamma \mathcal{P}(\tilde{\mathbf{l}}) \right) \text{ from (5.6) and (5.7)} \tag{5.8}$$

where, γ is a hyperparameter controls the tradeoff between goal selection to satisfy the stealth penalty and the GUTS reward term. We found that the best value for γ is 1 combined with normalising both the reward and stealthy penalty terms between 0 and 1.

5.4.3 UGV Sensing Action model

We use a set of high-resolution cameras with total FOV 193° for the ground vehicles. This allows the perception system to pick up detections several hundred metres out. The primary difference with Sec. 4.3.5 is that the sensing action model is sensitive to the topography. It's full extent is upto $210m - 300m$ in front of the robot as shown in Fig. 5.3 subject to occlusions.



(a) (b)

(c) (d)

Figure 5.4: Representative examples of extent of field-of-view of ground based agents. (Left) Following [77] we generate the viewshed of the agent in all directions if it were located at the red dot. White indicates visible to the agent and black indicates invisible. (Right) We impose viewing limits ($200m - 300m$), discretise the viewshed to the $60m \times 60m$ cell size grid and superimpose this viewshed on the topographical map. Note that the when the agent is located between the two mountains its viewshed is constricted, whereas out in the open it can see till the modelled limits.

In the grid representation, this would be a trapezium of fifteen cells along the bearing of the UGV. We approximate the FOV conservatively as we have found the perception module to be less reliable in the periphery of the 193° FOV. Note that the perception module still reports positive detections even beyond the modelled FOV

when testing on real hardware, hence there is no loss of information.

Fig. 5.4 shows a representative example of the viewshed computed at two locations in the the ficitious map shown earlier in Fig. 1.5 assuming a 360° FOV. As we can see, based on sightlines the area of the map that is visible is much greater than the $300m$ limit we mentioned earlier, however, in practical scenarios beyond a certain distance even if the terrain is visible, it is not possible to make accurate detections of targets as they are just a few pixels in the image. The real-world interpretation of this is that the depth-aware noise gets so high beyond a certain point that it overshadows the true signal which is the presence/absence of the targets.

We do not consider possible targets en route to a candidate sensing action $\mathbf{X}_t$ for computing the reward function for UGVs because computing and integrating over trajectories for each possible sensing action $\mathbf{X}_t$ is expensive. Each robot still logs and shares observations along the way to a waypoint even though they weren't explicitly accounted for. We can intuitively posit that the expected information gain as a result of executing a sensing action will be greater than or equal to what is calculated with the sensing model described above. Hence the expected reward and stealth penalty is at least that which is computed in Eqn. 5.8.

5.4.4 Target Sensing Action Model

Fig. 5.5 shows a representative example of the viewshed of the targets. They are not subject to depth aware noise but they are subject to topographical constraints, i.e., they cannot see through occlusions. Since we don't have information on the direction of facing of the targets, we model the FOV such that targets see in all directions subject to the topography. the $210m - 300m$ viewing limit but without depth aware noise.

5.4.5 Visibility Risk Aware Path Planning

A natural question at this point, is that our viewing models imply that detecting a target is accompanied by the target detecting the search agent, if this is the case, then how can stealth be achieved? The value in our approach is that identifying a target and being identified once does not mean the mission is over, there are more possible targets to be located and known targets should be avoided for the remainder of the

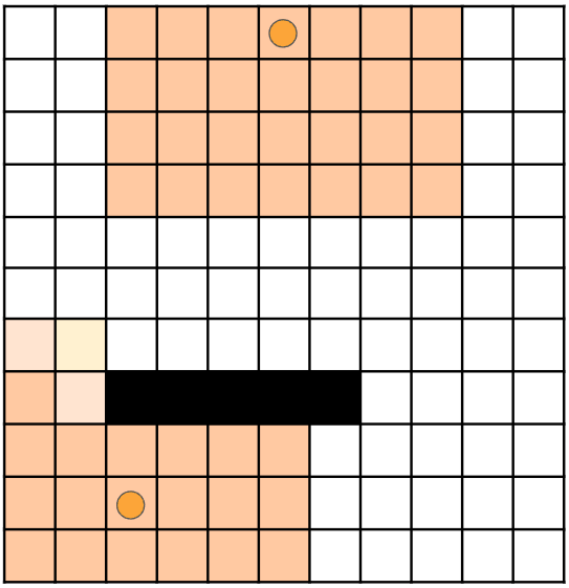


Figure 5.5: A simple illustration of the maximum extent of the target's viewable region given its coordinates. We assume we don't know it's direction of facing and hence it has a 360° field-of-view. The black cells represent obstructed cells. The targets cannot see beyond the limits of the map or through obstructions as shown by the viewable region of the blue robot.

search. Hence we expect to minimize the stealth penalty over the course of the run but don't expect it to be zero. As an aside, were we to employ asymmetric viewing models such that viewing targets without being viewed was possible, we might aim to have zero risk policies but we outline this for future work. In order for the search agents to respect the visibility risk map, an example of which can be seen in Fig. 5.6, we use the OMPL planner [66] which is an off-the-shelf path planning library available in ROS [65] which can plan paths within time constraints and subject to state costs, which in our case is the visibility risk map, and a traversability map, which is an occupancy map of obstacles.

5.5 Experiments and Results

Our experiments aim to demonstrate the improved search efficiency of our proposed algorithm STAR compared to existing search methods: GUTS, RSI, coverage-based search and random search. The GUTS algorithm is as described in Sec. 4.3 except that the sensing model used follows that described in Sec. 5.4.3, i.e., it is topography-aware

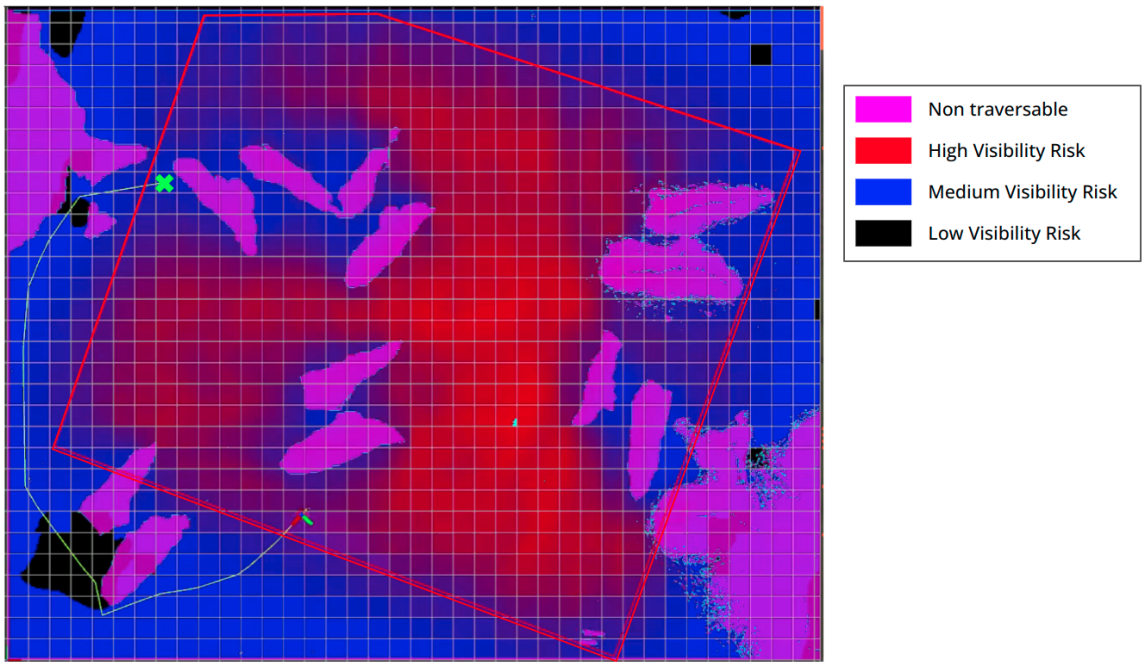


Figure 5.6: Costmap of search region. The grid size here is $60m \times 60m$. The area inside the red polygon is $2.48km^2$. Magenta indicates occupied space where the robot cannot traverse. Black to blue to red indicate increasing topography-aware visibility risk. The green cross is the goal location provided to the OMPL [66] path planner and the green path is the path computed to the goal location while minimizing the visibility risk.

GUTS. Region Sensing Index (RSI) [44] is an active search algorithm that locates sparse targets while taking into account realistic sensing constraints based on greedy maximisation of information gain. The coverage-based baseline myopically chooses the next waypoint in an unvisited part of the search region while the random search policy randomly selects a cell to visit. We evaluate these methods on a realistic multi-robot Gazebo [40] simulation. Our simulation uses a pre-mapped costmap of our field test site and accurately simulates the robots' sensing, trajectory planning, and traversal.

We also seek to validate that STAR can run in real-time on a multi-robot team. We conduct field tests in a natural unstructured environment with a search area of $\sim 4km^2$. This was a restricted site and thus we do not present those results here.

5.5.1 Testing Setup

Each search run is initialized by specifying the search region for each robot (see Fig. 5.6). We evaluate the various search algorithms under two target sampling paradigms: uniform and adversarial. In uniform sampling the targets are placed uniformly at random in the search region. In adversarial sampling the targets are placed stealthily, i.e., they are placed in locations with a lower average visibility within the search region. We control for the locations of the targets using these two paradigms. There are always 5 targets in the simulated experiments.

5.5.2 Evaluation Metrics

Our primary evaluation metric is the recovery rate, which is defined as the fraction of targets the search method has located against the number of search decisions made within the time budget. Our secondary metric is the stealth penalty incurred by the team of search agents.

For our simulated experiments, we measure search efficiency by plotting recall versus number of decision steps per agent for each algorithm. We prefer to isolate each search algorithm's performance from physical factors such as terrain conditions in our tests and measure the recall in terms of the number of decisions rather than the total time taken. Our total search time per run is always under 1 hr 15 mins. As a reminder, the search region is massive ($\sim 2.5km^2$) and the vehicle speeds are restricted to 5 m/s.

Similarly, we measure the stealth performance by plotting the stealth penalty incurred against the number of decision steps per agent for each algorithm.

5.5.3 Simulated Results

We compare the search efficiency of STAR, with GUTS, RSI, random search and coverage-based search in Fig. 5.7 for adversarially placed targets and Fig. 5.8 for uniformly placed targets. In both figures, we plot results for a single UGV (upper plots) and two UGVs (lower plots) with five targets and a 1 hr 15 min runtime budget averaged across 10-12 runs.

We can see that STAR outperforms our baselines: GUTS, RSI, coverage-based

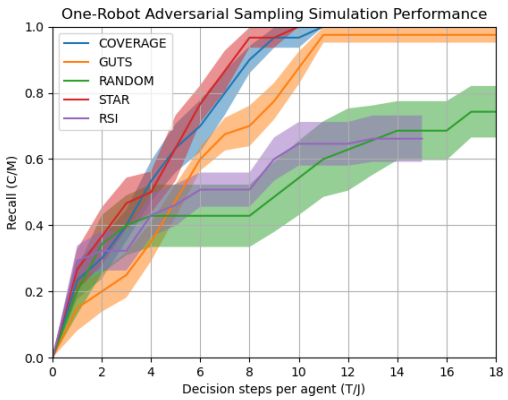
search and random search on the recovery rate (left column) metric as well as in terms of the stealth penalty (right column) regardless if the targets are placed adversarially or uniformly at random.

What is most interesting is that despite optimising to reduce the stealth penalty, STAR still outperforms the other algorithms on recovery rate. This indicates that the stealth penalty term which encodes topography-awareness improves search quality by encouraging the agents to stay in and hence look in more well-hidden locations. We also note that RSI, which is an information-greedy approach, fails to show improved performance in the multi-agent setting, whereas, even the random policy outperforms it in the multi-agent setting due to the diversity in actions.

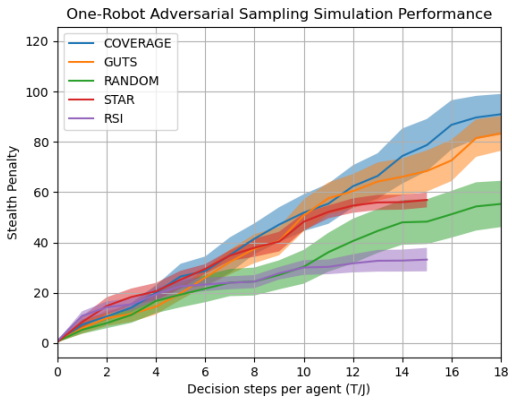
STAR uses a multi-objective reward function that not only prioritises rate of recovery of targets but also limits exposure to potential targets. STAR is optimized to run on large environments, realistically models sensing noise, and uses topographical information to inform search and path planning. We show that STAR outperforms GUTS, the previous best algorithm in terms of search efficiency, and RSI, an information-greedy approach designed to handle noisy observations and sparsely located targets.

5.5.4 Field-Testing Results

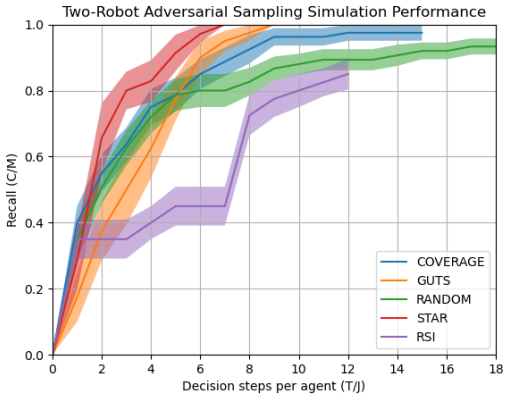
We tested STAR at a restricted field testing site and thus do not present those results here. We can report that the algorithm worked well in that setting, just as the simulation results have demonstrated.



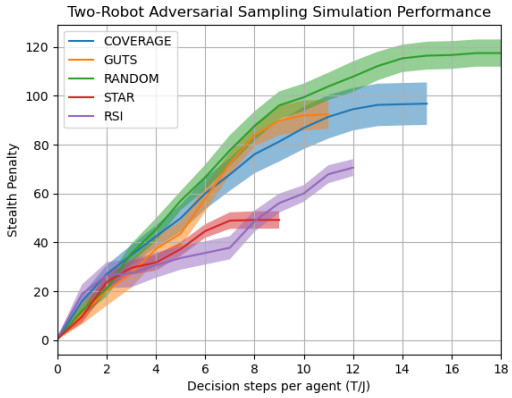
(a) Single Robot Search Performance



(b) Single Robot Stealth Performance

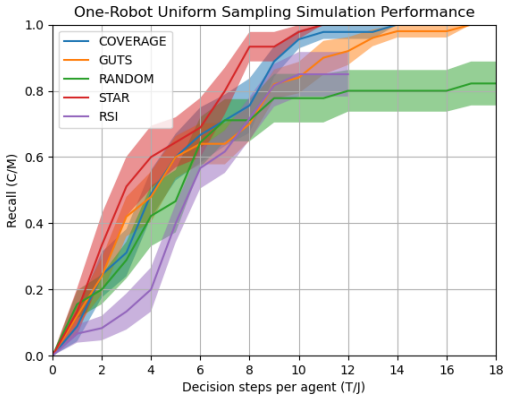


(c) Multi-Robot Search Performance

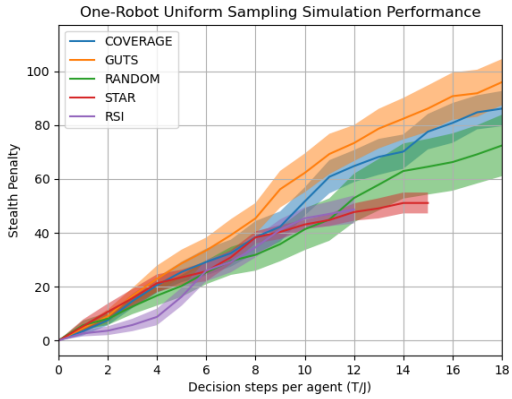


(d) Multi-Robot Stealth Performance

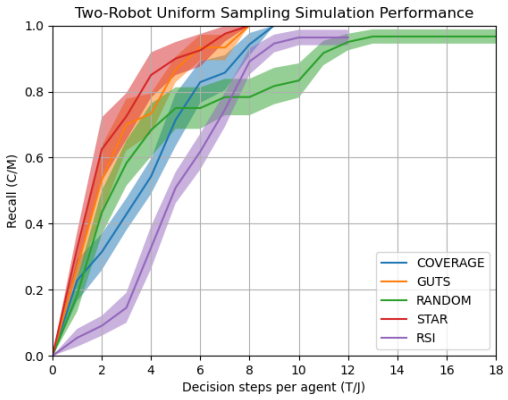
Figure 5.7: Comparison of Search Efficiency and Stealth Performance When Targets Are Placed Adversarially. **(a) and (c)**: These graphs show that STAR outperforms GUTS, RSI, coverage-based search and random search on the primary metric of recovery (recall) rate. It also shows that increasing the number of search agents improves search efficiency. **(b) and (d)**: We observe that STAR outperforms GUTS, RSI, coverage-based search and random search on the secondary metric of stealth penalty. The end of the line indicates the end of the runtime budget and hence the stealth penalty at the end of each curve is the final stealth penalty for that run. We note that while STAR starts off similar to the other approaches (as the agent must show itself to an target in order to detect it), it proceeds cautiously later in the run in order to achieve a better score. RSI is able to have a lower stealth penalty than STAR in the single-robot case as it fails to locate the targets in most runs.



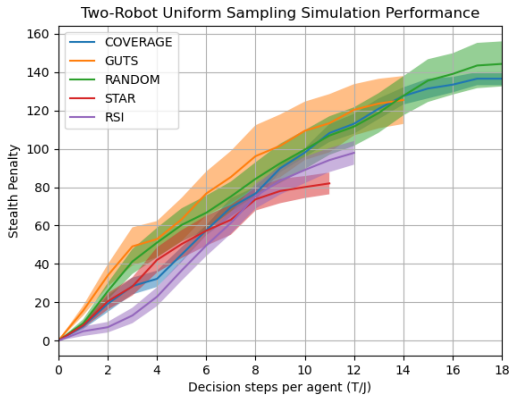
(a) Single Robot Search Performance



(b) Single Robot Stealth Performance



(c) Multi-Robot Search Performance



(d) Multi-Robot Stealth Performance

Figure 5.8: Comparison of Search Efficiency and Stealth Performance When Targets Are Placed Uniformly at Random. **(a) and (c)**: These graphs show that STAR outperforms GUTS, RSI, coverage-based search and random search on the primary metric of recovery (recall) rate. Notably, we show that STAR's good performance in Fig. 5.7 was not due to how the targets were placed, rather STAR encourages searching in well-hidden locations in an effort to remain concealed and even when targets are placed uniformly at random it gives it an edge as invariably some targets will end up in well concealed locations. **(b) and (d)**: We again observe that STAR outperforms GUTS, RSI, coverage-based search and random search on the secondary metric of stealth penalty. The end of the line indicates the end of the runtime budget and hence the stealth penalty at the end of each curve is the final stealth penalty for that run. We note that while STAR starts off similar to the other approaches (as the agent must show itself to an target in order to detect it), it proceeds cautiously later in the run in order to achieve a better score.

Chapter 6

Conclusion

Whether for disaster response or for reconnaissance, search missions pose several challenges: time-criticality, unreliable communication, noisy observations, occlusions, terrain constraints, uncertainty over the number of targets, possibility of hardware failures and limited on-board compute.

We presented two novel algorithms, Generalized Uncertainty-aware Thompson Sampling (GUTS) and Stealthy Topography-Aware Reconnaissance (STAR), for efficient multi-agent active search and validated them in simulation and on real hardware.

GUTS uses a reward function that prioritises rate of recovery of targets, realistically models sensing noise, and is computationally optimized to be run on large environments. We show that GUTS outperforms other state-of-the-art parallelized Thompson sampling-based approaches and coverage-based search to more quickly and consistently recover all objects on our simulated experiments.

STAR utilizes a strong topographical prior to inform search and path planning in order to locate hostile targets in a time-efficient manner while minimizing exposure to the targets. STAR outperforms GUTS, RSI, coverage-based search and random search on rate of recovery of targets while having minimal exposure to the threats.

Both algorithms handle realistic constraints on robot traversal in heterogeneous teams, are robust to robot hardware and communication failures, and have been experimentally validated in field tests in a large unstructured environment and in simulations that accurately model our field testing sites.

While the results of both algorithms are promising and experimentally validated, the question of theoretical bounds on performance remains open and shall be tackled in future work. Additionally, while the parallelized Thompson sampling framework tends to converge to the optimal policy in the long run, the stealth penalty used in STAR does not have any optimality guarantees. STAR outperforming GUTS is a positive empirical result but further analysis is needed to assess STAR from an optimality perspective.

The results shown in this dissertation demonstrate that fully autonomous robots can effectively search in complicated natural terrains in a time efficient manner with limited on-board compute. We believe the algorithms presented in the paper pave the way for more ubiquitous application of autonomous robotics in multi-agent search for disaster response and reconnaissance and will save human effort and human lives with greater adoption.

Bibliography

[1] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984. ISSN 0166-218X. doi: https://doi.org/10.1016/0166-218X(84)90073-8. 1.2

[2] Wael Al Enezi and Clark Verbrugge. Skeleton-based multi-agent opponent search. In *2021 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2021. 1.2, 5.1

[3] David Assaf and Shmuel Zamir. Optimal Sequential Search: A Bayesian Approach. *The Annals of Statistics*, 13(3):1213 – 1221, 1985. 5.1

[4] Elif Ayvali, Hadi Salman, and Howie Choset. Ergodic coverage in constrained environments using stochastic trajectory optimization. 2017. 4.1

[5] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. Emergent tool use from multi-agent autocurricula. In *International Conference on Learning Representations*, 2020. 5.1

[6] J. L. Baxter, E. K. Burke, J. M. Garibaldi, and M. Norman. *Multi-Robot Search and Rescue: A Potential Field Based Approach*, pages 9–16. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 978-3-540-73424-6. doi: 10.1007/978-3-540-73424-6_2. 1.1

[7] Tamer Başar and Geert Jan Olsder. *Dynamic Noncooperative Game Theory, 2nd Edition*. Society for Industrial and Applied Mathematics, 1998. 5.1

[8] Graeme Best, Oliver M Cliff, Timothy Patten, Ramgopal R Mettu, and Robert Fitch. Dec-MCTS: Decentralized planning for multi-robot active perception. *Int. J. Rob. Res.*, 38(2-3):316–337, 2019. 4.1

[9] Richard Borie, Craig Tovey, and Sven Koenig. Algorithms and complexity results for pursuit-evasion problems. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, IJCAI'09, page 59–66, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc. 1.2

[10] Frédéric Bourgault, Ali Göktogan, Tomonari Furukawa, and Hugh F. Durrant-Whyte. Coordinated search for a lost target in a bayesian world. *Advanced Robotics*, 18(10):979–1000, 2004. 5.1

[11] Jan Buermann and Jie Zhang. Multi-robot adversarial patrolling strategies via lattice paths. *Artificial Intelligence*, 311:103769, 2022. ISSN 0004-3702. 1.2, 5.1

[12] Jan Buermann and Jie Zhang. Multi-robot adversarial patrolling strategies via lattice paths. *Artificial Intelligence*, 311:103769, 2022. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2022.103769. 5.1

[13] Shih-Yi Chien, Huadong Wang, and Michael Lewis. Human vs. algorithmic path planning for search and rescue by robot teams. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 54(4):379–383, 2010. doi: 10.1177/154193121005400423. 4.1

[14] Timothy H Chung, Geoffrey A Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics: A survey. *Auton. Robots*, 31(4):299–316, 2011. 5.1

[15] Daniel S Drew. Multi-agent systems for search and rescue applications. *Curr Robot Rep*, 2(2):189–200, 2021. 1.1

[16] Peter Fisher. First experiments in viewshed uncertainty: The accuracy of the viewshed area. *Photogrammetric Engineering Remote Sensing*, 57, 01 1991. 5.2

[17] Peter F. Fisher. Extending the applicability of viewsheds in landscape planning. *Photogrammetric Engineering and Remote Sensing*, 52:1297–1302, 1996. 5.2

[18] Brian P. Gerkey, Sebastian Thrun, and Geoff Gordon. Parallel stochastic hill-climbing with small teams. In Lynne E. Parker, Frank E. Schneider, and Alan C. Schultz, editors, *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III*, pages 65–77, Dordrecht, 2005. Springer Netherlands. ISBN 978-1-4020-3389-6. 5.1

[19] Payam Ghassemi and Souma Chowdhury. Decentralized informative path planning with balanced exploration-exploitation for swarm robotic search. 08 2019. doi: 10.1115/DETC2019-97887. 1.1, 4.1

[20] Daniel Aiham Ghazali, Maximilien Guericolas, Frédéric Thys, François Sarasin, Pedro Arcos González, and Enrique Casalino. Climate change impacts on disaster and emergency medicine focusing on mitigation disruptive effects: an international perspective. *Int J Environ Res Public Health*, 15(7), July 2018. 1.1

[21] Ramina Ghods, William J Durkin, and Jeff Schneider. Multi-Agent active search using realistic Depth-Aware noise model. *CoRR*, abs/2011.04825, 2020. 1.1, 1.1, 1.2, 1.3, 2.1, 2.3, 4.1, 4.3, 4.3.1, 4.3.4, 4.3.6, 4.3.7, 4.4, 5.4.1

[22] Ramina Ghods, Arundhati Banerjee, and Jeff Schneider. Decentralized multi-agent active search for sparse signals. In Cassio de Campos and Marloes H. Maathuis, editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 696–706. PMLR, 27–30 Jul 2021. 1.1, 1.1, 1.2, 1.3, 2.1

[23] Arthur S. Goldstein and Edward M. Reingold. The complexity of pursuit on a graph. *Theor. Comput. Sci.*, 143:93–112, 1995. 1.2

[24] Michael F. Goodchild and Jay Lee. Coverage problems and visibility regions on topographic surfaces. *Annals of Operations Research*, 18(1):175–186, Dec 1989. ISSN 1572-9338. doi: 10.1007/BF02097802. 5.2

[25] Faine Greenwood, Erica L. Nelson, and P. Gregg Greenough. Flying into the hurricane: a case study of uav use in damage assessment during the 2017 hurricanes in texas and florida. *PLoS one*, 15(2):e0227808–e0227808, 2020. ISSN 1932-6203. doi: 10.1371/journal.pone.0227808. 1.1

[26] Geoffrey Hollinger, Sanjiv Singh, Joseph Djugash, and Athanasios Kehagias. Efficient multi-robot search for a moving target. *Int. J. Rob. Res.*, 28(2):201–219, 2009. 5.1

[27] Geoffrey Hollinger, Athanasios Kehagias, and Sanjiv Singh. Gsst: Anytime guaranteed search. *Auton. Robots*, 29(1):99–118, jul 2010. ISSN 0929-5593. doi: 10.1007/s10514-010-9189-9. 5.1

[28] Geoffrey Hollinger, Sanjiv Singh, and Athanasios Kehagias. Improving the efficiency of clearing with multi-agent teams. *The International Journal of Robotics Research*, 29(8):1088–1105, 2010. doi: 10.1177/0278364910369949. 5.1

[29] Conor Igoe, Ramina Ghods, and Jeff Schneider. Multi-agent active search: A reinforcement learning approach. *IEEE Robot. Autom. Lett.*, 7(2):754–761, 2022. 4.1

[30] Damián Isla. Third eye crime: building a stealth game around occupancy maps. In *Proceedings of the Ninth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, page 206. AAAI Press, 2013. 1.2, 5.1

[31] Volkan Isler, Sampath Kannan, and Sanjeev Khanna. Randomized pursuit-evasion with local visibility. *SIAM j. discrete math.*, 20(1):26–41, 2006. 1.2

[32] Bruno Jedynak, Peter I. Frazier, and Raphael Sznitman. Twenty questions with noise: Bayes optimal policies for entropy loss. *Journal of Applied Probability*, 49 (1):114–136, 2012. doi: 10.1239/jap/1331216837. 1.1, 2.2, 4.1

[33] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, 03 1960. ISSN 0021-9223. doi: 10.1115/1.3662552. 3.3

[34] Kirthevasan Kandasamy, Akshay Krishnamurthy, Jeff Schneider, and Barnabas Poczos. Parallelised bayesian optimisation via thompson sampling. In Amos Storkey and Fernando Perez-Cruz, editors, *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, pages 133–142. PMLR, 09–11 Apr

2018. 1.2, 2.2, 2.3

[35] Kirthevasan Kandasamy, Willie Neiswanger, Reed Zhang, Akshay Krishnamurthy, Jeff Schneider, and Barnabas Poczos. Myopic posterior sampling for adaptive goal oriented design of experiments. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pages 3222–3232. PMLR, 2019. (document), 1.1, 1.2, 2.1, 2.2, 2.3

[36] Wei Ke, Tianliang Zhang, Zeyi Huang, Qixiang Ye, Jianzhuang Liu, and Dong Huang. Multiple anchor learning for visual object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3.2

[37] Athanasios Kehagias, Geoffrey Hollinger, and Sanjiv Singh. A graph search algorithm for indoor pursuit/evasion. *Mathematical and Computer Modelling*, 50 (9):1305–1317, 2009. ISSN 0895-7177. doi: https://doi.org/10.1016/j.mcm.2009.06.011. 5.1

[38] George Kimeldorf and Furman H. Smith. Binomial searching for a random number of multinomially hidden objects. *Management Science*, 25(11):1115–1126, 1979. 5.1

[39] A. Kleiner and A. Kolling. Guaranteed search with large teams of unmanned aerial vehicles. In *2013 IEEE International Conference on Robotics and Automation*, pages 2977–2983, 2013. doi: 10.1109/ICRA.2013.6630990. 5.1

[40] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2149–2154 vol.3, 2004. doi: 10.1109/IROS.2004.1389727. 4.4, 5.5

[41] Moshe Kress, Kyle Y. Lin, and Roberto Szechtman. Optimal discrete search with imperfect specificity. *Mathematical Methods of Operations Research*, 68(3): 539–549, Dec 2008. ISSN 1432-5217. 5.1

[42] Lanny Lin and Michael A. Goodrich. Uav intelligent path planning for wilderness search and rescue. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 709–714, 2009. doi: 10.1109/IROS.2009.5354455. 4.1

[43] Yifei Ma, Roman Garnett, and Jeff Schneider. Active search for sparse signals with region sensing. *Proc. Conf. AAAI Artif. Intell.*, 31(1), 2017. 1.1, 2.2, 4.1

[44] Yifei Ma, Roman Garnett, and Jeff Schneider. Active search for sparse signals with region sensing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017. 1.2, 5.5

[45] Ryan J Marcotte, Acshi Haggenmiller, Gonzalo Ferrer, and Edwin Olson. Prob-

abilistic multi-robot search for an adversarial target. 5.1

[46] B McCue and National Defense University Press. *U-boats in the bay of Biscay: An essay in operations analysis*. National Defense University Press, 1990. 1.2, 5.1

[47] Carla Mouradian, Jagruti Sahoo, Roch H. Glitho, Monique J. Morrow, and Paul A. Polakos. A coalition formation algorithm for multi-robot task allocation in large-scale natural disasters. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 1909–1914, 2017. doi: 10.1109/IWCMC.2017.7986575. 1.1

[48] Robin R Murphy, Satoshi Tadokoro, and Alexander Kleiner. Disaster robotics. In *Springer Handbook of Robotics*, pages 1577–1604. Springer International Publishing, Cham, 2016. 1.1

[49] Keiji Nagatani, Yoshito Okada, Naoki Tokunaga, Seiga Kiribayashi, Kazuya Yoshida, Kazunori Ohno, Eijiro Takeuchi, Satoshi Tadokoro, Hidehisa Akiyama, Itsuki Noda, Tomoaki Yoshida, and Eiji Koyanagi. Multirobot exploration for search and rescue missions: A report on map building in RoboCupRescue 2009. *J. field robot.*, 28(3):373–387, 2011. 1.1

[50] Richard Nowakowski and Peter Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2):235–239, 1983. ISSN 0012-365X. doi: https://doi.org/10.1016/0012-365X(83)90160-7. 1.2

[51] Matthew R O'Shaughnessy, Mark A Davenport, and Christopher J Rozell. Sparse bayesian learning with dynamic filtering for inference of time-varying sparse signals. *IEEE Trans. Signal Process.*, 68:388–403, 2020. 2.3

[52] Phillip Quin, Dac Dang Khoa Nguyen, Thanh Long Vu, Alen Alempijevic, and Gavin Paul. Approaches for efficiently detecting frontier cells in robotics exploration. *Front. Robot. AI*, 8:616470, 2021. 4.1

[53] Purnima Rajan, Weidong Han, Raphael Sznitman, Peter I Frazier, and Bruno M Jedynak. Bayesian multiple target localization. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, pages 1945–1953. JMLR.org, 2015. 1.1, 2.2, 4.1

[54] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. 3.2

[55] Scott D Roth. Ray casting for modeling solids. *Computer Graphics and Image Processing*, 18(2):109–144, 1982. ISSN 0146-664X. doi: https://doi.org/10.1016/0146-664X(82)90169-1. 5.2

[56] Nicholas Roy, Geoffrey Gordon, and Sebastian Thrun. Finding approximate pomdp solutions through belief compression. *J. Artif. Int. Res.*, 23(1):1–40, jan

2005. ISSN 1076-9757. 5.1

[57] Daniel Russo, Benjamin Roy, Abbas Kazerouni, and Ian Osband. A tutorial on thompson sampling. *Foundations and Trends® in Machine Learning*, 11, 07 2017. doi: 10.1561/2200000070. 2.2, 4.1

[58] A. Ryan and J.K. Hedrick. A mode-switching path planner for uav-assisted search and rescue. In *Proceedings of the 44th IEEE Conference on Decision and Control*, pages 1471–1476, 2005. doi: 10.1109/CDC.2005.1582366. 4.1

[59] Armin Sadeghi and Stephen L Smith. Heterogeneous task allocation and sequencing via decentralized large neighborhood search. *Unmanned syst.*, 05(02): 79–95, 2017. 4.1

[60] Hadi Salman, Elif Ayvali, and Howie Choset. Multi-agent ergodic coverage with obstacle avoidance. *Proc. Int. Conf. Autom. Plan. Sched.*, 27:242–249, 2017. 1.1, 4.1

[61] Hiroyuki Sato and Johannes O Royset. Path optimization for the resource-constrained searcher. *Nav. Res. Logist.*, pages NA–NA, 2010. 1.2

[62] Dieter W Schuldt and Joel A Kurucar. Efficient partitioning of space for multiple UAS search in an unobstructed environment. *Int. J. Intell. Robot. Appl.*, 2(1): 98–109, 2018. 1.1

[63] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x. 2.1

[64] Matthijs T J Spaan, Geoffrey J Gordon, and Nikos Vlassis. Decentralized planning under uncertainty for teams of communicating agents. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems - AAMAS '06*, New York, New York, USA, 2006. ACM Press. 4.1

[65] Stanford Artificial Intelligence Laboratory et al. Robotic operating system. 5.4.5

[66] Ioan A. Şucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012. doi: 10.1109/MRA.2012.2205651. (document), 1.2, 3.4, 5.4.5, 5.6

[67] Ansel Teng, Daniel DeMenthon, and L.S. Davis. Stealth terrain navigation. *Systems, Man and Cybernetics, IEEE Transactions on*, 23:96 – 110, 02 1993. doi: 10.1109/21.214770. 1.2, 5.1

[68] A.D. Tews, G.S. Sukhatme, and M.J. Mataric. A multi-robot approach to stealthy navigation in the presence of an observer. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 3, pages 2379–2385 Vol.3, 2004. doi: 10.1109/ROBOT.2004.1307417. 1.2, 5.1

[69] Vinod Thomas and Rammn LLpez. Global increase in climate-related disasters.

SSRN Electron. J., 2015. 1.1

[70] William R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933. ISSN 00063444. 4.1

[71] Michael E. Tipping. Sparse bayesian learning and the relevance vector machine. 1, 2001. ISSN 1532-4435. 2.3, 4.3.1, 5.4.1

[72] John Tisdale, Zuwhan Kim, and J. Karl Hedrick. Autonomous uav path planning and estimation. *IEEE Robotics Automation Magazine*, 16(2):35–42, 2009. doi: 10.1109/MRA.2009.932529. 5.1

[73] Michael R. Travis, Gary H. Elsner, Wayne D. Iverson, and C. G. Johnson. Viewit: computation of seen areas, slope, and aspect for land-use planning. 1975. 5.2

[74] R. Vidal, O. Shakernia, H.J. Kim, D.H. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Transactions on Robotics and Automation*, 18(5):662–669, 2002. doi: 10.1109/TRA.2002.804040. 5.1

[75] Marcos Vieira, Ramesh Govindan, and Gaurav Sukhatme. Scalable and practical pursuit-evasion with networked robots. *Intelligent Service Robotics*, 2:247–263, 10 2009. doi: 10.1007/s11370-009-0050-y. 5.1

[76] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C. J. Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, Aditya Vijaykumar, Alessandro Pietro Bardelli, Alex Rothberg, Andreas Hilboll, Andreas Kloeckner, Anthony Scopatz, Antony Lee, Ariel Rokem, C. Nathan Woods, Chad Fulton, Charles Masson, Christian Häggström, Clark Fitzgerald, David A. Nicholson, David R. Hagen, Dmitrii V. Pasechnik, Emanuele Olivetti, Eric Martin, Eric Wieser, Fabrice Silva, Felix Lenders, Florian Wilhelm, G. Young, Gavin A. Price, Gert-Ludwig Ingold, Gregory E. Allen, Gregory R. Lee, Hervé Audren, Irvin Probst, Jörg P. Dietrich, Jacob Silterra, James T. Webber, Janko Slavič, Joel Nothman, Johannes Buchner, Johannes Kulick, Johannes L. Schönberger, José Vinícius de Miranda Cardoso, Joscha Reimer, Joseph Harrington, Juan Luis Cano Rodríguez, Juan Nunez-Iglesias, Justin Kuczynski, Kevin Tritz, Martin Thoma, Matthew Newville, Matthias Kümmerer, Maximilian Bolingbroke, Michael Tartre, Mikhail Pak, Nathaniel J. Smith, Nikolai Nowaczyk, Nikolay Shebanov, Oleksandr Pavlyk, Per A. Brodtkorb, Perry Lee, Robert T. McGibbon, Roman Feldbauer, Sam

Lewis, Sam Tygier, Scott Sievert, Sebastiano Vigna, Stefan Peterson, Surhud More, Tadeusz Pudlik, Takuya Oshima, Thomas J. Pingel, Thomas P. Robitaille, Thomas Spura, Thouis R. Jones, Tim Cera, Tim Leslie, Tiziano Zito, Tom Krauss, Utkarsh Upadhyay, Yaroslav O. Halchenko, Yoshiki Vázquez-Baeza, and SciPy 1.0 Contributors. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3):261–272, Mar 2020. ISSN 1548-7105. doi: 10.1038/s41592-019-0686-2. 4.3.7

[77] Jianjun Wang, Gary Robinson, and Kevin White. Generating viewsheds without using sightlines. 66, 01 2000. (document), 5.2, 5.4

[78] D.P. Wipf and B.D. Rao. Sparse bayesian learning for basis selection. *IEEE Transactions on Signal Processing*, 52(8):2153–2164, 2004. doi: 10.1109/TSP.2004.831016. 4.3.1, 5.4.1

[79] El-Mane Wong, F. Bourgault, and T. Furukawa. Multi-vehicle bayesian search for multiple lost targets. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3169–3174, 2005. doi: 10.1109/ROBOT.2005.1570598. 5.1

[80] Nuo Xu. Active object searching on mobile robot using reinforcement learning. In *2021 2nd International Conference on Computing and Data Science (CDS)*, pages 296–300. IEEE, 2021. 1.1, 2.1, 4.1

[81] Xin Ye, Zhe Lin, Haoxiang Li, Shibin Zheng, and Yezhou Yang. Active object perceiver: Recognition-guided policy learning for object searching on mobile robots. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6857–6863. IEEE, 2018. 1.1, 2.1, 4.1

[82] Zhongshun Zhang and Pratap Tokekar. Tree search techniques for adversarial target tracking with distance-dependent measurement noise. *IEEE Trans. Control Syst. Technol.*, 30(2):712–727, 2022. 5.1

[83] Lifeng Zhou and Vijay Kumar. Robust multi-robot active target tracking against sensing and communication attacks. In *2022 American Control Conference (ACC)*, pages 4443–4450, 2022. doi: 10.23919/ACC53348.2022.9867575. 1.2

[84] Lifeng Zhou, Vasileios Tzoumas, George J. Pappas, and Pratap Tokekar. Resilient active target tracking with multiple robots. *IEEE Robotics and Automation Letters*, 4(1):129–136, 2019. 1.2

[85] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3357–3364. IEEE, 2017. 1.1, 2.1, 4.1