EFFICIENT AND LOW-COST LOCALIZATION OF RADIO SOURCES WITH AN AUTONOMOUS DRONE

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF AERONAUTICS
ASTRONAUTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Louis Kenneth Dressel
December 2018

© Copyright by Louis Kenneth Dressel 2019
All Rights Reserved

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Mykel J. Kochenderfer) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Mac Schwager)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(J. David Powell)

Approved for the Stanford University Committee on Graduate Studies

Abstract

A radio source is anything that emits radio signals. It might be a signal jammer, a cellphone, a wildlife radio-tag, or the telemetry radio of an unathorized drone. It is often critical to find these radio sources as quickly as possible. For example, if the radio source is a GPS jammer, it must be found and stopped so nearby users can continue to use GPS signals for navigation. Traditional methods for localizing radio sources are expensive and often labor-intensive. This thesis explores the use of an autonomous drone (a small aircraft) to efficiently localize a single radio source. This thesis takes a holistic approach to the problem, making contributions to both the hardware and algorithms needed to solve it.

Because drones offer a low-cost platform to quickly localize radio sources, there has been much research into drone-based radio localization. However, previous work has limitations that this thesis attempts to address. In terms of hardware, previous approaches use sensors that are either inefficient or expensive and complex. In terms of algorithms, most work uses greedy (also called myopic or one-step) optimizations to guide the drone. While these methods work well, they are generally suboptimal.

The first contributions of this thesis relate to hardware. Two sensing modalities are presented and evaluated for drone-based radio source localizaton. These modalities are simple, easily constructed, inexpensive, and leverage commercial-off-the-shelf components. Despite their simplicity, these modalities outperform sensors commonly used in prior work and are robust to radio sources with unknown or time-varying transmit power. These modalities are validated in simulation and in flight tests localizing a cellphone, a wildlife radio-tag, and another drone by its telemetry radio.

Secondly, this thesis makes contributions to the field of principled, multi-step

belief-space planning. When performing localization, the drone maintains a belief, or distribution over possible radio source locations. Its goal is to select control inputs that lead to informative sensor measurements and a highly concentrated belief, implying high confidence in its estimate of the radio source's location. This multi-step problem is cast as a partially observable Markov decision process (POMDP). This thesis expands on recent work to incorporate belief-dependent rewards in offline POMDP solvers. In this respect, the chief contribution of this thesis is an improved lower bound that greatly reduces computation. Despite this improvement, it was found that offline solvers could not scale to handle realistic scenarios. To solve the problem in real-time, an online POMDP solver based on Monte Carlo tree search is used. In simulations, this method outperforms a greedy method in a multi-objective localization problem where the seeker drone must avoid near-collisions with a moving radio source. This method was implemented in a flight test localizing another drone by its telemetry radio.

The third set of contributions made by this thesis relate to ergodic control for information gathering, in which a sensing agent selects trajectories that are ergodic with respect to an information distribution. This thesis briefly explores the conditions under which ergodic control might be optimal. Ergodic control is shown to be the optimal information gathering strategy for a class of problems which unfortunately does not include drone-based radio localization. In another contribution, it is shown how neural networks can quickly generate information maps, a key step to generating ergodic trajectories. The resulting approximations are accurate and yield orders of magnitude reduction in computation, allowing information maps to be generated in real-time. Finally, simulations are used to evaluate ergodic control in drone-based radio source localization. While the resulting performance depends on the method used to generate ergodic trajectories, ergodic control can offer modest improvements over greedy methods in nominal conditions and greater improvements in the presence of significant unmodeled noise.

Acknowledgments

Thank many people.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This thesis considers the efficient localization of a single radio source by a single autonomous drone.

A *drone* is an unmanned aircraft. Common alternative terms include "aerial robot" or "unmanned aerial vehicle" (UAV). The term "drone" includes a wide range of vehicles, including multimillion dollar military aircraft, but this thesis limits its scope to consumer drones, such as those produced by the company DJI. While this work exclusively uses a multirotor drone, many of the techniques in this thesis could be extended to other aircraft types. The drone in this work is also autonomous, meaning it plans and executes its flight without input from a pilot on the ground.

A *radio source* is something that radiates in the electromagnetic spectrum. It can be something meant to radiate, such as a radio or transmitter, or something that accidentally radiates, such as faulty electrical equipment. A variety of radio sources are used in this work, including an amateur radio, a wildlife radio-tag, and a cell phone. These sources range in frequency from about 200 MHz to 2.4 GHz, covering parts of the VHF and UHF bands. While the techniques in this thesis are designed for this frequency range, many of them can be extended to other frequencies.

To *localize* roughly means "to locate". Whereas locating implies finding an exact location, localizing implies confining to a small area. When the drone starts localizing a hidden radio source, there is a large area in which the source might reside. This space of possible source locations is reduced with successive measurements; efficient

localization reduces this space quickly and confines possible source locations to a small area.

In the context of robotics, localization often means localizing the robot itself. However, this thesis assumes the drone knows its position and orientation. This assumption is reasonable as most drones are equipped with GPS receivers, magnetometers, and other sensors. Any uncertainty in the drone's own position is ignored as it is much smaller than uncertainty in the radio source's position. It is possible the radio source interferes with GPS signals, forcing the drone to operate in a GPS-denied environment. However, the drone can use alternative positioning techniques, such as other satellite navigation systems or optical flow of the terrain. While these methods might not be as reliable as GPS, they are acceptable for a small, inexpensive drone. The specific methods of localization in GPS-denied environments is beyond the scope of this work.

The contributions of thesis aim to make drone-based radio localization efficient in time, cost, and human effort. Because a practical solution is desired, many flight tests are flown to evaluate and validate the proposed techniques.

1.1 Motivation

This work was originally funded by the Federal Aviation Administration (FAA) through the Stanford GPS Lab. The FAA's interest in rapidly localizing radio sources comes from their desire to protect aviation and the national airspace [1]. As aviation relies more heavily on GPS for precise navigation, it becomes vulnerable to disruptions of GPS. Therefore, early work aimed to rapidly localize anything radiating at the GPS frequencies and interfering with navigation solutions.

GPS is prone to interference because its signals are weak once they reach Earth. Each GPS satellite flies at an altitude of 20 000 km and radiates with 27 W of power. By the time these signals reach Earth, they are received with about 1×10^{-16} W [2]. For comparison, a cell phone radiates with about 0.1 W. Because GPS signals are so weak, they can be jammed, or overwhelmed, by any radiation in the GPS frequency band, denying navigation solutions.

This jamming is often accidental. In 1999, a camera on Stanford's campus unintentionally jammed GPS in a 1 km radius, even affecting helicopters flying to Stanford Hospital [3]. The camera transmitted pictures of a construction site to construction headquarters. The camera's designers mistakenly thought transmissions at 1570 MHz would not interfere with the GPS L1 frequency (1575.45 MHz). Using a golf cart and directional antenna, the Stanford GPS Lab found the camera and, terminating it with extreme prejudice, restored GPS to campus. In another incident from 2001, boats in Moss Landing Harbor reported a GPS outage. An investigation revealed that defective amplifiers on television antennas were accidentally radiating in the GPS frequency band [4].

Not all GPS jamming is accidental, as some criminals actively jam it for nefarious purposes. Car thieves jam GPS to cirumvent anti-theft devices that report the car's position, and some truck drivers do so to avoid GPS-based road tolling [5], [6]. A stationary jammer detection device on a three-lane highway reported 45 jamming events over 115 hours of operation [7]. Exacerbating the jamming problem, the contemporary concern for privacy has led to the proliferation of personal privacy devices [3], [8]–[10]. These small GPS jammers often affect other users and are illegal to sell or operate in many countries. Drivers with these devices have disrupted FAA GPS-based systems as they drive or park near Newark Liberty International Airport [11]. The ability to rapidly localize sources of GPS interference could mitigate the risk GPS jamming poses to aviation.

GPS interference is not the only threat to aviation, as manned aircraft are threatened by the rising popularity of consumer drones. In a three-month span in 2017, the FAA recorded 634 sightings of unmanned aircraft operating near airplanes, helicopters, and airports [12]. In 2017 the UK experienced 92 "Airprox" events in which drones compromised the safety of manned aircraft [13]. The FAA has had to warn drone pilots not to fly near wildfires, as it forces firefighting aircraft to land [14]. While it is often illegal to fly near airports, aircraft, and emergency operations, some drone pilots are unaware of the laws or ignore them.

Dangerous and illegal drone operations could be mitigated with rapid radio localization. Trespassing drones could be localized by their telemetry signals, or the drone

pilot's transmitter could be localized. Although a technically competent adversary could avoid detection by programming an autonomous path and maintaining radio silence, radio localization is useful in many scenarios and is a tool that should be available to enforcement personnel.

Rapid localization of radio sources is useful in many applications beyond protection of the national airspace. An important example is localization of radio-tagged wildlife [15]. Ecologists tag animals with radio beacons and track their movements to learn about their motion. This effort is critical to helping animals and conservation efforts. Another application is localization of avalanche beacons, where quickly localizing victims drastically improves the survival rate [16].

Existing localization techniques are expensive in time, cost, and human effort. For example, ecologists laboriously localize radio-tagged wildlife by hiking over rough terrain and manually rotating a directional antenna. A flying solution allows rough terrain to be bypassed while reducing radio reflections from obstacles on the ground [17], [18]. The FAA has proposed using small manned aircraft to localize sources of GPS interference [19]. However, a manned solution is expensive.

A drone could localize radio sources efficiently and with low cost. A low-cost, consumer drone could overfly rough terrain and ground clutter while costing much less than a manned aircraft. Drone autonomy could reduce the operational burden on researchers.

It is impossible to forsee the countless applications of drone-based radio localization that might arise in the future; a solution that is simple, low-cost, and light-weight is somewhat future-proofed. For example, the U.S. Marine Corps recently stated that infantry squads will soon include a drone operator with a small drone [20]. A low-cost, light-weight localization system could be applied to this platform or unanticipated future applications.

1.2 Related Work

Drone-based radio localization consists of many subproblems, each of which have their own, extensive literature. Detailed background for each area is presented in the

individual chapters, but this section provides a brief, holistic overview of attempts to use drones for localizing radio sources.

Perhaps the earliest work in using drones to localize radio sources was described by Gabe Hoffmann at Stanford University in 2008 [21]. This work's main contribution was a greedy, information-theoretic trajectory planner for drones localizing a stationary radio source [16]. This method is generally suboptimal but computationally efficient, so it has been used in much subsequent research [15], [22]–[25]. However, Hoffmann's flight tests were limited to a small search area ($9\,\mathrm{m} \times 9\,\mathrm{m}$) and a sensor that only worked for a specific avalanche beacon [21]. More general sensors, capable of finding other radio sources, were only simulated and not realized in hardware.

Between 2008 and 2010, significant work was done in the context of radio-tagged wildlife [18], [26], [27]. This work proposed mounting directional antennas on fixed-wing drones and using a measurement model based on signal strength. Predicting signal strength requires the radio source's transmit power, which is unknown for sources like GPS jammers. Further, signal propagation is complicated and depends on many factors, resulting in much unmodeled noise. Therefore, this modality was limited to simulations and ground tests.

Rotating a directional antenna can yield bearing estimates to a radio source without knowing the transmit power. In 2013, this method was applied to a drone that constantly rotates to keep itself airborne (inspired by maple seeds) [28]. However, this kind of drone is uncommon and difficult to control. In 2014, this constantly-rotate-for-bearing modality was applied to a conventional quadcopter, but constantly rotating the drone complicates control loops and severely limits translational speed and range [29].

In 2014, the Stanford GPS Lab began work on a drone to localize GPS jammers, with the aim of eliminating the drawbacks in previous work. We equipped a DJI S-1000 octocopter with a directional antenna. Instead of constantly rotating, the drone only rotates once to make a bearing estimate, fly normally to a new location, and rotate again for a new bearing estimate. In 2015, we demonstrated this rotate-for-bearing modality and localized a WiFi router [22]; in 2016, we localized GPS jammers in excercises hosted by the Department of Homeland Security [23]. This modality was

simultaneously developed and deployed to localize wildlife radio-tags [15], [30], [31].

This early work at the Stanford GPS lab forked into two branches. One branch has continued to focus specifically on GPS jammer localization, leading to research into beam-steering and navigation in GPS-denied environments [32]. A critical limitation of the rotate-for-bearing modality is the long time required to make a single bearing estimate [15], [31]. Beam-steering addresses this limitation and allows for near-instantaneous bearing estimates to be made by measuring the phase differences measured by an antenna array. However, beam-steering is complex and antenna arrays can be heavy, which could impede adoption in other application areas.

The research in this thesis represents the second branch, which is focused on extending early work to other applications, such as localizing radio-tagged wildlife. Therefore, simplicity and low-cost are major goals of this work. The limitations of of the rotate-for-bearing modality are addressed, including the slow measurement rate. This work also devotes significant attention to evaluating and improving the algorithms used to localize radio sources.

1.3 Contributions

This thesis presents both hardware and algorithmic solutions to challenges in drone-based localization of radio sources. These contributions covers three main areas: hardware, planning, and ergodic control.

Hardware

Hardware contributions focus on the how the drone pulls useful information from the radio waves transmitted by the radio source:

1. Two sensing modalities for drone-based radio localization are presented and evaluated; these modalities are simple and efficient, leading to fast localization.

2. It is shown how these modalities can be realized with low cost and simple electrical components.

3. These modalities are demonstrated localizing a number of radio sources, including a cell phone and a moving drone by its telemetry radio; to our knowledge, these are novel applications.

Planning

Localization is a type of information gathering task. Multi-step planning for information gathering is very difficult, so most prior work uses greedy, single-step optimizations. However, greedy solutions are generally suboptimal. This thesis frames the multi-step optimization problem as a partially observable Markov decision process (POMDP). The radio source's location is unknown, so the drone maintains a belief, or probability distribution over possible source locations. Belief-dependent reward functions can guide the drone to take informative measurements leading to concentrated beliefs, which imply confidence in the target estimate. Unfortunately, incorporating belief-dependent rewards into POMDPs is non-trivial, and this thesis makes contributions in this area. Planning contributions focus on improving and evaluating multi-step planning for information gathering tasks:

1. An improved lower bound for offline POMDP solvers with belief-dependent rewards is presented.

2. An online method is developed, analyzed in simulations, and deployed in a flight test localizing a moving radio source.

Ergodic Control

Scalable heuristic methods are another alternative to the difficulties of multi-step planning. Ergodic control is one such method that has been recently proposed in the context of information gathering. Ergodic control contributions focus on evaluation and implementation improvements:

1. The optimality of ergodic control for information gathering is explored, and ergodic control is shown to be optimal for a specific class of information gathering problems.

2. Neural networks are used to generate information maps orders of magnitude faster than directly computing them, allowing ergodic control to be performed in real-time.

3. Ergodic control is empirically evaluated for drone-based ergodic control, including simulated environments with significant unmodeled noise.

1.4 Organization

This thesis is organized as follows.

Chapter 2 presents preliminary information that appears throughout the thesis. It describes the drone and radio sources used in experiments, the models and assumptions used, and basic filtering and localization techniques.

Chapter 3 describes the problem of pulling information from radio waves emanating from a radio source. This chapter presents two sensing modalities for drone-based radio localization. These modalities are evaluated in simulation and flight tests localizing three different radio sources.

Chapter 4 explores the use of offline belief-space planning techniques for information gathering tasks, using the partially observable Markov decision process (POMDP) framework. The traditional POMDP formulation does not allow belief-dependent rewards, which is critical for information gathering tasks. This chapter describes recent work to allow these rewards and presents an improved lower bound that drastically improves computational efficiency. While an important theoretical contribution, this approach did not scale beyond simplified simulations.

Chapter 5 attempts to improve the computational efficiency of belief-space planning techniques by using more scalable online techniques. These techniques are evaluated in simulations and a flight test. These tests include localizing a moving drone by its telemetry radio.

Chapter 6 introduces ergodic control and its use in information gathering tasks. Methods for generating ergodic trajectories are briefly described. The optimality of ergodic control for information gathering tasks is explored, resulting in a class of

information gathering task under which ergodic control is optimal. This class includes important concepts like information submodularity.

Chapter 7 presents an important improvement to performing ergodic control in real-time. Ergodic trajectories require an information map that describes how information is distributed over the drone's state space. Generating this map is computationally expensive and can prevent real-time implementation. This chapter describes how neural networks can generate the maps in real-time.

Chapter 8 presents an empirical evaluation of ergodic control in drone-based radio source localization. Simulations are run to test the performance of ergodic control in the presence of unmodeled sensing noise. This noise is represented with a simplified multipath model that degrades observations made by the drone and its sensing modality.

Chapter 9 concludes the thesis and discusses avenues for future research.

Readers interested in specific contribution areas can selectively read certain chapters: Chapter 3 covers hardware contributions, Chapters 4 and 5 cover planning contributions, and Chapters 6 to 8 cover contributions related to ergodic control.

Chapter 2

Preliminaries

This chapter presents material that will be used throughout the remaining chapters. The radio sources and drone used to localize them are described. Then the basic models and assumptions are presented, along with basic filtering and localization algorithms.

2.1 Experimental Drone Platform

The methods presented in this thesis can be applied by many different types of drones and aircraft types. However, in this thesis, localization is performed by a DJI Matrice 100 (M-100) quadcopter, which DJI markets as a stable airframe for developers of drone applications. During experiments, the M-100 was both stable and easy to work with. Including its battery, the M-100 is 2.4 kg and has a max takeoff capacity of 3.4 kg, allowing for a 1 kg payload. When carrying a payload, the M-100 has a flight time of about 15 minutes. The M-100's maximum speed is 22 m/s. The M-100 has a built-in flight controller that provides low-level commands to the motors, keeping itself stable. A serial connection to the flight controller allows flight data to be queried. Position and velocity commands can be provided to the flight controller over the same link. An onboard computer provides velocity commands, which the flight controller executes while taking care of low-level motor inputs to keep the drone stable. An M-100 equipped with two Moxon antennas can be seen in Figure 2.1.

Figure 2.1: Matrice drone in flight with 782 MHz antennas mounted underneath.

The drone's onboard computer is the DJI Manifold, which retails for about $500 USD. The Manifold has 2 GB RAM and four ARM Cortex-A15 cores that clock up to 2.3 GHz. The Manifold is designed to analyze video in flight, but this research does not use video so a less expensive alternative could probably be used. For example, ODROID computers retail for under $100 USD and have been used in previous drone-based localization tasks [22], [33].

The Manifold runs Ubuntu and ROS [34]. In flight, the Manifold collects measurements from the radio sensors and queries the M-100's flight controller over a serial connection. The Manifold filters the measurements and drone position to estimate the radio source's location. The Manifold then computes and provides velocity commands to the drone's flight controller.

2.2 Radio Sources

Four radio sources are used in localization experiments. The first is a Baofeng UV-5R radio. This radio is popular with amateur radio enthusiasts and can transmit and

receive in portions of the VHF and UHF bands. The radio is set to 432.7 MHz, which is in the middle of the 70 cm amateur band (420-450 MHz). In the United States, an amateur radio license is needed to radiate in this band. The radio is set to its low power setting (1 W) and radiates constantly for the duration of a flight.

The second transmitter is a Nitehunters RATS-8 tracking collar. The collar pulses once a second at 217.335 MHz and is designed to be worn by hunting dogs so their owners can find them. While not sold as a wildlife radio-tag, it will be referred to as such in this work because it operates similarly. The 216-220 MHz band is commonly used for wildlife tracking, and wildlife transmitters typically pulse as well. However, this collar retails for $90 USD, which is a discount compared to collars sold for wildlife. Wildlife collars are ruggedized and often sell for hundreds of dollars.

The third radio source is a Samsung Galaxy S3 cell phone. To have it transmit, a voice call is placed over Verizon's LTE network. Around Stanford, this network operates in the 700 MHz band. Using a cheap software-defined-radio, the phone uplink frequency was found to be 782 MHz.

The fourth radio source is the SiK telemetry radio of a DJI F550 Flamewheel hexcopter. This radio communicates at 915 MHz with a corresponding radio attached to a ground station computer. This target drone serves as a moving radio source to be localized by the M-100 quadcopter.

Figure 2.2 shows the four radio sources.

2.3 Dynamic Models

The drone state, x_t, is modeled as a point in the special Euclidean group SE(2), meaning it consists of a 2D position and a drone heading. The radio source location $\theta_t \in \mathbb{R}^2$ only consists of a 2D position. When the radio source is stationary, the time subscript can be dropped so that θ denotes its location. Explicitly, x_t and θ_t are

$$\begin{aligned} x_t &= \left[x_t^{\mathrm{n}}, x_t^{\mathrm{e}}, h_t\right]^\top, \\ \theta_t &= \left[\theta_t^{\mathrm{n}}, \theta_t^{\mathrm{e}}\right]^\top, \end{aligned} \tag{2.1}$$

Figure 2.2: Transmitters used in experiments. From left to right: wildlife collar, Baofeng UV-5R, Samsung Galaxy S3, 915 MHz telemetry radio.

where x_t^{n} and x_t^{e} represent the north and east components of the drone position. Likewise, θ^{n} and θ^{e} represent the north and east components of the radio source position. The drone heading, denoted h_t, is measured east of north and defines the direction the drone faces.

Altitude is not included in the drone state because the drone is restricted to a constant altitude. Most antennas used for radio localization have roughly constant gain over elevation angle to the radio source, so changes in drone altitude do not yield much information about the radio source location. It is possible to use antennas that are sensitive to elevation angle, but this scheme would require enhanced antenna modeling. Such a scheme would also be vulnerable to uncertainty in the radio source's altitude. Because our goal is a simple, robust system, we adhere to the reasonable constant-altitude restriction, which is also common in previous work [15], [22], [23], [33].

The drone is assumed to have deterministic, single integrator dynamics, meaning the planar and rotational speeds are controlled directly. The control u_t applied at time t is

$$u_t = \left[\dot{x}_t^{\mathrm{n}}, \dot{x}_t^{\mathrm{e}}, \dot{h}_t\right]^\top, \tag{2.2}$$

where the dots above the state variables indicate time derivatives. Single integrator dynamics are simple and easy to control for, and they are a reasonable model in this case. The M-100 drone accepts velocity commands and has a low-level controller to execute them. A multirotor drone is also maneuverable and can change directions quickly. Of course, the drone spends some time accelerating to commanded velocities, but the approximation is not unreasonable. Further, all control schemes explored in this thesis involve re-planning, so errors due to mis-modeling or noise can be corrected.

Each control input is applied for Δt seconds, so the drone's state updates according to

$$x_{t+\Delta t} = x_t + u_t \Delta t. \tag{2.3}$$

Because the dynamics are noiseless, and the drone's current location x_t is known, a control input u_t perfectly determines the next state $x_{t+\Delta t}$.

It is assumed that the drone has perfect knowledge of its own state. The drone's magnetometer provides heading information, and GPS is used for 2D position. The noise in these sensors is neglected because any uncertainty in the drone's location is much smaller than uncertainty in the radio source location. The assumption of known drone position might seem questionable in certain applications, like when hunting GPS jammers. However, there are many alternative methods for a drone to estimate its own position, from vision to satellite navigation systems at other frequencies [23].

In some applications, the radio source is either stationary or moves so slowly with respect to the drone that its motion can be neglected. However, sometimes the radio source moves too quickly for its motion to be ignored; for example, when a seeker drone is localizing a target drone by its telemetry radio, the seeker drone cannot ignore the motion of the target when filtering.

When the radio source does move, it is assumed to move at an unknown, constant velocity. This assumption provides the filtering and estimation techniques with a simple motion model but is not overly restrictive. For example, many GPS jamming incidents have involved stationary jammers or those in cars moving along the New Jersey Turnpike [11]; a car moving along a highway can be reasonably modeled as moving at a constant velocity. A migrating animal might also be reasonably modeled

as moving at a constant velocity; so might a drone transiting airport's area of operations. Fully adversarial trajectories designed to confound searchers are beyond the scope of this work.

The rate of change in the radio source location is

$$\dot{\theta}_t = \left[\dot{\theta^{\mathrm{e}}}, \dot{\theta^{\mathrm{n}}}\right]^\top, \tag{2.4}$$

where $\dot{\theta^{\mathrm{e}}}$ and $\dot{\theta^{\mathrm{n}}}$ are the constant velocity components in the east and north directions. Similar to the drone state update, the radio source location update is

$$\theta_{t+\Delta t} = \theta_t + \dot{\theta}_t \Delta t. \tag{2.5}$$

Of course, when the radio source is stationary, $\dot{\theta^{\mathrm{e}}} = \dot{\theta^{\mathrm{n}}} = 0$.

2.4 Sensor Models

Beyond its physical implementation, a sensing modality requires a sensor model for filtering and estimation. A sensor model is a probabilistic model that defines the probability of making measurement z_t at time t if the drone state is x_t and the radio source location is θ_t. This probability is denoted $P(z_t \mid x_t, \theta_t)$ and is used with Bayes' rule to update the distribution of possible radio source locations. The set of possible observations is denoted $\mathcal{Z}$. Measurements are assumed to be received every Δt seconds, matching the rate at which commands are given to the drone.

Bearing and relative bearing will play an important role in the sensing modalities. The bearing β_t is defined as the angle, measured east of north, of a ray pointing from the drone position to the position of the radio source:

$$\beta_t = \arctan\left(\frac{\theta_t^{\mathrm{e}} - x_t^{\mathrm{e}}}{\theta_t^{\mathrm{n}} - x_t^{\mathrm{n}}}\right). \tag{2.6}$$

We define the quantity $\beta_t - h_t$ as the relative bearing. The relative bearing is 0° when the front of the drone points directly at the radio source.

2.5 Beliefs and Filtering

The radio source location θ_t is unknown, so the drone maintains a distribution over possible radio source locations. This distribution is called the belief, and the belief at time t is denoted b_t.

Gaussian distributions are commonly used to represent the belief in aerospace applications, with filtering handled by some variant of the Kalman filter [35]. Because Gaussians are parametric representations, they are easy to represent and update. However, Gaussians are only appropriate if the underlying distributions are unimodal and roughly Gaussian; Kalman filters are only appropriate if the dynamic and sensing models are linear or easily linearized.

The sensing modalities presented in the next chapter can lead to strongly non-Gaussian beliefs, so this thesis uses two non-parametric belief representations. For stationary radio sources, a discrete Bayes' filter is used. For moving radio sources, a particle filter is used. In both representations, the search area is modeled as a square. Other shapes could be used, but there are no compelling reasons to use one for general testing purposes. The belief is initialized as a uniform distribution, meaning the radio source is equally likely to be at any location in the search area.

2.5.1 Discrete Bayes' Filter

A discrete Bayes' filter is sometimes called a histogram filter or just a discrete filter [36], [37]. In a discrete Bayes' filter, the search area is split into a grid, where the density of each grid cell represents the probability that the radio source is in that cell. It is common in drone localization because it can handle non-Gaussian priors and non-linear dynamic and measurement models [15], [22], [23].

The belief b_t is computed from the preceding belief $b_{t-\Delta t}$, the drone state x_t, the observation z_t, the measurement model, and Bayes' rule. If the radio source is stationary, the update simplifies to

$$b_t(\theta_i) \propto b_{t-\Delta t}(\theta_i) P(z_t \mid x_t, \theta_i), \tag{2.7}$$

where θ_i is a cell and $b_t(\theta_i)$ represents the probability the radio source is in cell θ_i. For simplicity, the center of a cell is used in the measurement model. The belief is always normalized so the sum of probabilities for all cells sums to one.

Discrete filters are simple and intuitive. If a grid cell has a probability of 0.1, there is a 10% chance the jammer is in that cell (assuming the measurement models are correct).

2.5.2 Particle Filter

Discrete filters have two drawbacks when tracking moving targets. First, they become computationally slower. The number of operations per update is the square of the number of grid cells, as compared to just the number of grid cells when the target is stationary. Second, the discrete filter requires any target motion to fit neatly into its organized set of cells; target motion must be modeled as the probability of traveling from one grid cell to another. Therefore, a particle filter is used when using a non-stationary radio source, so a simple motion model for the radio source can be used.

When using a particle filter, the belief is represented as a set of particles. Each particle is a hypothesis, or a possible radio source location and velocity. After each time step, a particle's location is updated according to its velocity. The velocity remains constant, because the radio source is assumed have a constant velocity.

Each particle has a weight describing how likely that particular hypothesis is. Once the particle's location is updated according to its velocity, this weight is updated with the measurement model. If the received measurement matches the measurement that would be expected from the particle location, the particle weight remains high. If the received measurement is unlikely, then the particle weight decreases.

2.6 Greedy Information-theoretic Localization

In the context of planning, greedy or myopic solutions optimize only for the next time step. Because they focus on short-term gain, they might lead to poor long-term performance and are generally suboptimal. However, optimizing for the next time

step is much easier than optimizing over a long (possibly infinite) time horizon, so greedy optimizations are often used in robotics [16].

In the context of localization tasks, greedy optimizations aim to minimize the belief uncertainty at the next time step. One measure of uncertainty is entropy, which captures the spread of a probability distribution. The entropy of discrete distribution b_t, denoted $H(b_t)$, is

$$H(b_t) = -\sum_{\theta_i \in \Theta} b_t(\theta_i) \log b_t(\theta_i), \tag{2.8}$$

where by convention $0 \log 0 = 0$. Entropy is minimized when the probability is concentrated in a single cell and maximized when all cells have equal probability.

This section shows how a drone can use greedy entropy minimization to localize a radio source. For simplicity, this section only considers a stationary radio source and use of the discrete filter, although it is trivial to expand to particle filters and moving radio sources.

At time t, the drone makes measurement z_t, yielding the current belief b_t. The drone picks an action so that the expected uncertainty in belief $b_{t+\Delta t}$ is as small as possible. In order to simplify the optimization, the drone reasons over a discrete set of possible control actions. Each action is evaluated by the expected reduction in belief entropy after taking a specific action and making a new measurement. The action set is the Cartesian product of the velocity and heading command sets. As an example, if the velocity set consists of eight actions: move 5 m/s in one of eight directions spaced 45° apart (north, northeast, etc.); and the heading set consists of three actions: rotate 10 °/s in either direction or do not rotate; then the total action set consists of 24 actions.

To evaluate an action u_t, the drone considers the resulting state $x_{t+\Delta t}$, which is fixed by knowledge of x_t and the deterministic dynamic model. The measurement received at this new state, $z_{t+\Delta t}$, will lead to a new belief $b_{t+\Delta t}$. In myopic entropy reduction, the objective is to minimize $E_{z \in \mathcal{Z}} H(b_{t+\Delta t})$, the expected entropy after taking measurement $z_{t+\Delta t}$ at the next step. The objective is equivalent to $H(b_t \mid z_{t+\Delta t})$, the conditional entropy between distribution b_t and $z_{t+\Delta t}$. This conditional entropy expresses what the uncertainty in the radio source location would be if $z_{t+\Delta t}$

were known. We treat $z_{t+\Delta t}$ as a random variable because it is an unknown future quantity—as opposed to $x_{t+\Delta t}$, which is specified by u_t.[1] Conditional entropy can be expanded:

$$H(b_t \mid z_{t+\Delta t}) = H(b_t) - I(z_{t+\Delta t}; b_t), \tag{2.9}$$

where $I(z_{t+\Delta t}; b_t)$ is the mutual information between the target and sensor distributions. The entropy of the current belief, $H(b_t)$, cannot be changed, so maximizing the mutual information $I(z_{t+\Delta t}; b_t)$ minimizes the posterior entropy. This result satisfies intuition, as the mutual information between $z_{t+\Delta t}$ and b_t expresses the reduction in uncertainty of belief b_t if we knew $z_{t+\Delta t}$ [38]. Mutual information is symmetric so $I(z_{t+\Delta t}; b_t) = I(b_t; z_{t+\Delta t})$. Re-expanding leads to

$$I(b_t; z_{t+\Delta t}) = H(z_{t+\Delta t}) - H(z_{t+\Delta t} \mid b_t). \tag{2.10}$$

The drone evaluates Equation (2.10) for each possible action u_t and corresponding future state $x_{t+\Delta t}$, selecting the maximizing action.

Breaking down the terms in Equation (2.10) provides an intuitive understanding of greedy entropy minimization [16]. The term $H(z_{t+\Delta t})$ represents uncertainty in $z_{t+\Delta t}$, the measurement to be received at the next state $x_{t+\Delta t}$. We want this term to be large; intuitively, we learn when we sample from outcomes we are unsure of. The term $H(z_{t+\Delta t} \mid b_t)$ represents the uncertainty $z_{t+\Delta t}$ would have if the radio source's location were known. The drone is evaluating an action u_t so $x_{t+\Delta t}$ is known. If the radio source's location is also known, any uncertainty in the measurement to be received is due to sensor noise. We might learn when we sample from outcomes we are unsure of, but not if the outcomes are very noisy. Therefore, picking $x_{t+\Delta t}$ to maximize $H(z_{t+\Delta t}) - H(z_{t+\Delta t} \mid b_t)$ is equivalent to picking $x_{t+\Delta t}$ such that the drone is uncertain about which measurement it will receive, but not simply because of sensor noise.

The objective function in Equation (2.10) can be computed using the current

[1] I abuse notation and use b_t as an argument to information-theoretic quantities, even though it is a distribution and not a random variable. When we do so, we imply a random variable describing the radio source location and having distribution b_t.

belief b_t, knowledge of $x_{t+\Delta t}$, and the measurement model. Consider the first term $H(z_{t+\Delta t})$:

$$H(z_{t+\Delta t}) = -\sum_{z \in \mathcal{Z}} P(z_{t+\Delta t} = z) \log P(z_{t+\Delta t} = z), \tag{2.11}$$

Because $x_{t+\Delta t}$ is implied by u_t, we can write $P(z_{t+\Delta t} = z) = P(z_{t+\Delta t} = z \mid x_{t+\Delta t})$. The measurement model depends on radio source location, so we apply the laws of total and conditional probability:

$$P(z_{t+\Delta t} = z \mid x_{t+\Delta t}) = \sum_{\theta_i \in \Theta} P(z_{t+\Delta t} = z \mid x_{t+\Delta t}, \theta_i) b_t(\theta_i). \tag{2.12}$$

The second term in the objective from Equation (2.10) is $H(z_{t+\Delta t} \mid b_t)$:

$$H(z_{t+\Delta t} \mid b_t) = -\sum_{\theta_i \in \Theta} b_t(\theta_i) \sum_{z \in \mathcal{Z}} P(z_{t+\Delta t} = z \mid x_{t+\Delta t}, \theta_i) \log P(z_{t+\Delta t} = z \mid x_{t+\Delta t}, \theta_i). \tag{2.13}$$

Chapter 3

Sensing Modalities

Before considering the planning problem, sensing modalities must be evaluated and selected. A sensing modality describes how information is pulled from the radio waves emanating from the radio source. In some work, abstract modalities are used, and it is assumed that range or bearing estimates will be provided. However, pulling these estimates from radio waves can be difficult. Because this work aims to demonstrate localization in flight tests, concrete and realizable modalities are required.

In this chapter, two sensing modalities for drone-based radio localization are presented and evaluated. The physical implementations, mathematical models, and flight test validations for each are presented. Simulations compare these modalities to each other and to prior methods. The modalities presented here are efficient yet simple, low-cost, and lightweight. They are a significant improvement over existing techniques, which are briefly discussed in the next section.

3.1 Related Work and Motivation

Early work in radio sensing for mobile robots was motivated by localization in WiFi networks. Many approaches relied on creating signal strengh maps [39]. A mobile robot can make such a map by moving around an environment and recording the strength of signals from a WiFi router. Because radio emissions are strongest at the source, a radio strength map provides an estimate of the radio source's location.

This mapping technique was used on a drone designed to localize radio-tagged sturgeon [40]. However, mapping signal strengths over a large area is inefficient, as the robot must traverse the entire search area. Signal strength mapping also restrictively assumes the radio source is stationary and transmits with constant power.

Another widely used method in early WiFi localization relied on strengh modeling instead of strengh mapping [39], [41], [42]. The expected signal strength can be computed from the radio source's transmit power, a possible source location, and a signal propagation model. This is equivalent to correlating measured strength with distance to the radio source. By comparing this expected strength with the measured value, the radio source location estimate can be updated.

A later variant of the strength modeling modality used a directional antenna on the mobile robot [43], [44]. A directional antenna measures different strength values depending on its orientation to the radio source. The description of how antenna gain changes with orientation to the source is called the gain pattern. When the strength model includes this gain pattern, strength measurements provide information about orientation to the radio source. The directional strength modeling modality was proposed for drones localizing radio-tagged wildlife [18], [26], [27].

Strength modeling modalities have two disadvantages. First, they assume the transmit power of the radio source is known. This assumption holds when tracking known wildlife radio-tags, but not when searching for adversarial transmitters such as GPS jammers. Second, strength models suffer from significant unmodeled noise, and it is difficult to model radio wave propagation and correlate measured signal strength with distance. A number of works show the high unmodeled noise affecting strength measurements [45], [46]. Further, the transmitting antenna is assumed to be omnidirectional, but these antennas often have imperfections and are not truly omnidirectional [47]. Depending on the radio source's orientation, measured signal strength may be different, even if the distance to the receiver is the same.

These disadvantages can be mitigated by using a series of strength measurements to estimate the bearing to the radio source, instead of directly using individual measurements. One such method uses the gradient in signal strength; if signal strength

increases as a robot is moving, it is likely moving towards the radio source. However, accurate bearing estimates require the robot to move in special, inefficient patterns [48], [49]. As a result, this gradient modality is rarely used on drones, though it has been used in the context of localizing wildlife radio-tags [33]. One benefit of using the strength gradient is that the transmitter strength can be unknown, as it affects all measurements equally and is effectively normalized out by the gradient. However, the radio source's radiated power must remain constant while making the strength measurements used to calculate the gradient. Otherwise, it is impossible to correlate changes in signal strength with bearing to the source. Even if the transmitter strength remains constant, time-varying effects can affect the radiated power. A radio-tagged animal that changes its orientation or position while strength measurements are collected will make it impossible to compute a meaningful gradient.

Another way to estimate bearing from a series of strength estimates is to rotate a directional antenna in place. Directional antennas provide the highest gain when pointed at the radio source. Strength measurements are made as the antenna is rotated, and the heading with the highest strength measurement is estimated to be the bearing to the source. An actuator can be added to a mobile robot to constantly rotate the directional antenna, and this technique has been applied on a variety of mobile robots [50], [51]. However, mounting an extra actuator is unsuitable for weight-constrained vehicles like drones, so an alternative is to have the drone constantly rotate instead. One option is to use a drone that must rotate to keep itself airborne, although such vehicles are rare and difficult to control [28]. Another option is to use a common multirotor and have it constantly rotate as it flies, but constantly rotating complicates control of the drone and significantly limits its translational speed [29]. Therefore, most current work uses multirotor drones that only rotate in place to make a bearing measurement, and then translate normally [15], [22], [23], [30], [31].

There are two main disadvantages to this rotate-for-bearing modality. The first is speed. Rotations are reported to take 25 s [22], 40 s [30], or even 45 s [15]. Small drones typically have battery lives of 10–15 minutes, so a 45 s rotation could amount to nearly 8% of available flight time. Spending so much time for a single measurement limits the number of measurements that can be made. It also slows down localization,

which needs at least a few bearing estimates for a decent source position estimate.

The second drawback of rotate-for-bearing is that, like the gradient modality, it assumes there are no time-varying factors affecting measured signal strength. These factors might affect each measurement made during the rotation differently, making it impossible to know if a strong measurement was received because the antenna pointed at the radio source or because the radio source happened to be stronger at that moment. Therefore, there can be no time-varying factors affecting signal strength while the drone is performing each rotation.

A heavy, complex solution to these challenges is to use an array of antennas to measure bearing instantly or near-instantly. In beam-steering, phase shifters allow a measurement beam to be electronically rotated near-instantly, allowing the bearing to the radio source to be estimated in real-time. However, beam-steering can be heavy, as it requires an array of antennas. Beam-steering also requires electrical engineering knowledge, custom circuits and electronics, and careful calibration [51]. While beam-steering has been proposed for drones [52] and has been experimentally validated on a drone hunting GPS jammers [32], its complexity might limit its adoption in other fields. An alternate array-based method uses the strength measured by an array of four well-modeled directional antennas [53], [54]. While electronically simpler than beam-steering, the weight requirement is more onerous—some drones simply cannot carry four directional antennas, and this method has not been applied on drones. A final option is to use commercial direction finding units, but they are not made for drones and have performed poorly in flight tests [55], [56].

As this section shows, previous work on sensing modalities has critical limitations. The modalities in this chapter were designed to overcome these limitations, with the specific goals of:

1. providing measurements more quickly than rotating in place;
2. not assuming the transmit strength of the radio source is known;
3. not assuming transmit strength remains constant;
4. being simpler than beam-steering and not requiring more than two antennas.

3.2 Modality Overview

The two sensing modalities presented and evaluated in this chapter are similar. This section describes the general concept behind both sensing modalities.

3.2.1 System Architecture

Both modalities are based on the principle of measuring signal strength simultaneously with two antennas carried by the drone. The strengths measured by the two antennas are compared to each other to prdouce a bearing-like measurement. These measurements are less informative than true bearing measurements, but can be made as quickly as the electronics can sample.

The basic setup for both modalities is shown in Figure 3.1. Each antenna is connected to a radio sensor that measures the signal strength for its antenna. These radio sensors are connected to the drone's onboard computer, which compares the strength measurements and performs filtering and path planning.

In one of the modalities, both antennas are slightly directional. If the front-facing antenna measures a higher strength than the rear-facing antenna, the radio source likely lies in front of the drone. In the other modality, one of the antennas is directional and the other is omnidirectional. The omnidirectional antenna cancels out unknown factors like distance to and transmit strength of the radio source, allowing the gain

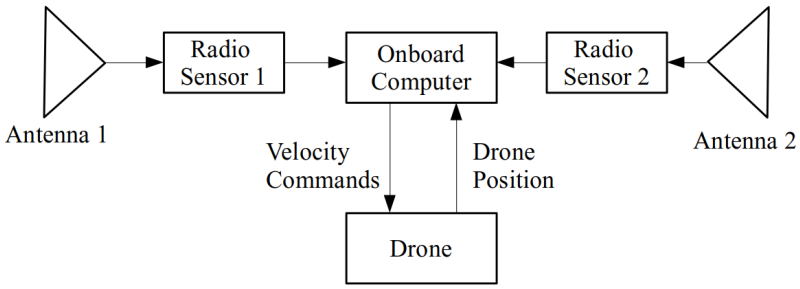

Figure 3.1: Both modalities consist of two antennas and two radio sensors. The radio sensors measure the strength received at each antenna.

contributed by the directional antenna to be estimated. Because the gain contributed by a directional antenna is a function of its orientation relative to the transmitter, this method provides a rough bearing estimate.

Because measurements are made simultaneously and compared, unknown factors affecting the strength measurements cancel out. Therefore, these modalities can handle unkown or time-varying transmit strength, as experiments will show.

3.2.2 Radio Sensing Hardware

Measuring signal strength at an antenna is a critical part of the sensing modalities presented in this chapter. Both modalities use the same radio sensors. The radio sensors used are commercial-off-the-shelf (COTS) software-defined radios (SDRs). They are easy to use, low-cost, lightweight, and allow for rudimentary spectrum analysis. There are many COTS SDRs that can be used to measure the signal strength at an antenna, and this work considers two: the RTL-SDR V3 and the HackRF One.

The RTL-SDR V3 is low-cost ($20 USD) and lightweight (30 g). RTL-SDR refers to any SDR based on the RTL2832U chipset. These chipsets were originally designed to receive television broadcasts, but they can be modified into SDRs if combined with a tuner. The RTL-SDR V3 uses the Rafael Micro R820T tuner. Each RTL-SDR V3 has a female SMA connector on one end and a USB connector on the other. Figure 3.2 shows two RTL-SDR V3s plugged into the drone's onboard computer.

While the RTL-SDR is both low-cost and lightweight, it has two main drawbacks. The first is that the R820T tuner has a maximum frequency of 1766 MHz. This upper limit covers most radio sources of interest, including wildlife radio-tags, the GPS frequency, ADS-B, and most cellular bands. But some commonly used frequencies, like WiFi (2.4 GHz), are outside this range. The second drawback is that the narrow bandwidth of 2.4 MHz struggles to capture emissions from a frequency-hopping spread spectrum radio source, such as a drone telemetry radio. The specific telemetry radio used in this thesis hops over a range of 26 MHz, so a receiver with narrow bandwidth will miss many emissions. It might be possible to quickly shift the center frequency of the RTL-SDR to capture more emissions, but that is left for future work.

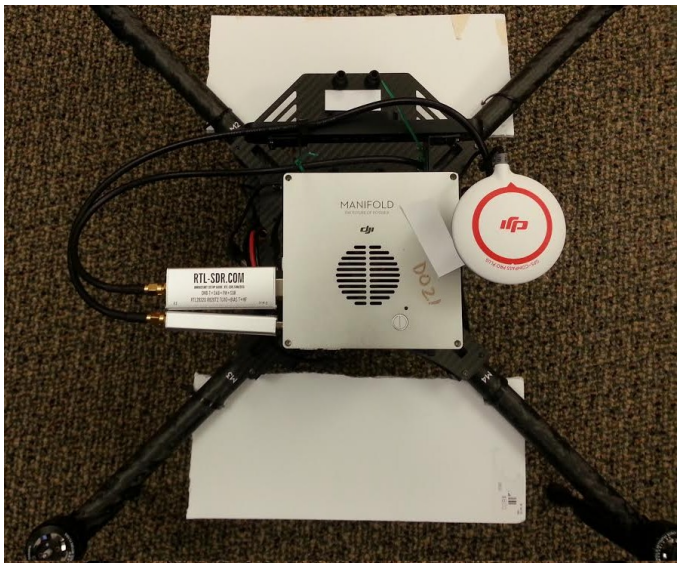

Figure 3.2: The Manifold onboard computer (center) has two RTL-SDR V3s in its USB ports (left). Each SDR is plugged into an antenna. The antennas (432.7 MHz in this picture) lie against the underside of styrofoam board.

Table 3.1: Comparing the two SDRs used in this work.

	RTL-SDR V3	HackRF One
Price	$20 USD	$300 USD
Mass	30 g	70 g
Lower Frequency Limit	0.5 MHz	1 MHz
Upper Frequency Limit	1766 MHz	6000 MHz
Bandwidth	2.4 MHz	20 MHz

A simple solution to the limited bandwidth and upper frequency limit is to use another SDR, like the HackRF One. At $300 USD, it is an order of magnitude more expensive than the RTL-SDR. However, the HackRF can reach up to 6 GHz, allowing it to track emissions at 2.4 GHz and 5.8 GHz, which are commonly used for drone videos in flight. Further, the HackRF has a bandwidth of 20 MHz.

Open-source C and Python libraries make it easy to use the RTL-SDR V3 and HackRF One. To measure signal power, the SDR is tuned to a frequency of interest. The SDR then reads radio samples, which can be converted into a periodogram, or an estimate of the power spectral density. The signal strength estimate is simply the

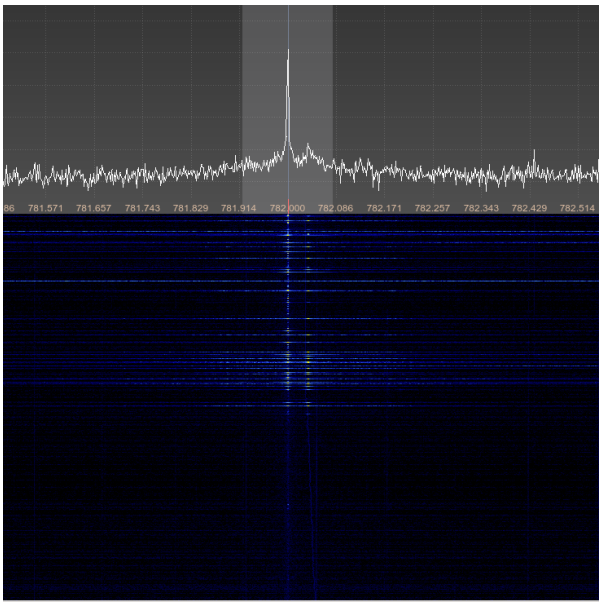

Figure 3.3: Using an RTL-SDR V3 with open-source gqrx radio software to analyze emissions from cell phone placing voice call over LTE connection at 782 MHz. The lower half of the waterfall plot corresponds to time before the call is placed; once the call is placed, emissions are logged.

largest density in the sampled bandwidth.

Each SDR can analyze a chunk of spectrum—equal to its bandwidth—at once, serving as a makeshift spectrum analyzer [57], [58]. Thus, the measurement device doubles as a troubleshooting or exploratory device. For example, cell phones radiate at many frequencies and it is not always clear which frequency is used in which region. An SDR can check the spectrum for emissions, a technique used to determine the operating frequency of the cell phone used in this work. As shown in Figure 3.3, emissions were found when an RTL-SDR V3 was tuned to 782 MHz, suggesting the cell phone operates in this band.

3.3 First Modality: Directional-Omni

In the first modality, one of the antennas is a directional antenna and the other is omnidirectional. The omnidirectional antenna is insensitive to changes in orientation with respect to the transmitter. As a result, the strength measured by the omnidirectional antenna captures the effects of transmitter power and distance to the transmitter. The directional antenna is affected not only by those factors but also the orientation to the transmitter. Because the distance to the radio source and the transmit power of the radio source are unknown, it is impossible to separate these factors from the effects of orientation. By subtracting the strength measured by the omnidirectional antenna from that measured by the directional antenna, factors like distance and transmit power are canceled out, leaving the orientation effect. It is then possible to estimate a range of possible bearings to the radio source.

3.3.1 Mathematical Model

Theoretical justification for the normalization process follows from antenna theory. It extends similar derivations in the localization literature [59]. The power P_{dir} received by a directional antenna is

$$P_{\text{dir}}(d) = \frac{P_t G_t G_{\text{dir}} \lambda^2}{(4\pi)^2 d^2 L}, \tag{3.1}$$

where P_t is the transmitter power, G_t is the transmitter antenna gain, G_{dir} is the directional antenna gain, L is a system loss factor, λ is the wavelength of the radio signal, and d is the distance between the transmitter and receiver. Received power is often expressed in dB:

$$10 \log P_{\text{dir}}(d) = 10 \log \frac{P_t G_t \lambda^2}{(4\pi)^2 d^2 L} + 10 \log G_{\text{dir}}. \tag{3.2}$$

The first term on the right-hand side of Equation (3.2) captures the effects of various factors on the measurement. Without loss of generality, this term can be denoted $P_f(d, P_t)$, ignoring effects other than distance and transmitter power. The second

term on the right-hand side is the directional antenna gain in dB and is denoted $g_{\text{dir}}(\beta)$, where β is the relative bearing from the receiving antenna to the transmitter.

The left-hand side of Equation (3.2) is the power received by the directional antenna in dB. Denoting this term s_{dir} yields a simplified power equation:

$$s_{\text{dir}} = P_f(d, P_t) + g_{\text{dir}}(\beta). \tag{3.3}$$

Equation (3.3) shows that strength measurements differ from the directional antenna gain by the factor $P_f(d, P_t)$, the unknown scale factor that requires normalization.

Normalization can be carried out by adding an omnidirectional antenna. The power received by an omnidirectional antenna is

$$s_{\text{omni}} = P_f(d, P_t) + g_{\text{omni}}, \tag{3.4}$$

where g_{omni} is the antenna's gain. This gain is independent of bearing to the radio source and typically known a priori.

If both antennas are colocated and measure simultaneously, the distance d, the transmitter power P_t, and the scale factor $P_f(d, P_t)$ will be the same for both antennas. By inserting Equation (3.4) into Equation (3.3), this scale factor can be eliminated:

$$g_{\text{dir}}(\beta) = s_{\text{dir}} - s_{\text{omni}} + g_{\text{omni}}. \tag{3.5}$$

Equation (3.5) shows that the gain contributed by the directional antenna can be estimated from the omnidirectional gain and the power measured by both antennas.

3.3.2 Physical Implementation

This modality requires one directional antenna and one omnidirectional antenna. The horizontal gain patterns of both antennas must be well characterized to carry out the normalization described in the previous subsection; Equation (3.5) relies on knowing the directional gain as a function of the bearing to the target. Ideally, the omnidirectional antenna should have no variation in gain as a function of bearing. Because high-fidelity characterization is needed, commercial antennas are used.

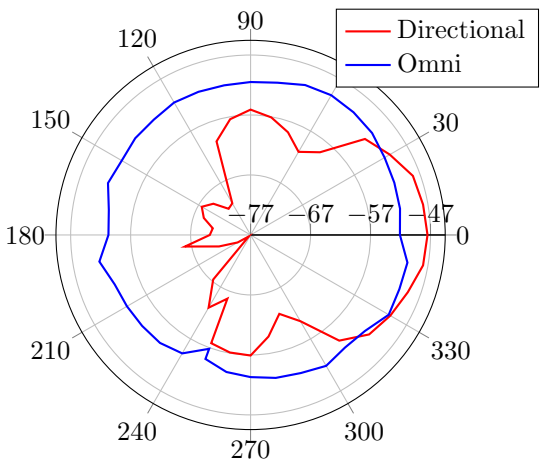

Figure 3.4: The mean power measurements made at a distance of 30 feet from the router. The omnidirectional antenna's gain is fairly constant.

The test frequency was 2.4 GHz. The directional antenna used in this work is a 9 dBi Yagi-Uda antenna (L-com model HG2409Y-RSP). It has a 60° beamwidth in the horizontal plane. The beamwidth is the angular width over which the gain is at least half (i.e., within 3 dB) of its highest value. This antenna costs $30 USD.

The omnidirectional antenna used is a 5 dBi rubber duck antenna (L-com model HG2405RD-RSP). This antenna is omnidirectional in the horizontal plane, which is the plane of interest because a multirotor drone rotates in this plane. In the vertical plane, the antenna has a large beam-width of 120°. A large vertical beamwidth is desirable because the radio source and drone will not be at the same altitude. This antenna costs $10 USD.

To test the setup, experiments were run on the ground. The antennas were rotated in place and signal strength was measured. Ten measurements per antenna were made at each 10° interval, allowing the construction of mean gain patterns for each antenna. Figure 3.4 shows patterns obtained 30 feet from the router. The power measured by the omnidirectional antenna is roughly constant, as expected.

The normalization was performed by applying Equation (3.5) to the mean gain patterns for each antenna. Figure 3.5 shows the normalization results. The unnormalized directional gain patterns are all similar, but differ greatly by a scale factor.

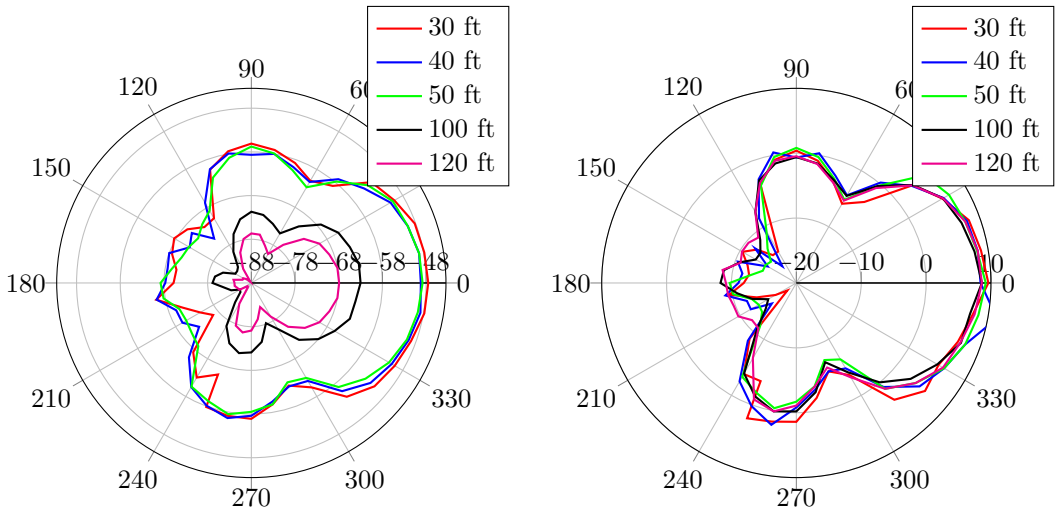

Figure 3.5: Strength measurements made by the directional antenna yield similar but scaled patterns depending on distance (top). This scale factor is eliminated with the use of the omnidirectional antenna, resulting in the gain induced by the directional antenna (bottom). The peak directional gain is roughly 9 dB at all distances, which is the nominal value for our antenna.

The normalized patterns do not differ by this scale factor. Furthermore, all normalized patterns have a peak gain of roughly 9 dBi and a beam-width of roughly 60°, matching manufacturer-provided values for the Yagi. The similarity of the normalized patterns to each other and the nominal values validates the proposed normalization procedure.

3.3.3 Flight Tests

Flight tests are necessary to discover any flight-induced effects on the sensing modality. For example, it was found that naively mounting the antennas to the drone led to poor results. If the omnidirectional antenna was mounted on one side of the drone body, the drone body would diminish signals reaching the antenna. This anisotropy ruins the normalization procedure, which assumes the omnidirectional antenna is truly omnidirectional. The solution to this was to hang the omnidirectional antenna.

The drone performed several rotations in place at 15 °/s while collecting and normalizing measurements to generate gain patterns. Figure 3.6 shows two patterns

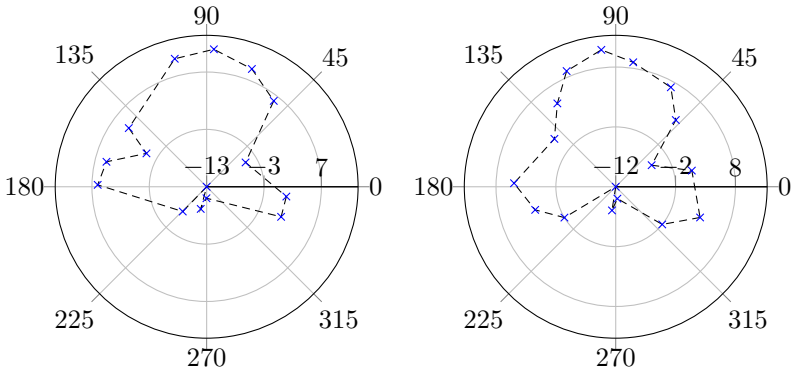

Figure 3.6: Two example patterns at a range of 40 meters and relative bearing of roughly 90° to the router.

measured in flight. The patterns are visually similar to the ground-based patterns in Figure 3.5—the side and main lobes are present, and the maximum gain is near the nominal value of 9 dB. The patterns are relatively sparse, making a more analytical comparison between the ground and airborne patterns difficult. However, the ground and airborne patterns are visually similar enough to validate aerial normalization.

Once the patterns were validated, three localization flight tests were flown in a 110 m × 110 m search area. After takeoff, the drone flew a fixed path at 10 m/s and constantly rotated at 15 °/s. This path was chosen so the resulting measurements would have good geometric diversity. The sensor model used in the filtering and estimation had a conservatively large standard deviation of 6 dB.

At the end of each flight, the Euclidean error of the mean target estimate was under three meters. Figure 3.7 shows the results of one flight test. The entire flight took 50 seconds, whereas previous bearing estimation methods reportedly spent 45 seconds [15] or 24 seconds [22] for a *single* bearing estimate. Despite the simple trajectory and low sampling rate, the flight tests validate the pseudo-bearing concept.

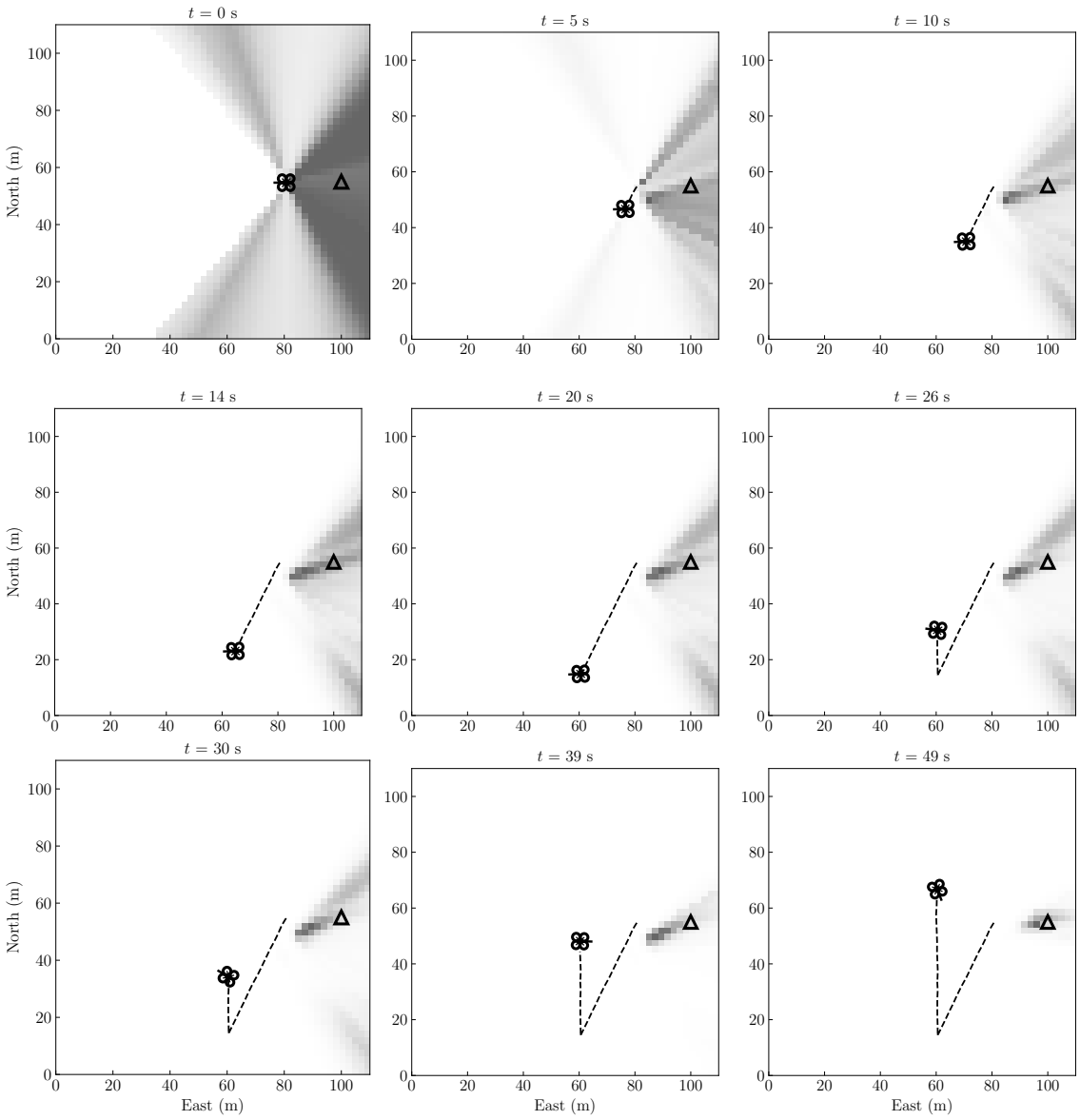

Figure 3.7: Beliefs and drone positions during a flight test with the directional-omni modality. The router (triangle) is effectively localized. The dashed line shows the path flown.

3.4 Second Modality: Double-Moxon

The modality described in the previous section worked well but has a critical flaw: it requires well-modeled antennnas with known gain patterns. If the directional antenna's gain pattern is innaccurate or unknown, or if the omnidirectional antenna is not truly omnidirectional, normalization will fail to produce reasonable measurements. It can be hard to find commercially available antennas that are sufficiently omnidirectional at certain frequencies, and it might be difficult for a user to construct their own antennas carefully enough to be omnidirectional.

This section describes a second modality designed to be overcome the modeling drawback. This modality uses two Moxon antennas to estimate a rough direction to the radio source. If the front-facing antenna measures higher strength, the radio source likely lies in front of the drone; if the rear antenna measures higher strength, the radio source likely lies behind the drone. While each measurement is less informative than in the directional-omni modality, the system as a whole is robust to errors in antenna models, and the antennas are easy to make.

3.4.1 Physical Implementation

Because this modality only discriminates between front and back, only slightly directional antennas are needed. In localization tasks, highly directional antennas are often preferred because they concentrate gain in a smaller beamwidth, leading to better directional discrimination. However, highly directional antennas can be large and unsuitable for small drones. For example, one way to increase the directionality of a Yagi antenna is to add elements, yielding a longer, heavier antenna. This concern is especially relevant at lower frequencies, as antennas scale inversely with frequency.

To limit antenna size, only "slightly directional" antennas are used. Specifically, Moxon antennas are used, which are similar to Yagi antennas with only two elements [60]. Moxon antennas are popular in the amateur radio community because they are easy to build and mechanically robust. They have low directionality, with most of their gain concentrated in a wide main lobe.

Design of these antennas requires no specialized electrical engineering knowledge.

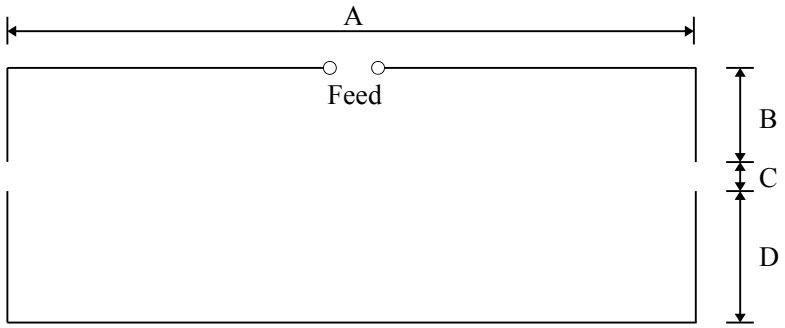

Figure 3.8: Top view of a basic Moxon antenna. Feed side points forward.

Table 3.2: Antenna sizes produced by Moxon generator [61] for different frequencies and 14 AWG copper wire. Lengths A, B, C, and D correspond to those from Figure 3.8. Mass includes coax cable.

Frequency (MHz)	A (cm)	B (cm)	C (cm)	D (cm)	Mass (g)
217.335	49.81	6.84	2.07	9.51	118
432.7	24.88	3.26	1.20	4.79	70
782.0	13.71	1.71	0.75	2.66	66

Because Moxon antennas are popular with amateur radio enthusiasts, there exist many free Moxon design generators. These tools interpolate between several well-modeled Moxon antennas for a range of frequencies and wire diameters. We use one such Moxon generator [61], inputting only frequency and wire diameter. Table 3.2 shows the resulting antenna sizes for different frequencies. The Moxon generator also produces an input file for NEC-2, an antenna analysis tool developed for the U.S. Navy [62]. Now in the public domain, NEC-2 can be used to tweak Moxon designs further.

Construction is also trivial and requires no mechanical skill beyond rudimentary soldering. The antenna dimensions are drawn on styrofoam board. Copper wire is bent and cut to fit the dimensions. The wire is taped to the board, and a thin cut is made in the upper wire for the feed. A suitable length of RG-58 coaxial cable is selected (about half a meter) and an inch of the cable is stripped of its inner and outer insulation. The inner conductor is soldered to one side of the copper feed, and

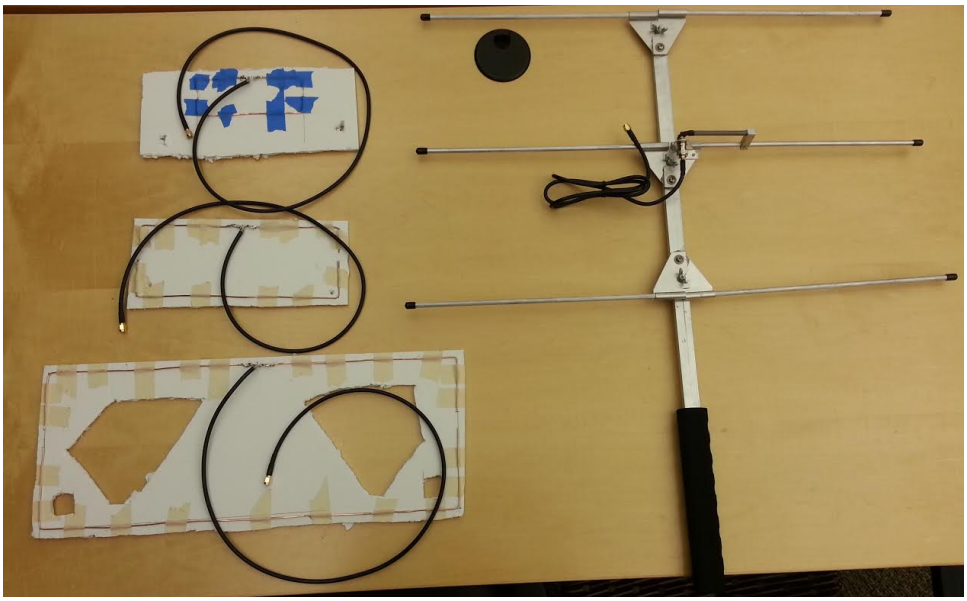

Figure 3.9: Custom Moxon antennas on the left, from top to bottom: 782 MHz, 432.7 MHz, 217.335 MHz. For size comparison, a commercially available 217 MHz Yagi is on the right.

the outer conductor is soldered to the other. A male SMA connector is soldered to the free end of the coax cable so it can feed into a radio. Design and construction can be completed in under an hour. Figure 3.9 shows completed antennas.

The use of custom antennas may seem counter-intuitive given the goal of making a hassle-free system. However, there are several reasons to use custom antennas. First, researchers might be interested in a frequency that is not commonly used and for which no commercial antennas exist. Second, custom antennas can be significantly less expensive than their COTS counterparts. This modality uses two antennas at once, so a researcher interested in three separate frequencies would have to purchase six antennas. Antennas range from tens to hundreds of dollars. In contrast, the material cost of these antennas—copper wire, styrofoam, coaxial cable—is negligible. Finally, most COTS antennas are not designed specifically for drones and can be heavy. For example, the commercially available 217 MHz antenna in Figure 3.9 is 0.54 kg, whereas the Moxon is only 0.12 kg.

3.4.2 Mathematical Model

The sensor system returns $z_t = 1$ if the front antenna measurement is higher and $z_t = 0$ if not. This model condenses two continuous, real-valued strength measurements into an observation $z \in \mathcal{Z} = \{0, 1\}$. The reduction is useful if planning requires an expectation over observations, which is often the case in information-theoretic control.

When the relative bearing is 0°, the front antenna points directly at the radio source, and a measurement of 1 is expected. When the relative bearing is 180°, the rear antenna points directly at the radio source, and a measurement of 0 is expected. A measurement of 1 is expected if the relative bearing is in the interval $[-90°, 90°]$, meaning the front antenna is pointed more closely to the radio source than the rear antenna. However, the front and rear antenna gains become similar at relative bearings near ±90°, so mistakes are more likely. Therefore, we define a cone width $\alpha \leq 180°$ over which we are confident the proper measurement will be returned. Intuitively, this setup can be thought of as two cones of width α and with vertices at the drone position; one cone is centered along the drone heading and the other in the opposite direction. If the radio source lies in the front cone, a measurement of 1 is expected; if the radio source lies in the rear cone, a measurement of 0 is expected. If the radio source lies between these cones, either measurement is equally likely. A mistake rate μ denotes the probability the drone misidentifies the cone containing the radio source, if the source lies in one. In flight tests and simulations, a value of $\mu = 0.1$ is assumed.

The mathematical expression of this model is

$$P(z_t = 1 \mid x_t, \theta) = \begin{cases} 1 - \mu, & \text{if } \beta_t - h_t \in \left[-\frac{\alpha}{2},\ \frac{\alpha}{2}\right] \\ \mu, & \text{if } \beta_t - h_t \in \left[180° - \frac{\alpha}{2},\ 180° + \frac{\alpha}{2}\right] \\ 0.5, & \text{otherwise.} \end{cases} \tag{3.6}$$

Figure 3.10 indicates that $\alpha = 120°$ is an appropriate cone width. If the radio source lies between front and rear cones of with 120°, either measurement is equally likely. A benefit of limiting cone width is that the resulting uncertainty region encodes uncertainty caused by imperfect antennas. Ideally, the front and rear cones would

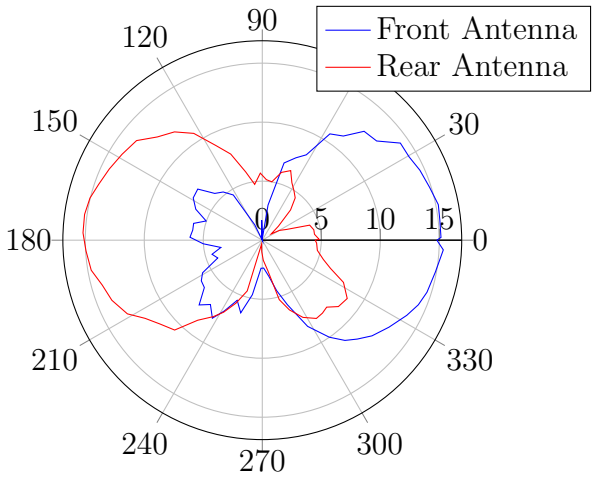

Figure 3.10: Signal strengths as functions of relative bearing to radio source (UV-5R radio). The front antenna receives higher strength when the drone faces the radio source (that is, when the relative bearing is 0°).

have a width $\alpha = 180°$, but this would require perfectly constructed and placed antennas. In reality, some antennas might have larger side and back lobes in their gain patterns; or, the front-facing antenna might not be placed exactly along the drone's heading axis. The uncertainty region obviates the need for operators to perfectly construct and align their antennas.

3.4.3 Flight Tests

Figure 3.10 shows measured signal strengths taken from antennas mounted on the drone in flight, with the UV-5R radio as the radio source. In-flight patterns for the wildlife radio-tag and the cell phone are shown in Figure 3.11. Figure 3.12 shows the Moxon antenna and resulting patterns for the 915 MHz telemetry radio.

The double-Moxon modailty is robust to antenna construction and placement, and it is also robust to time-varying factors affecting signal strength. For example, transmitter strength and orientation can change during a rotation. This robustness was tested by manually rotating the UV-5R radio as the drone rotated in place and made strength measurements. The radio's antenna is a dipole, so it has low gain along its antenna's axis. The resulting strength measurements can be seen in Figure 3.13. The

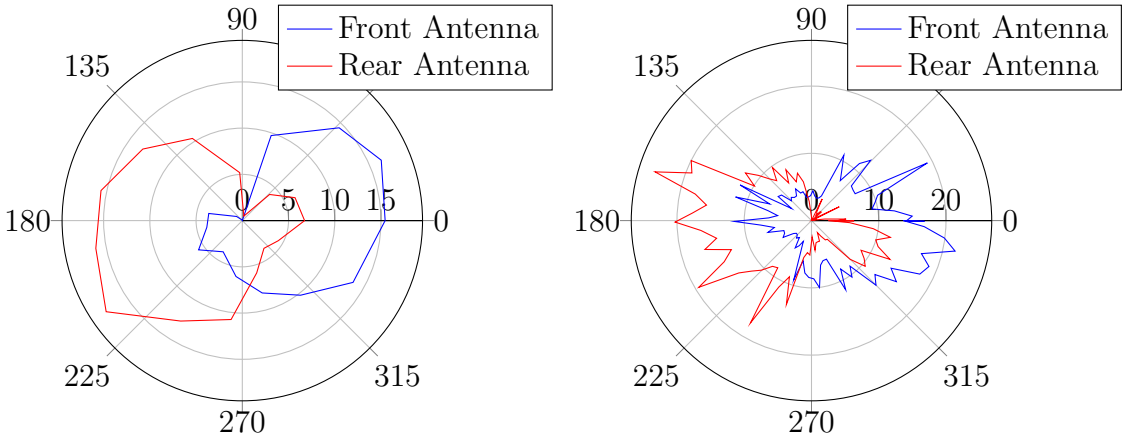

Figure 3.11: (Left) Signal strength measurements made 20 m from the wildlife collar. (Right) Signal strength measurements made 100 m from a cell phone placing a voice call over LTE.

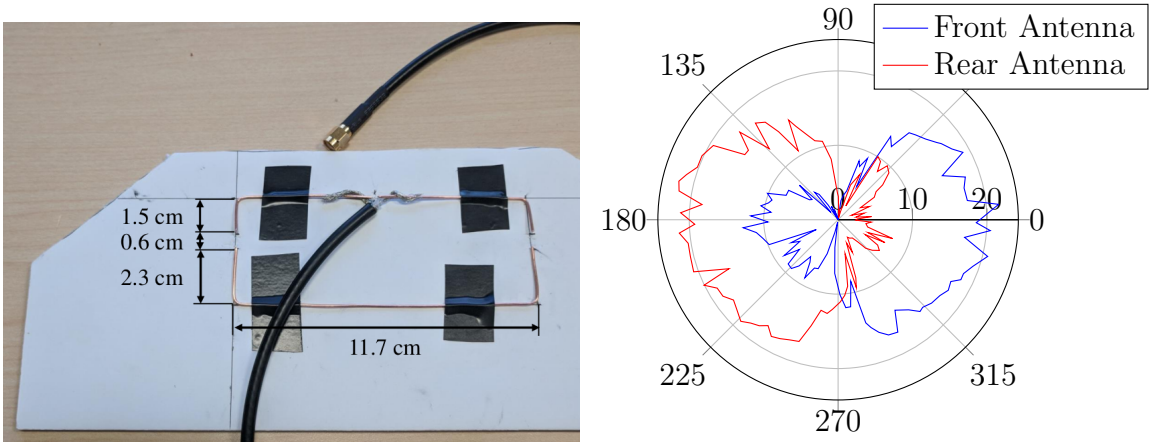

Figure 3.12: (Left) Moxon antenna built from 18 AWG copper wire for 915 MHz. (Right) Strength measurements made 62 m from 915 MHz telemetry radio.

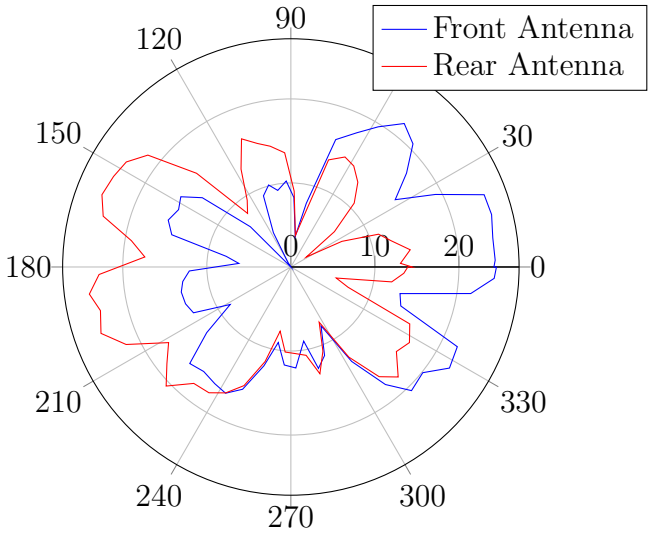

Figure 3.13: Strength measurements while rotating UV-5R so received strength changes. Both front and rear measurements are affected equally.

patterns are distorted because changes in the radio's orientation affect the strength reaching the drone's position. Traditional rotate-for-bearing approaches would have difficulty estimating bearing from these patterns. However, the two-antenna approach is not affected because both antennas are affected equally. The front-facing antenna measures greater strength in the front cone, and the rear antenna measures greater strength in the rear cone.

To validate the modality in localization, autonomous localization tests were flown in a 400 m × 400 m search area with the different transmitters. The drone flew at an altitude of 10 m, moved at 5 m/s, made measurements at 1 Hz, and used a greedy, information-theoretic policy. Figure 3.14 shows one flight test trajectory. The beliefs shown were generated on the drone; no post-processing was done. After 37 s, the drone is fairly certain of the radio's location; localization occurs in roughly the time it would take to perform one rotation in the rotate-for-bearing modalities.

Using the received observations and the GPS coordinates of the drone and radio source, the true mistake rate can be estimated. Across localization attempts, the drone made 179 measurements where the transmitter was either in the front or rear

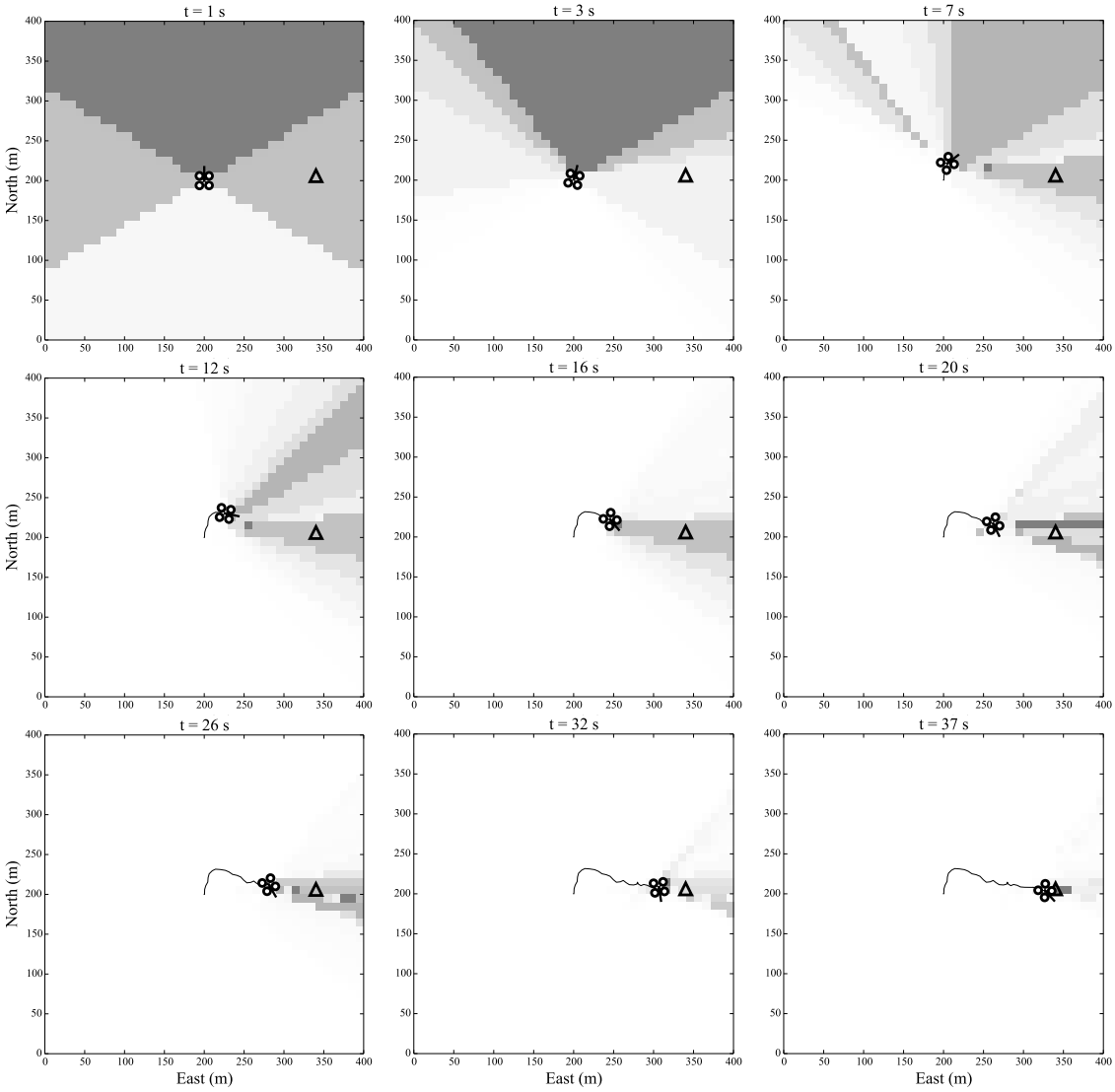

Figure 3.14: Flight test trajectory localizing the UV-5R radio (triangle). After 37 seconds, the drone is fairly certain of the radio's location.

cone; of these, 166 observations corresponded to the correct cone. This ratio corresponds to a mistake rate of 0.073, which is slightly less than the value of 0.1 assumed earlier. This mistake rate held across the different transmitters; even though the phone's strength varied rapidly with time, it only made 4 mistakes in 79 observations, again validating the modality's robustness. Of the 203 measurements made when a transmitter was in the uncertainty region between the cones, a measurement of 1 was observed 111 times, or 54.7% of the time. This value agrees with the model, which assumes either observation is equally likely in the uncertainty region.

3.5 Simulations

Both sensing modalities were validated in flight experiments. However, it is difficult to run enough flight tests to quantitatively compare the sensing modalities. Therefore, simulations are run to analyze the performance of each modality.

3.5.1 Comparing Modalities

The directional-omni (DO) and double-Moxon (MM) modalities are compared to two existing modalities. The first is an instantaneous bearing (IB) modality that provides bearing estimates in real-time, which might be implemented by a complex method like beam-steering. This modality is included to quantify the performance decrease incurred by avoiding the complexities of beam-steering. The second modality is the rotate-for-bearing (RFB) method in which the drone rotates in place and estimates the bearing with a directional antenna. The drone then moves to the next measurement location and rotates again.

The DO modality samples at 1 Hz and is assumed to have noise standard deviation of 2 dB. The MM modality samples at 1 Hz and is assumed to have a cone width of $\alpha = 120°$ and mistake rate of $\mu = 0.1$. The IB method also samples at 1 Hz. Bearing estimates for both the IB and RFB methods have zero-mean Gaussian noise with a standard deviation of 5°. This noise level is roughly half that reported in some work [22]. When using the RFB method, the drone takes 24 s to rotate. With all

Table 3.3: Mean time to concentrate 50% of the belief in a single 5 m × 5 m cell in a 200 m × 200 m search area.

Sensing Modality	Localization Time (s)	Noise Parameters
Rotate for Bearing (RFB)	99.2	$\sigma = 5°$
Directional-Omni (DO)	22.1	$\sigma = 2\,\text{dB}$
Double-Moxon (MM)	30.8	$\alpha = 120°,\ \mu = 0.1$
Instantaneous Bearing (IB)	17.5	$\sigma = 5°$

modalities, the drone moves at 5 m/s. Greedy, information-theoretic control is applied for all modalities. For the RFB modality, this greedy controller picks the measurement location that most reduces entropy. The drone then moves to this point and rotates to get a bearing estimate. The controller then picks a new measurement location and the process repeats.

A total of 1000 localization simulations are run for each modality. A 200 m × 200 m search area is split into 1600 5 m × 5 m cells. The stationary target is considered localized when 50% of the belief is conentrated in a single cell. Shown in Table 3.3, the simulation results show value of the modalities presented in this chapter. First, both DO and MM sensing can significantly outperform the RFB scheme, localizing the target in roughly the time a *single* bearing measurement is made—or less, if you use 40 and 45 s per rotation as reported in some work [15], [30]. Overall, RFB localization is much slower, and this estimate is conservative; the method would be even slower if longer rotation times or larger bearing standard deviations were used. The performance of the DO and MM modalities is much closer to that of IB, despite being much less complex than beam-steering.

To better visualize how informative each measurement is, the evolution of the belief max-norm during a single simulation is shown in Figure 3.15. When using the IB and MM methods, the belief max-norm rises much more quickly; this result satisfies intuition as measurements are made every second. In contrast, the max-norm during the RFB method only jumps up every after each roation is completed. While the MM measurements are less informative than the IB measurements, performance is much closer to IB, despite being far easier to implement on a real drone.

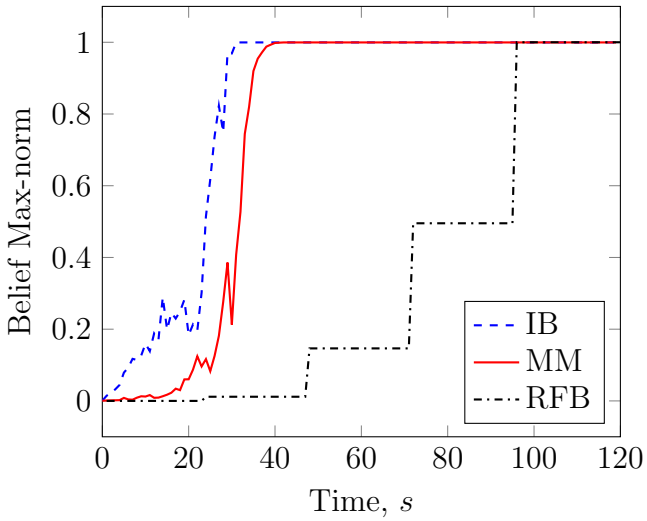

Figure 3.15: Evolution of belief uncertainty for different modalities during a single simulation.

3.5.2 Measurement Quality

For the double-Moxon modality, measurement quality is modeled with the mistake rate μ and the cone width α. The latter results in uncertainty regions where either observation is equally likely. A cone width of 120° was selected by examining experimentally derived gain patterns such as those shown in Figures 3.10 and 3.13. The flight tests described in Section 3.4.3 suggested a mistake rate of less than 0.1. For the directional-omni modality, measurement quality is modeled with the standard deviation.

It is important to understand how sensitive localization performance is to measurement quality. For example, if performance drastically improves with a wider cone width, it may make sense to invest in more precise antennas. To test this sensitivity, 1000 simulations were run for each of various combinations of sensing modality and measurement quality. Greedy information-theoretic policies are used as before. Results can be seen in Figure 3.16.

For the double-Moxon modality, localization time decreases as the cone width increases (which reduces the uncertainty region). At the ideal cone width of 180°,

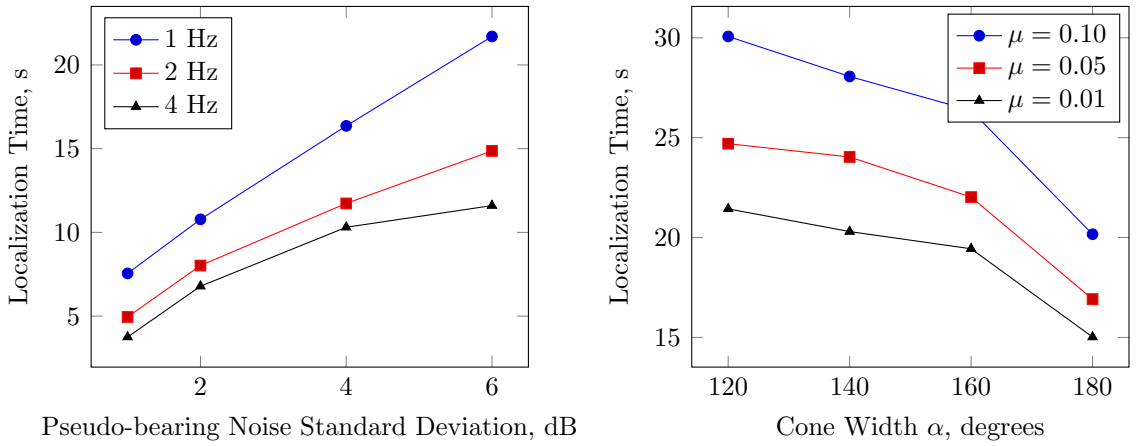

Figure 3.16: Directional-Omni (left): Effect of sampling rate and noise on localization. Double-Moxon (right): As the cone width increases, the uncertainty region shrinks, leading to faster localization.

localization takes roughly two-thirds the time it does at a cone width of 120°. However, this reduction only corresponds to savings of 10 s, and the mechanical difficulty of building near-perfect antennas is probably not worth the time savings. Likewise, a lower mistake rate μ reduces noise and localization time.

3.5.3 Measurement Quantity

Localization depends not only on measurement quality but also measurement quantity. Measurement quantity is dictated by the measurement sample rate. In the RFB method, this rate is limited by the time to make a full rotation. However, in the DO, MM, and IB methods, the sample rate is limited only by the electronics involved. Because the wildlife collar only transmits pulses at 1 Hz, that sample rate is the nominal for all radio sources. However, a higher sample rate can be used if the radio source continuously transmits. Higher sample rates yield more measurements which reduce localization time by providing more information about the transmitter's location.

To test the effect of increased sample rates, 1000 simulations for each of a variety of sample rates while using greedy, information-theoretic policies. The cone width and mistake rate were set to the default values of 120° and 0.1. Regardless of the sample rate, the drone was limited to a planar speed of 5 m/s and an angular speed

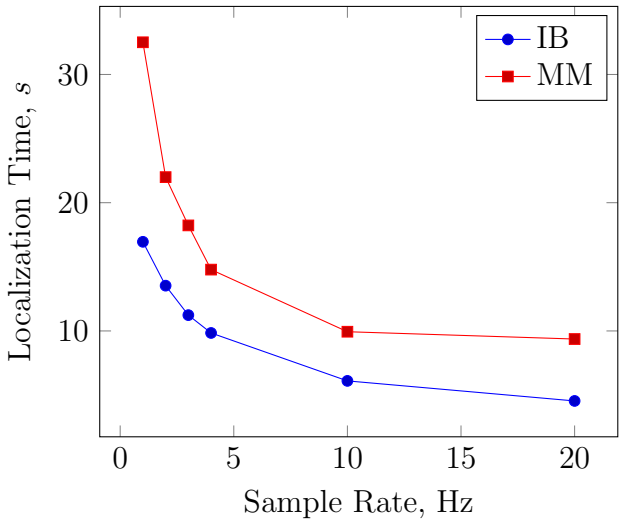

Figure 3.17: As the sample rate increases, the time to localization decreases.

of 10 °/s. The results can be seen in Figure 3.17. Increasing the sample rate can drastically improve localization time. At 10 Hz, localization time is less than half the time to perform a single rotation in the RFB modality.

3.6 Discussion

In this chapter, two sensing modalities for drone-based radio localization are presented and evaluated. These methods approach the performance of beam-steering, despite being much simpler.

The performance of the directional-omni method closely approaches that of instantaneous bearing measurements. While this modality has good performance, it has practical issues that make it unattractive. The modality did not work when tested at 432.7 MHz, because no commercially available antennas that were sufficiently omnidirectional could be found. It is likely difficult that custom-built antennas will be sufficiently omnidirectional, so this modality is less widely applicable than the double-Moxon modality. Further, having to hang the omnidirectional antenna to keep it omnidirectional presents more practical difficulties.

In contrast, the double-Moxon modality seemed to be far more robust, having

successfully been tested at 217, 432.7, 782, and 915 MHz. The Moxom antennas are also inexpensive to make and can be made for any frequency, freeing users from having to search for commercially available antennas. Therefore, the double-Moxon modality is preferred and used throughout the rest of the thesis.

Chapter 4

Belief Rewards in Offline POMDP Solvers

In the previous chapter, a greedy optimization guided the drone during localization, meaning the drone acted to minimize the expected entropy at the next step. Greedy, single-step planners are attractive because they are computationally simple; they do not require planning how the belief (target distribution) might evolve several steps in the future. However, greedy methods are generally suboptimal, so this thesis explores non-myopic belief-space planners.

Partially observable Markov decision processes (POMDPs) offer a principled, decision-theoretic approach to multi-step, closed-loop control under uncertainty [63]. POMDPs can be solved offline, allowing robotic agents to quickly query control actions given new measurements without having to do heavy computation online. Although solving POMDPs exactly is computationally intractable [64], recent offline algorithms generate approximately optimal policies with tight bounds on suboptimality, even for large problems. This progress makes POMDPs attractive for real robotic tasks involving uncertainty. Unfortunately, tasks like target localization or active sensing are ill-served by this approach because the traditional POMDP framework requires costs to depend on state and action only. Expressions of uncertainty, such as distribution entropy, depend instead on the belief.

Early efforts to overcome this limitation used surrogate rewards or belief compression. These techniques often lack bounds on suboptimality. Fortunately, recent work shows that belief-dependent rewards can be used in the POMDP framework with modifications to existing offline solvers [65]. However, issues like bounds and performance merit further investigation.

This chapter expands on this recent work on offline POMDP solvers for problems with belief-dependent rewards. SARSOP, a state-of-the-art offffline POMDP solver [66], is modified to handle belief-dependent rewards. Compact representations are provided for these rewards; these representations do not require adding many actions or α-vectors, in contrast to prior work. An improved lower bound that significantly reduces computation time is also presented and validated in simulations. A simple version of the drone-based radio localization problem is simulated to test the scalability of the resulting offline solver.

4.1 Background

4.1.1 POMDP Preliminaries

A POMDP consists of a state space $\mathcal{S}$, action space $\mathcal{A}$, observation space $\mathcal{O}$, transition function T, observation function Z, reward function R, and discount factor γ. At each time step t, the agent takes action $a \in \mathcal{A}$ from state $s \in \mathcal{S}$, arriving in some new state $s' \in \mathcal{S}$ with probability $P(s' \mid s, a) = T(s, a, s')$. The agent also receives a reward $r_t = R(s, a)$. The agent's goal is to maximize the expected discounted reward $\mathbb{E}[\sum_{t=0}^{\infty} r_t \gamma^{-t}]$, where $\gamma \in [0, 1)$ ensures a finite sum.

If the agent always knows its state, the problem is fully observable and simply called a Markov decision process (MDP). In an MDP, a policy π maps states to actions. The expected discounted reward starting from state s and following policy π is called the value of state s and is denoted $V^{\pi}(s)$. The goal is to find an optimal policy π^* that maximizes the value from every state. This optimal value function V^*

can be found by iteratively applying the Bellman update to convergence:

$$V(s) = \max_a R(s,a) + \gamma \sum_{s'} T(s,a,s')V(s'). \tag{4.1}$$

In a POMDP, the state is not fully observable. Instead, a noisy observation $o \in \mathcal{O}$ is sampled according to the observation function $Z(a,s',o) = P(o \mid a,s')$ representing the probability of observing o after taking action a and ending up in s'. These noisy observations are combined with a prior to maintain a belief b, or probability distribution over the states. After taking action a from b and observing o, a new belief b' can be generated with Bayes' rule. The solution to a POMDP is a mapping from belief to action. The Bellman update for POMDPs is

$$V(b) = \max_a \rho(b,a) + \gamma \int_{b'} \tau(b,a,o,b')V(b')db'. \tag{4.2}$$

Equation (4.2) is similar to Equation (4.1), where the states are now beliefs. The transition function τ describes the probability of transitioning from b to b' given action a and observation o. It can be rewritten in terms of T and Z. The belief-dependent reward function $\rho(b,a)$ is rewritten using a state-based reward $R(s,a)$:

$$\rho(b,a) = \sum_s R(s,a)b(s). \tag{4.3}$$

Because ρ is expressed as an expectation of state-based reward conditioned on belief, value functions generated with Equation (4.2) are piecewise linear and convex (PWLC) over belief. Therefore, the optimal value function can be approximated arbitrarily well with a set of linear functions [67]. These linear functions are called α-vectors. The value function is the upper surface of a set Γ of α-vectors:

$$V^\Gamma(b) = \max_{\alpha \in \Gamma} \ \alpha^\top b, \tag{4.4}$$

where $\alpha^\top$ is the transpose of α. At belief b, the policy defined by Γ recommends the

action $\pi^{\Gamma}(b)$ according to:

$$\pi^{\Gamma}(b) = \underset{\alpha \in \Gamma}{\operatorname{argmax}} \ \alpha^{\top} b. \tag{4.5}$$

4.1.2 Offline Solvers

Offline POMDP solvers typically update Γ until the resulting value function closely approximates the optimal. These updates are carried out with point-based value iteration, where Bellman backups are performed at a set of points in belief space [68]. Belief space is infinite, so these methods only consider the reachable space, the set of beliefs that can be reached from an initial belief. A search tree is created from this initial belief, and the transition and observation functions are used to generate new belief nodes. A benefit of offline solvers is that all solving happens before problem execution; while executing the policy, an agent simply querries the offline results. A downside of offline solvers is that they traditionally require reward functions that only depend on state and action to ensure the value function is PWLC.

SARSOP is an offline, point-based solver that reduces computation time by estimating the reachable space under optimal policies [66]. SARSOP maintains upper and lower bounds on the value function and uses heuristics to predict the value of new beliefs. These techniques reduce the size of the search tree.

4.1.3 Prior POMDP Localization Approaches

Surrogate rewards are a common approach to circumventing the limitation of state-dependent rewards [69], [70]. Surrogate rewards "trick" the agent into desired behavior with a state-based reward that requires solving the localization problem. A surrogate reward might be given if the agent reaches the target's location. Although this reward encourages the agent to find the target, the agent is also incentivized to stay near the target, even if better measurements can be made farther away. As a result, localization performance may be suboptimal with surrogate rewards.

Another POMDP localization technique is to augment the state space with a compressed version of the belief [36], [71], [72]. The compressed belief commonly

consists of the belief entropy and the state with highest probability. The dynamics of transitioning between these augmented states can be learned through Monte Carlo simulations [36], [70]. Although learning these dynamics allows for non-greedy planning to minimize entropy, compressing different beliefs might lead to the same compressed belief. This loss of information can lead to suboptimal control.

Another common approach is to abandon long-term planning and focus instead on the next time-step. In localization tasks, these greedy approaches guide agents to take the control action leading to lowest expected entropy after a single step [16]. Entropy is a measure of spread in a distribution (a uniform distribution maximizes entropy), making it a good objective function. However, greedy behavior can be suboptimal as the agent trades long-term optimality for short-term gain.

4.2 Belief-Dependent Rewards

As explained in Section 4.1.2, POMDP solvers like SARSOP rely on state-dependent rewards to maintain a PWLC value function that can be approximated with α-vectors. A key insight by Araya et al. [65] was that so long as $\rho(b, a)$ was itself PWLC, then value functions generated with Equation (4.2) would also be PWLC. The term "ρPOMDP" refers to POMDPs with PWLC belief-dependent rewards.

Another framework is the POMDP with information rewards (POMDP-IR), which adds "guess" actions performed simultaneously with normal actions [73]. There is one guess action per state, each yielding a state-based reward if it corresponds to the true state. Although these actions greatly increase the action space, they decompose nicely out of the Bellman update because they do not affect the system dynamics. It has actually been shown that a POMDP-IR is equivalent to a ρPOMDP [74].

Here, three PWLC belief-dependent reward functions are examined. None rely on entropy—a common uncertainty measure—because it is not piecewise linear. One could generate a PWLC approximation with tangential hyperplanes at selected points, but generating a good, dense approximation before solving can lead to an enormous set of hyperplanes. An alternative is to only generate hyperplanes at new nodes in the belief tree, but this requires extra computation at each new node.

4.2.1 Max-Norm Reward

An alternative reward is the ℓ_∞-norm, or max-norm, proposed by Eck and Soh in the context of ρPOMDPs [75]. This PWLC function can be represented exactly with the standard basis of $\mathbb{R}^{|\mathcal{S}|}$:

$$\rho(b,a) = \max_{\alpha \in \Gamma_\rho} \alpha^\top b, \quad \Gamma_\rho = \left\{e_1, ..., e_{|\mathcal{S}|}\right\}, \tag{4.6}$$

where e_i is a vector of zeros except for element i, which is 1. The ability to compactly and exactly represent the max-norm reward is a great advantage over negative entropy. Surprisingly, a sparse approximation of negative entropy can perform worse than a max-norm reward, even when evaluated by the expected sum of negative entropy [76]. The max-norm is also more intuitive—a max-norm of 0.6 suggests there is a 60% chance the agent is in the most likely state, whereas a distribution entropy of 2 nats is less useful to a human evaluator.

4.2.2 Threshold Reward

A disadvantage of the max-norm is that the agent always receives some reward, even at uniform beliefs. Sometimes, we want an agent to reach a highly concentrated belief as quickly as possible, but the agent might be driven by the max-norm reward to collect rewards at less-concentrated beliefs in the near-term. Spaan, Veiga, and Lima suggested thresholded rewards in the POMDP-IR framework, but this requires an additional guess action per state [73]. This ρPOMDP version does not:

$$\rho(b,a) = \max\left(\frac{\|b\|_\infty - c_\rho}{1 - c_\rho}, 0\right), \tag{4.7}$$

where c_ρ is the max-norm cutoff. A belief max-norm below c_ρ induces no reward. Above c_ρ, the reward increases linearly until it reaches a maximum value of 1. An exact representation of the threshold reward only needs one hyperplane per state and an additional $\vec{0}$ hyperplane.

4.2.3 Guess Reward

This chapter examines a final reward function introduced in the POMDP-IR literature [73]. In a POMDP-IR, the agent guesses the true system state at each time step. The agent is rewarded 1 for guessing correctly and 0 otherwise. Satsangi, Whiteson, and Spaan showed that this reward function is equivalent to the max-norm, because the expected reward of the guess equals the belief max-norm [74]. In one variant, the agent can guess *instead of* taking a normal action. The agent's action space is augmented with a single guess action independent of the problem dynamics; it is assumed the state with highest belief density is chosen for the guess. This guess reward function can be represented as

$$\rho(b, a) = \mathbb{1}\{a = \text{guess}\} \left(\max_{\alpha \in \Gamma_\rho} \alpha^\top b \right), \tag{4.8}$$

where Γ_ρ is defined in Equation (4.6) and $\mathbb{1}\{x\}$ is the indicator function that returns 1 if x is true. Eck and Soh pointed out that purely belief-dependent rewards require an external termination condition, like an entropy threshold [75]. The guess action forces the agent to reason about the cost of acquiring new information, removing the need for external stopping conditions.

4.2.4 Action Rewards

Often there is a cost to performing sensing actions—they might take longer than other actions or use more resources. Adding an action-dependent reward $R(a)$ maintains the PWLC property.

4.3 SARISA

Here, the PWLC rewards from the previous section are incorporated into an offline, point-based solver. Specifically, SARSOP is modified, and the resulting algorithm is called SARSOP with information-seeking actions (SARISA).

4.3.1 Backup

The backup operation uses the Bellman update to improve the value function at belief b using information at the child beliefs of b. First, the vector $\alpha_{a,o}$ is selected for every action a and observation o. This vector maximizes the value at the child belief reached when taking a from b and observing o:

$$\alpha_{a,o} = \underset{\alpha \in \Gamma}{\operatorname{argmax}} \ \alpha^\top b^{ao}, \tag{4.9}$$

where b^{ao} is the belief reached when taking a from b and observing o. Then, a set of α-vectors is created, with one for each action a, where α_a describes the α-vector created for action a. The entry in α_a for state s is updated:

$$\alpha_a(s) = R(s,a) + \gamma \sum_{o,s'} T(s,a,s') Z(a,s',o) \alpha_{a,o}(s'). \tag{4.10}$$

To extend Equation (4.10) for a PWLC belief-dependent reward, we can define $\alpha_b = \operatorname{argmax}_{\alpha \in \Gamma_\rho} \alpha^\top b$. Adding the action-dependent reward, the update becomes

$$\alpha_a(s) = \alpha_b(s) + R(a) + \gamma \sum_{o,s'} T(s,a,s') Z(a,s',o) \alpha_{a,o}(s'). \tag{4.11}$$

where $\alpha_b = \operatorname{argmax}_{\alpha \in \Gamma_\rho} \alpha^\top b$. If the max-norm reward is used, the update is:

$$\alpha_a(s) = \mathbb{1}\{s = \underset{s'}{\operatorname{argmax}}\, b(s')\} + R(a) + \gamma \sum_{o,s'} T(s,a,s') Z(a,s',o) \alpha_{a,o}(s'). \tag{4.12}$$

A similar update can be written for the threshold reward:

$$\alpha_a(s) = \frac{\mathbb{1}\{s^* > c_\rho\}}{1 - c_\rho} \Big[\mathbb{1}\{s = s^*\} - c_\rho \Big] + R(a) + \gamma \sum_{o,s'} T(s,a,s') Z(a,s',o) \alpha_{a,o}(s'), \tag{4.13}$$

where $s^* = \operatorname{argmax}_s b(s)$. Equations (4.12) and (4.13) represent a computational benefit over the traditional ρPOMDP backup shown in Equation (4.11)—there is no need to maintain a set Γ_ρ or compute α_b at each backup, a criticism of ρPOMDPs [74].

4.3.2 Upper Bound

In SARSOP, the upper bound is represented with a set of belief-value pairs, and the sawtooth approximation [77] is used to interpolate for values at new beliefs. The fast informed bound (FIB) approximation generates this upper bound. FIB switches max and sum operators in the Bellman update and is an upper bound on the value function [78]. Here, FIB is derived for rewards depending on belief and action. FIB is initialized with a set Γ of α-vectors, with one α-vector α_a per action a, each of which is usually initialized to zeros. Starting with a variant of the Bellman update, a max and sum operator are switched, allowing $b(s)$ to be pulled out:

$$V(b) = \max_a \left[\rho(b,a) + \gamma \sum_o \max_{\alpha \in \Gamma} \sum_{s,s'} b(s) P(s', o \mid s, a) \alpha(s') \right] \tag{4.14}$$

$$= \max_a \left[\max_{\alpha \in \Gamma_\rho} \sum_s b(s)\alpha(s) + \sum_s b(s) R(a) \; + \gamma \sum_o \max_{\alpha \in \Gamma} \sum_{s,s'} b(s) P(s', o \mid s, a) \alpha(s') \right] \tag{4.15}$$

$$\leq \max_a \sum_s b(s) \left[\max_{\alpha \in \Gamma_\rho} \alpha(s) + R(a) \; + \gamma \sum_o \max_{\alpha \in \Gamma} \sum_{s'} P(s', o \mid s, a) \alpha(s') \right]. \tag{4.16}$$

Note that $P(s', o \mid s, a) = T(s, a, s') Z(a, s', o)$. The PWLC belief-dependent reward is assumed to be uniform at the corners of the belief simplex (when the belief is concentrated in a single state), as is the case with negative entropy and the max-norm. This corner reward is denoted r_{b^*}. Assuming no state-dependent rewards, every element in a specific α-vector will have the same value. For α-vector α, this constant value is denoted α_c. It does not rely on s' or o and can be pulled out of the

summation, which, by the laws of probability, sums to 1:

$$V(b) \leq \max_a \sum_s b(s) \left[r_{b^*} + R(a) + \gamma \max_{\alpha \in \Gamma} \alpha_c \sum_{o,s'} P(s', o \mid s, a) \right] \tag{4.17}$$

$$= \max_a \sum_s b(s) \underbrace{\left[r_{b^*} + R(a) + \gamma \max_{\alpha \in \Gamma} \alpha_c \right]}_{\alpha_a^{(k+1)}(s)}. \tag{4.18}$$

Each α_a can now be updated iteratively, independently of belief. The element corresponding to state s is updated in step $k+1$ using the α-vectors from step k:

$$\alpha_a^{(k+1)}(s) = r_{b^*} + R(a) + \gamma \max_{\alpha^{(k)} \in \Gamma} \alpha_c^{(k)}. \tag{4.19}$$

This iteration can be represented as a geometric sum because the α-vector maximizing $\alpha_c^{(k)}$ always belongs to the action with highest reward. Thus, every element in α_a converges to

$$R(a) + \frac{r_{b^*} + \gamma \max_a R(a)}{1 - \gamma}. \tag{4.20}$$

Because every element in an α-vector is the same, the α-vector belonging to the highest reward action dominates at any belief—and each element has the value

$$\frac{r_{b^*} + \max_a R(a)}{1 - \gamma}. \tag{4.21}$$

This dominant α-vector is used to generate a set of belief-value pairs, initializing the upper bound. The result in Equation (4.21) is easy to compute and requires no iteration. However, this FIB-generated upper bound is equivalent to a naïve upper bound that simply discounts the maximum possible reward to infinity. This result is unsurprising because the FIB iterations include no notion of belief and should not be able to capture the effect of belief-dependent rewards. However, the result is shown here for completeness. Improved upper bounds are an area of future research.

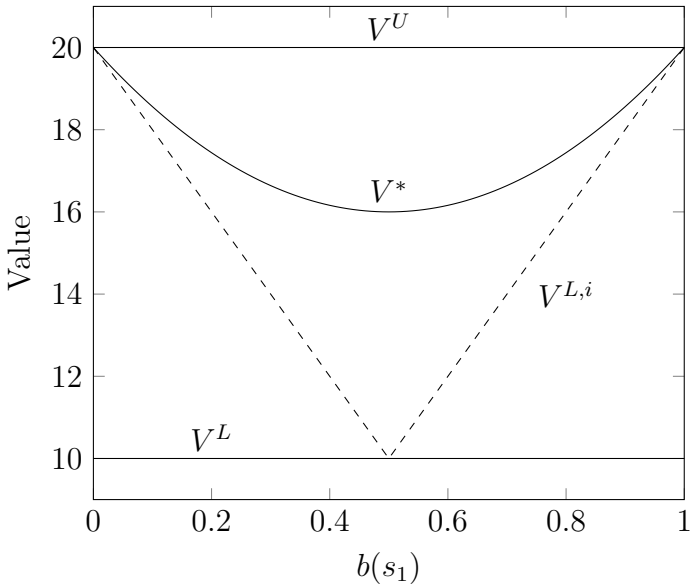

Figure 4.1: Example two-state problem with the max-norm reward, $\gamma = 0.95$, and no action costs. The true value V^* is bounded by upper and lower bounds V^U and V^L. The improved bound $V^{L,i}$ is much tighter than V^L.

4.3.3 Lower Bound

The lower bound maintained by SARSOP is the set Γ of α-vectors representing the value function. This bound is initialized with one α-vector per action using a blind policy [79]. In a POMDP with belief and action-dependent rewards, the worst greedy reward is equal to $r_{b_w} + \max_a R(a)$, where r_{b_w} is the worst belief-dependent reward, typically achieved when the belief is uniform. As with the upper bound, the resulting α-vectors will be dominated by the α-vector corresponding to the action with the highest reward, where every element is

$$\frac{r_{b_w} + \max_a R(a)}{1 - \gamma}. \tag{4.22}$$

The lower bound can be initialized to a single α-vector belonging to the highest reward action, with each element equal to the value shown in Equation (4.22), but this bound is very loose.

A tighter lower bound for the max-norm reward can be derived if the agent has

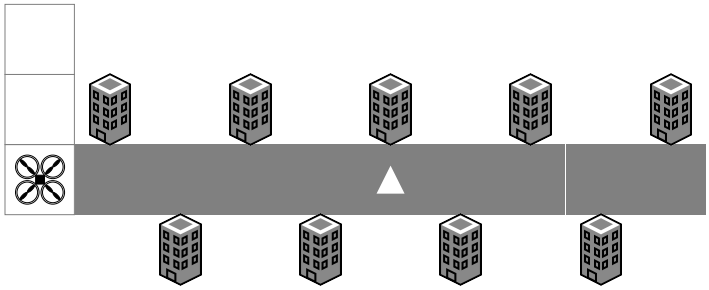

Figure 4.2: The LazyScout problem. The drone must find a radio beacon (white triangle) located between some buildings. Grey cells indicate possible locations of the hidden beacon. The drone can climb above the buildings to receive a perfect observation.

an action that is guaranteed not to change the belief. The belief max-norm remains unchanged after applying this action, and the infinitely discounted max-norm is a lower bound on the value at the belief. Figure 4.1 shows the improved bound, which directs exploration and helps convergence.

Localization of a stationary target always satisfies this assumption. The agent only needs a non-observing action that returns a null observation—common in target localization, where agents often have the option to move or make a measurement. If the agent is always sensing, we can simply add an action that discards the observation. This action only exists to guide exploration during solving, and it is unlikely to be the optimal action selected during execution.

If non-zero, the non-observing action's reward can be included in the infinite discounting. The improved bound can be expressed compactly with one α-vector per state, each corresponding to the non-observing action. The same bound holds for the guess reward function—the guess action takes the role of the non-observing action. A similar bound can be derived for the threshold reward function. An additional $\vec{0}$ α-vector represents the no reward belief region.

4.4 Example Problems

In this section, SARISA is compared against surrogate and greedy methods in toy problems. Toy problems are small problems that make it easy to diagnose the performance of different algorithms.

4.4.1 LazyScout

This subsection presents LazyScout, a toy localization task showing the possible suboptimality of greedy entropy minimization and surrogate rewards. A drone equipped with a range sensor seeks a radio beacon located between buildings. The drone knows its own location, so each range measurement implies a beacon location. When the drone travels between buildings, its observations are degraded by clutter and multipath. It might observe the grid cell containing the beacon, the cell before, or the cell after, each with equal probability. Alternatively, the drone can climb above the buildings. Climbing takes two time steps and no measurements can be made while climbing. However, once above the buildings, the drone observes the true beacon location. Figure 4.2 is a graphical representation of LazyScout.

The optimal action is to climb above the buildings, which ensures localization in two steps. However, both greedy entropy minimization and a surrogate reward strategy act suboptimally. Greedy entropy minimization fails because the noisy measurement received through the buildings is "better" than receiving no measurement while climbing. If we define a surrogate, state-dependent reward function that rewards the drone for reaching the beacon location, the drone will try to stay near the estimated location of the beacon. The extra time required to climb and descend is not worth it—the drone can piece together enough noisy measurements as it moves through the buildings and closer to the beacon. Localization might take longer, but the time to physically reach the beacon is reduced.

Simulation results comparing surrogate rewards, greedy rewards, and SARISA with the max-norm reward function are shown in Table 4.1. SARISA chooses the correct first action, cutting localization time in half (here localization means concentrating belief to a single cell). SARISA's bounds converge to 18.266, the theoretically

Table 4.1: Reward comparison for LazyScout

reward structure	first action	reward	steps to localize	solve time (s)
surrogate	buildings	17.521	4.3	0.86
greedy	buildings	17.521	4.3	-
max-norm	climb	18.266	2.0	0.06

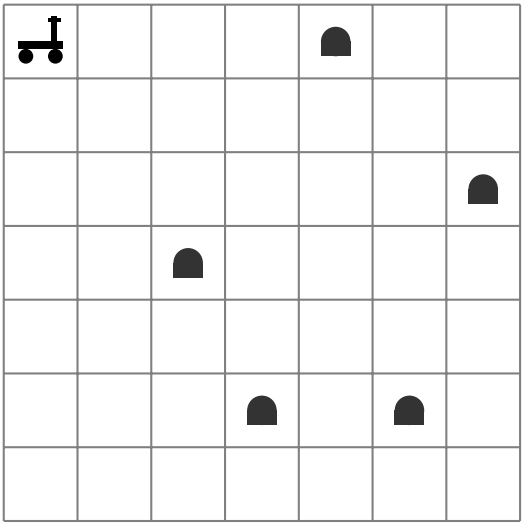

Figure 4.3: Grid used for rock problems: five rocks, $\gamma = 0.95$, rover starts in upper left.

correct initial value when evaluating with the max-norm reward and $\gamma = 0.95$.

The SARISA solver used the improved lower bound, leading to a solve time of 0.06 s. When this improved bound was not used, convergence took 0.99 s, nearly a factor of 17 longer. The improved bound drastically reduces the number of backups necessary: the improved version only used 237 backups while the unimproved version needed 1,961.

4.4.2 RockSample and RockDiagnosis

RockSample is commonly used to test the effectiveness of POMDP solvers [80]. A rover moves in a square grid and samples rocks that exist at known locations and

Table 4.2: Reward comparison for RockSample, when evaluated by max-norm reward.

policy	solve time (s)	reward
surrogate	34	7.6
SARISA (max-norm)	7200	12.7
reach	-	8.1
random	-	8.4

might have scientific value. From a given grid cell, the rover can move to a non-diagonal neighbor cell, use a laser to scan any rock, or sample a rock occupying the same cell. Scanning a rock provides a noisy measurement of its value. Sensor noise increases with the rover's distance from the rock. The rover is rewarded for sampling a valuable rock and penalized for sampling a worthless one.

The goal in a modified version of RockSample called RockDiagnosis is only to determine whether each rock is valuable [76]. The rover has no sample action—instead, it maneuvers and scans to learn the worth of each rock. The original RockSample can be seen as RockDiagnosis with a surrogate reward, where the sample costs exist only to encourage this learning, and a RockDiagnosis agent should outperform it.

The RockDiagnosis problem shown in Figure 4.3 was first solved with SARISA and the max-norm reward function, The resulting policy was compared to a "surrogate" policy solved on the RockSample model with SARSOP, a random action policy, and a "reach" policy that moved the agent in the shortest path to each rock, making a perfect observation at each (there is no noise when the distance to a rock is zero).

Table 4.2 shows the mean sum of discounted max-norm rewards during 2000 simulations of 100 steps for each policy. SARISA yields the highest reward, which is unsurprising because its reward function matches the evaluation reward function. However, the result is not insignificant. An early attempt at solving RockDiagnosis of the same size used a modified version of Perseus [81] and could not outperform the random policy [76], suggesting SARISA is an improvement over early POMDP solvers incorporating belief-dependent rewards.

A notable result is the slow convergence of SARISA—after 7200 s, the lower and upper bounds were 12.3 and 14.4. In contrast, the RockSample policy bounds converged to a width of 0.001 in just 34 s. One way to improve convergence is to find

Table 4.3: Reward comparison for RockSample, when evaluated by threshold reward.

policy	solve time (s)	reward
SARISA (thresh 0.9)	7200	6.1
SARISA (max-norm)	7200	4.2

tighter starting bounds, a subject of future research.

The effect of other reward functions is also explored. Suppose we want the rover to be 95%-confident—according to its model—in a rock configuration, as fast as possible. We might use the threshold reward from Equation (4.7) with a cutoff $c_\rho = 0.9$. Because beliefs with max-norm below 0.9 yield no reward, the agent is encouraged to reach highly concentrated beliefs more quickly. Figure 4.4 shows how quickly policies solved with max-norm and threshold rewards reach a belief with a max-norm of 0.95. The max-norm policies almost always failed to reach the desired confidence if they had been solved for less than an hour. After solving for two hours, the performance was much better, probably because SARISA had time to reach further down the belief tree to more highly-concentrated beliefs. In contrast, threshold policies solved for even a short amount of time reach the desired confidence quickly.

Policies were evaluated using the threshold reward. Mean discounted rewards are shown in Table 4.3. As expected, the threshold policy outperforms the max-norm policy because it was trained on the evaluation reward function. However, the SARISA policy is likely suboptimal because its bounds were unconverged; the lower and upper bounds were 4.6 and 13.7 after 7200 s. These bounds are much wider than in the max-norm case, most likely because rewards only occur deep in the search tree at concentrated beliefs. The improved lower bound also assigns no value to beliefs below the threshold max-norm, so the lower bound is probably loose, leading to poor convergence. Still, the improved lower bound significantly helps SARISA's performance. As Figure 4.5 shows, the improved lower bound is higher after 30 seconds of solving than the unimproved bound after two hours.

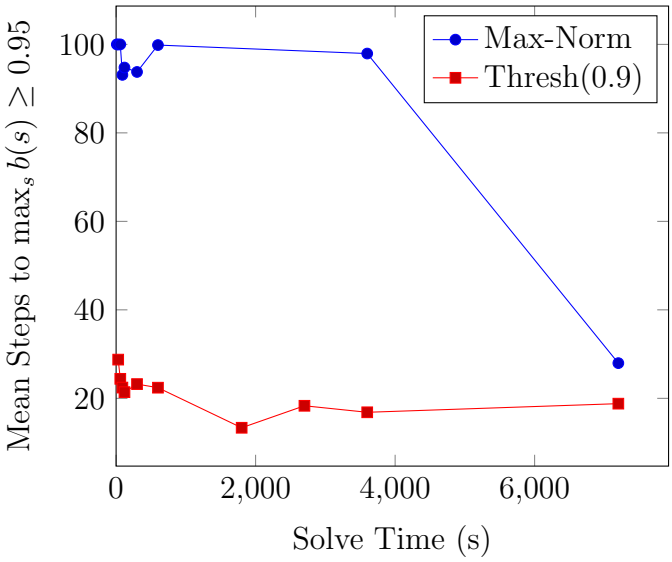

Figure 4.4: Average steps to reach a highly concentrated belief. If a trajectory did not reach the desired max-norm, the worst-case value of 100 was assigned.

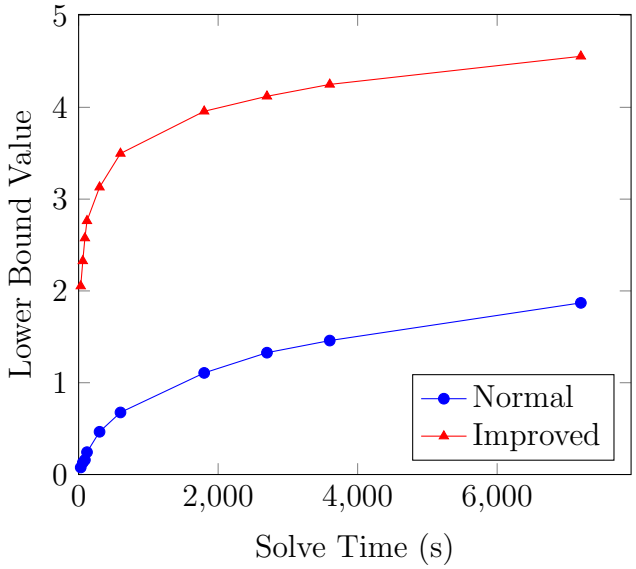

Figure 4.5: Lower bound on RockDiagnosis when using threshold reward with cutoff of 0.9. The improved lower bound improves convergence.

4.5 Simulating Drone-based Radio Localization

To test the viability of SARISA for drone-based radio localization, it is tested in a simulated, simplified radio localization problem. In this simplified problem, the search area is modeled as an 11×11 grid with $10\,\mathrm{m} \times 10\,\mathrm{m}$ cells. A state s consists of the known drone position $p_d = (x_d, y_d)$ and unknown target position $p_t = (x_t, y_t)$. At each step, the drone can deterministically move to a neighboring grid cell, rotate in place, or hover (terminate the search).

The drone uses the rotate-for-bearing scheme in which rotations in place yield a bearing estimate. The zero-mean Gaussian noise on the bearing measurements has a standard deviation of 13° at most ranges, but it increases to roughly 40° if the target and drone are in adjacent cells. To reduce computation, the angular space is split into 10° bins. An additional null measurement is received when the drone does not rotate, yielding 37 possible observations.

To make the drone reason about when to stop making measurements, a guess reward is used:

$$\rho(b, a) = \mathbb{1}\{a = \text{hover}\}\|b\|_\infty + \lambda R(a), \tag{4.23}$$

where $R(a)$ is the action reward and λ is a scale factor relating action and information rewards. The sensing reward $R(a)$ depends roughly on the time to complete an action; $R(a) = -1$ for moving in a cardinal direction, $R(a) = -\sqrt{2}$ for moving diagonally, and $R(a) = -3$ for rotating to measure bearing. A similar surrogate reward replaces the max-norm reward with 1 if the drone hovers over the target.

The value of λ is varied and resulting policies are evaluated. For each value of λ, a policy is generated by running SARISA for 12 hours; then 1210 simulations are run to completion, with the target at random locations and the drone starting at the center of the search area. Policies are evaluated by the time to make a decision (hover) and whether the drone's guess—the state with the highest probability—matches the true state. The SARISA policies are compared to a greedy policy that moves the drone to the cell that, after rotation, yields the lowest expected entropy. These greedy policies were stopped at different cutoff max-norm values. Additionally, SARISA policies are compared to SARSOP policies solved with a state-based surrogate reward R_{sur} that

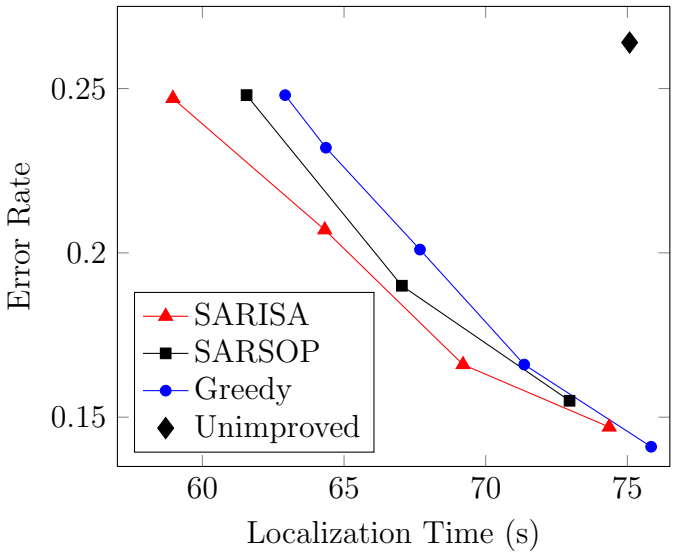

Figure 4.6: Simulation-produced Pareto curve showing the effectiveness of belief-dependent rewards in the simplified drone-based target localization problem.

rewards the drone for hovering over the target:

$$R_{\text{sur}}(s,a) = \mathbb{1}\left(a = \text{hover},\ p_d = p_t\right) + \lambda R(a). \tag{4.24}$$

As seen in Figure 4.6, SARISA policies achieve slightly less error in less time. However, at lower error rates than shown, POMDP methods underperform the greedy method, probably because this requires reaching further down the search tree. While SARISA is slightly better over the region shown, the other methods perform comparably. It is possible the greedy method is nearly optimal for this particular problem. The more striking result is the effect of the improved lower bound. Solving for $\lambda = 2$ yielded bounds of (45.8, 91.6) for the improved bound and (7.6, 92.6) for the unimproved bound. This inferior bound limited the depth of search tree exploration, and highly concentrated beliefs were not reached. As Figure 4.6 shows, only a single value of λ yielded a comparable error rate, and this point is Pareto dominated by all other solvers. In this problem, the improved lower bound enables use of belief-dependent rewards and a point-based POMDP solver. Another important insight arises from the surrogate's bounds: (44.1, 86.4). These are similar to SARISA's, suggesting

information-gathering problems are inherently difficult, even if the belief-dependent reward is wrapped into a similar state-dependent reward.

4.6 Discussion

This chapter explored the use of offline POMDP solvers for drone-based radio source localization, improving upon previous work showing how to incorporate belief-dependent rewards into these solvers. Different belief-dependent rewards and their effects on information-gathering behavior were explored. It was shown that the backup operations of these rewards do not need a set Γ_ρ of linear functions, leading to reduced computation during backup—the core, inner loop of POMDP solvers. An improved lower bound that greatly improves performance was also introduced.

Unfortunately, the evaluations in this chapter suggest offline POMDP solvers are not yet ready for realistic robotics problems. SARISA showed only slight improvement on a heavily simplified version of the drone-based radio localization problem. Despite a coarsely discretized search area, convergence was not reached after 12 hours of solving; the bounds would be far looser for a more realistically sized problem. Further, the improved lower bound makes the limiting assumption of a stationary target. Future work to improve the loose upper bound might explore myopic policy bounds [82]. The next chapter will explore online solvers, which typically scale better.

Chapter 5

Online Planning

In the last chapter, offline belief-space planning techniques were explored for the drone-based radio localization problem. While improvements were made to offline POMDP solvers with belief-dependent rewards, offline solvers seem to be ill-suited. It was shown that adequate performance on coarsely discretized problems requires a stationary target, so that a good lower bound can be used. To avoid the discretization and stationary assumption, online planners are explored in this chapter.

5.1 Background

It is well known that online POMDP solvers scale better than offline variants. This improved scaling results from the smaller search tree made by online solvers. An offline solver typically creates a search tree from an initial belief. As belief nodes resulting from different actions and observations are expanded, the number of nodes grows exponentially. In contrast, online solvers create a search tree from the current belief, effectively limiting their search to beliefs reachable from the current belief. As a result, their search area is much smaller, and good solutions can be achieved. Of course, this computation must be done with knowledge of the current belief; this tree search must happen in real-time on the robot. However, it is still generally worth it.

Online POMDP solvers have made remarkable progress. A number of online solvers work by running many simulations from the initial belief [83]–[85]. Each

simulation assumes a current true state, and evaluates the reward as this state is propagated forward down the search tree. These methods work well when evaluating state-dependent rewards, but they cannot reward or penalize belief dynamics; if our goal is to minimze belief uncertainty, we cannot get this information by watching the simulation of a single state in our belief. We need to see how the belief changes in time and assign rewards based on these changes. Therefore, the POMDP is formulated as a belief-state MDP, where the state incorporates the belief. This state can then be penalized or rewarded, allowing the agent to reason about how to reduce uncertainty in its target location estimate.

5.2 Method

Once the seeker drone makes a measurement and updates its belief, it selects a control input. This planning is performed by the seeker drone's onboard computer. The planning algorithm uses the Markov decision process (MDP) framework.

5.2.1 Markov Decision Processes

One way to model an MDP is with a state space $\mathcal{S}$, a control space $\mathcal{U}$, a cost function J, a generative model G, and a timestep horizon T. The model G generates the state at the next timestep, $s_{t+\Delta t} \in \mathcal{S}$, given the current state $s_t \in \mathcal{S}$ and control input $u_t \in \mathcal{U}$. This model can be stochastic so that $s_{t+\Delta t} \sim G(s_t, u_t)$.

A policy $\pi : \mathcal{S} \to \mathcal{U}$ maps each state to an action. The solution to an MDP is an optimal policy π^* that minimizes the expected total cost during the horizon T:

$$\pi^*(s_t) = \underset{u_t}{\operatorname{argmin}}\, \mathbb{E} \sum_{\tau=1}^{T} J(s_{t+\tau\Delta t}). \tag{5.1}$$

The expectation accounts for transition uncertainty.

In contrast, a greedy solution minimizes the expected cost at the next timestep:

$$\pi^g(s_t) = \underset{u_t}{\operatorname{argmin}}\, \mathbb{E}\, J(s_{t+\Delta t}). \tag{5.2}$$

Greedy policies are generally suboptimal as they value short-term gain at the expense of long-term optimality, but they are easy to implement and computationally inexpensive, so they have been used extensively for drone-based radio localization [15], [16], [24], [25]. However, all of these works assume the target is stationary.

5.2.2 Formulation

A traditional formulation of the tracking problem folds the seeker and target drone states into an overall system state [69]. Because the target drone state is unknown, this formulation is actually a partially observable MDP (POMDP). While there has been extensive work in solving POMDPs, they have a critical drawback in localization problems.

The classic definition of a POMDP requires that the cost function be defined in terms of the state. However, a belief-dependent reward often makes sense for tracking, where the goal is to have a belief with low uncertainty, leading to good estimates. One way to get around this problem is to define an equivalent state-dependent cost; such a cost function might reward the seeker for reaching the target's state [69]. This surrogate cost function encourages the seeker to take information-gathering actions and learn the target state. But we might want to encourage the seeker drone to avoid flying too close to the target, so there is no obvious surrogate cost. Rewarding the seeker for staying away from the target encourages the seeker to gather only as much information as needed to avoid collisions.

As the last chapter showed, it is possible to modify classic POMDP solvers to handle belief-dependent rewards, but these offline methods are slow even with coarse discretizations [65], [86]. An alternative is to formulate the POMDP as a belief-MDP, which is an MDP where a belief is part of the state. While the target state is unknown, the belief over possible target states is known. The agent can be penalized if the belief is spread out and contains a lot of uncertainty; this is just a state-dependent cost and can be easily handled in the MDP framework. The state at time t is

$$s_t = (b_t, x_t), \tag{5.3}$$

where b_t is the belief over possible target states and x_t is the position and heading of the seeker drone.

The control space is a discrete set of velocity commands that can be given to the seeker drone. Given u_t and s_t, the components of the next state $s_{t+\Delta t} = (b_{t+\Delta t}, x_{t+\Delta t})$ can be obtained with the particle filter update and the seeker drone state update. This update is stochastic because noise in the sensor model affects the resulting belief.

The cost function should encourage the seeker to make measurments that lead to good target estimates while keeping it a safe distance from the target drone. Good target estimates are more likely if there is low uncertainty in the belief. Because the belief is part of the state, the seeker can be penalized when the belief uncertainty is large. The seeker is penalized for near-collisions, which occcur if $\|x_t - \theta_t\| < d$, where d is a distance threshold. The following cost function penalizes belief entropy and near-collisions:

$$J(s_t) = H(b_t) + \lambda \mathop{\mathbb{E}}_{b_t} \mathbb{1}(\|x_t - \theta_t\| < d), \tag{5.4}$$

where $H(b_t)$ is the entropy of belief b_t, $\mathbb{1}$ is an indicator function that equals 1 if its argument is true and 0 otherwise, and the weight λ encodes the tradeoff between tracking and collision avoidance. A higher value of λ represents a higher penalty on near-collisions. The collision penalty is the expecation over all particles in the current belief.

Only the particle positions are used when computing belief entropy, capturing position uncertainty. To compute entropy from the particle filter, the particles are binned into M grid cells. The resulting discrete distribution is denoted $\tilde{b}_t$. Entropy is computed with

$$H(b_t) = -\sum_{i=1}^{M} \tilde{b}_t[i] \log \tilde{b}_t[i], \tag{5.5}$$

where $\tilde{b}_t$ is the proportion of particles in bin i.

5.2.3 Solution Method

To solve the MDP, the UCT variant [87] of Monte Carlo tree search (MCTS) is used. As its name implies, MCTS generates a tree from the current state s_t by

running simulations to evaluate the cumulative cost of different control inputs. After simulating, the lowest-cost control input is selected.

A drawback to using MCTS for belief-MDPs is that each simulation step requires a belief update, which can be computationally expensive [85]. One solution is to use fewer particles, but this can lead to poor target estimation. The compromise adopted here is to downsample the particle filters before running MCTS to generate a control input. The seeker maintains the higher-fidelity belief for target estimation, but uses the downsampled belief for efficient planning.

5.3 Simulations

The planner is validated with simulations. The near-collision threshold is $d = 15\,\text{m}$. The timestep duration is $\Delta t = 1\,\text{s}$, after which a new measurement is made and a new control input is generated. The seeker drone can travel at $5\,\text{m/s}$ and rotate at $15\,^\circ\text{/s}$. The target drone starts in one corner of a $200\,\text{m} \times 200\,\text{m}$ search area and travels across it at $1.7\,\text{m/s}$. The particle filter has 8000 particles and is initialized with random positions and velocities. For MCTS, these beliefs are downsampled to 200 particles before planning the next control input, and 1000 simulations with a timestep horizon of $T = 10$ steps are used to generate the next action.

The value of λ is varied for both the greedy and MCTS methods. For each value of λ, 1000 80-timestep simulations are run, and the resulting near-collision rate and the mean tracking error are logged. The near-collision rate is the proportion of timesteps that a near-collision has occured. The mean tracking error is the average Euclidean distance per timestep between the particle filter position mean and the true target drone position. This error is only measured after timestep 20 to avoid skew from the large uncertainty of the initial uniform particle distribution.

The results are shown in Figure 5.1. MCTS outperforms the greedy strategy for most values of λ; for the same near-collision rate, MCTS can offer a tracking error reduction of about $5\,\text{m}$, which is often a reduction of over 20%.

Pareto dominance eludes MCTS because it performs worse when λ is small. One explanation is that the optimal policy is less complicated when near-collisions are not

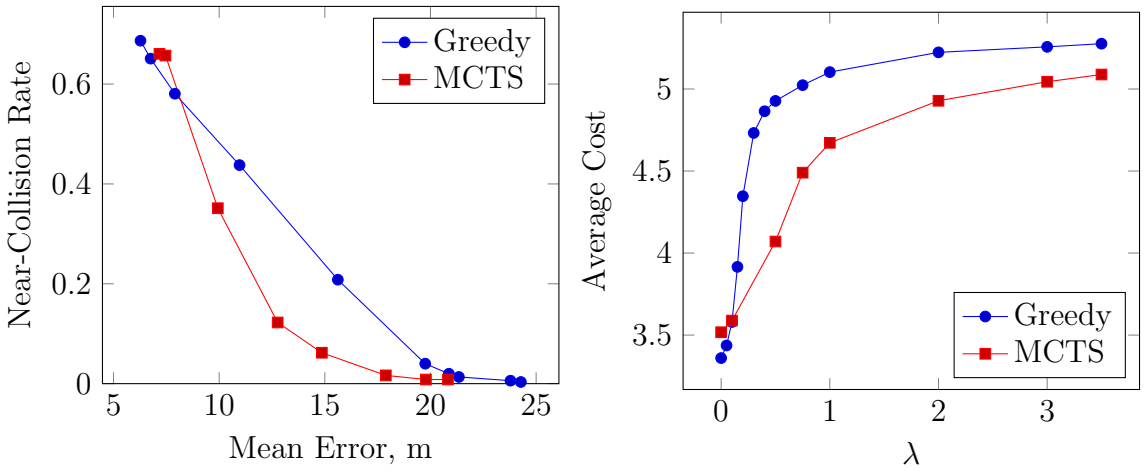

Figure 5.1: Comparison of greedy and MCTS methods. Left: human-readable performance metrics. Right: objective function costs against λ.

penalized. Both the greedy and MCTS policies lead the seeker drone to fly close to the target drone, where the best measurements are made. No long-term planning is needed as the seeker stays close to the target. Instead, small adjustments are made in the vicinity of the target to get the most information. The greedy method, which uses the full particle set in its planning, is able to make slightly more efficient adjustments because it plans with the particle set used for localization. In contrast, the MCTS method uses the lower-fidelity particle set when planning, and can only estimate transition probabilities between beliefs from transitions observed in its simulations. Therefore, its adjustments in the vicinity of the target are worse.

In contrast, MCTS performs much better when near-collisions are penalized. It is likely that the optimal policies in this case are more complicated. For example, it might make sense to be risky early on, flying near the target to get a good estimate. The seeker drone can then stay conservatively far away, with a good target estimate. The greedy method is unequipped to make these calculations, as it only plans one step into the future. Indeed, when observing greedy trajectories, the seeker drone often gets "stuck", where the only action that immediately reduces belief uncertainty carries some risk of near-collision. If the drone could plan farther ahead, it might see the small near-collision risk is worth the large reductions in belief uncertainty several

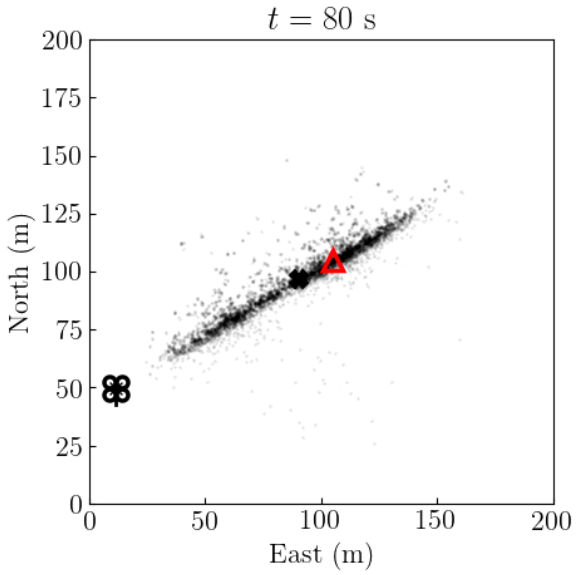

Figure 5.2: An example of the greedy policy getting "stuck" in beliefs with high uncertainty; it cannot plan far enough into the future to see the highly informative regions orthogonal to the long axis of the belief.

steps into the future. Figure 5.2 shows an example of this behavior. In contrast, the MCTS method can see the highly informative beliefs that might take several steps to reach; as a result, it leads to more concentrated beliefs and better estimates.

MCTS actions take longer to generate; the mean MCTS action time was made in 0.12 s compared to 0.02 s for the greedy method on a laptop with an i7 processor. But this time penalty is acceptable if measurements arrive at 1 Hz.

5.3.1 Effect of Planning Horizon

A key parameter in MCTS is the depth of the search tree, also called the planning horizon T. Generally, a deeper tree performs better (although not necessarily [88]), as it allows the agent to evaluate the effects of its actions further into the future. Of course, this improvement comes at more computational expense. Theoretically, the computational expense of MCTS grows linearly with the search tree depth.

Figure 5.3 shows the effect of the planning horizon on tracking performance. The

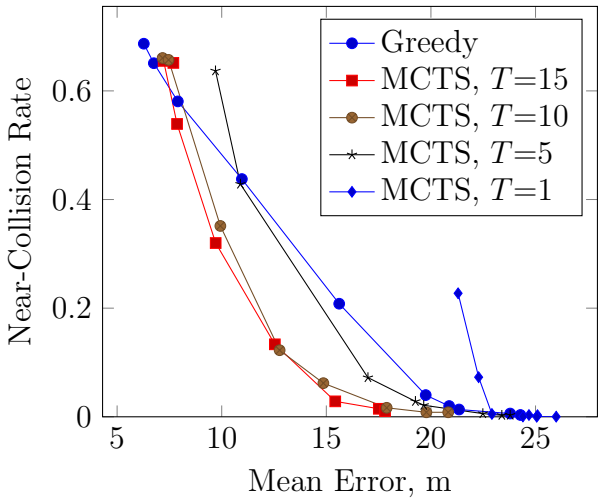

Figure 5.3: Effect of planning horizon on MCTS performance.

results satisfy intuition. First, performance generally increases as the horizon increases. Second, MCTS underperforms the greedy strategy when the planning horizon is 1. Theoretically, these planners should perform the same, as a greedy strategy also only looks one step into the future. However, the MCTS policy is only an approximation and uses the downsampled belief during planning. Therefore, it is reasonable that it performs worse than the exactly computed greedy strategy.

5.3.2 Effect of Downsampling

Another key parameter is the number of particles in the downsampled belief. Figure 5.4 shows the results of simulations for different particle counts in the downsampled belief; 1000 simulations were run for each setting. Performance generally improves with the number of particles. The most particles used, 1000, results in 12.5% of the full particle set used for localization. But performance when using only 50 particles (0.65% of the full set) is only slightly worse. Because computational expense scales linearly with the number of particles in the belief, the time to generate an action with only 50 particles is 0.05 the time required of 1000 particles. As a result, using fewer particles to improve solution performance seems to be a good tradeoff. It is not until the downsampled particle sets contain fewer than 50 particles that the

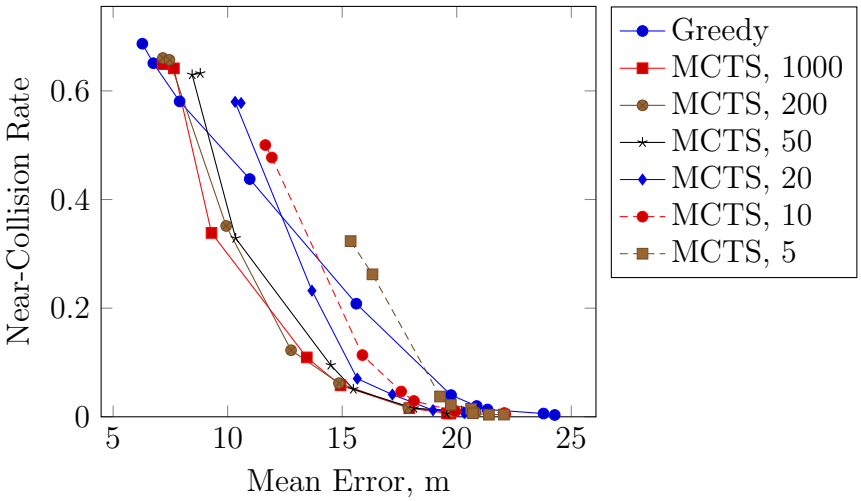

Figure 5.4: Effect of particle count in downsampled belief.

degradation becomes severe.

The results suggest solution quality is robust to downsampling. A possible explanation for this is that MCTS replans after each observation. Even if a reduced particle set converges to a poor target estimate during the MCTS simulations, the drone only acts for one step according to this poor estimate. Once that step is taken and a new observation is received, MCTS is fed a new subset of particles from the full set. In contrast, reducing the number of particles used for localization (the full set) would have a cumulative effect. Therefore, maintaining one particle set for localization and another for planning can work well.

5.4 Flight Test

The online algorithm was tested in a flight test with two drones. The seeker drone, the M-100, was used to localize a target drone, a DJI F550, by its telemetry radio. Both drones are shown in Figure 5.5. The target drone flew south at 1 m/s, and the seeker drone was limited to 5 m/s and 15 °/s. Measurements were collected and new control inputs were generated at 1 Hz. Figure 5.6 shows the resulting trajecotry. The seeker drone tracked the target's position (with some error) and avoided near-collisions.

Figure 5.5: M-100 seeker drone (left) and F550 target drone (right).

This result is limited because the drones move slowly, the flight is short, and not enough flights were run for a quantitative analysis. However, the flight test is meaningful in that nothing is simulated or post-processed—the measurements were taken by the drone, and the drone trajectories come from their GPS logs. The seeker drone performed filtering and selected its actions in real-time.

5.5 Discussion

This chapter explored the use of online solvers for a drone localizing a radio source. It shows that Monte Carlo tree search can reduce tracking error while reducing the number of near-collisions with the target. Even if the target is not flying (making near-collisions impossible), reducing the time spent directly over the target might be desirable. For example, flying directly overhead might scare radio-tagged wildlife.

The successful tracking of a moving target drone by its telemetry radio has important practical implications for protecting critical infrastructure from unauthorized drone flights. Detection and tracking form a critical layer in a "defense in depth" approach to countering drones [89]. Cameras offer an intuitive solution for tracking

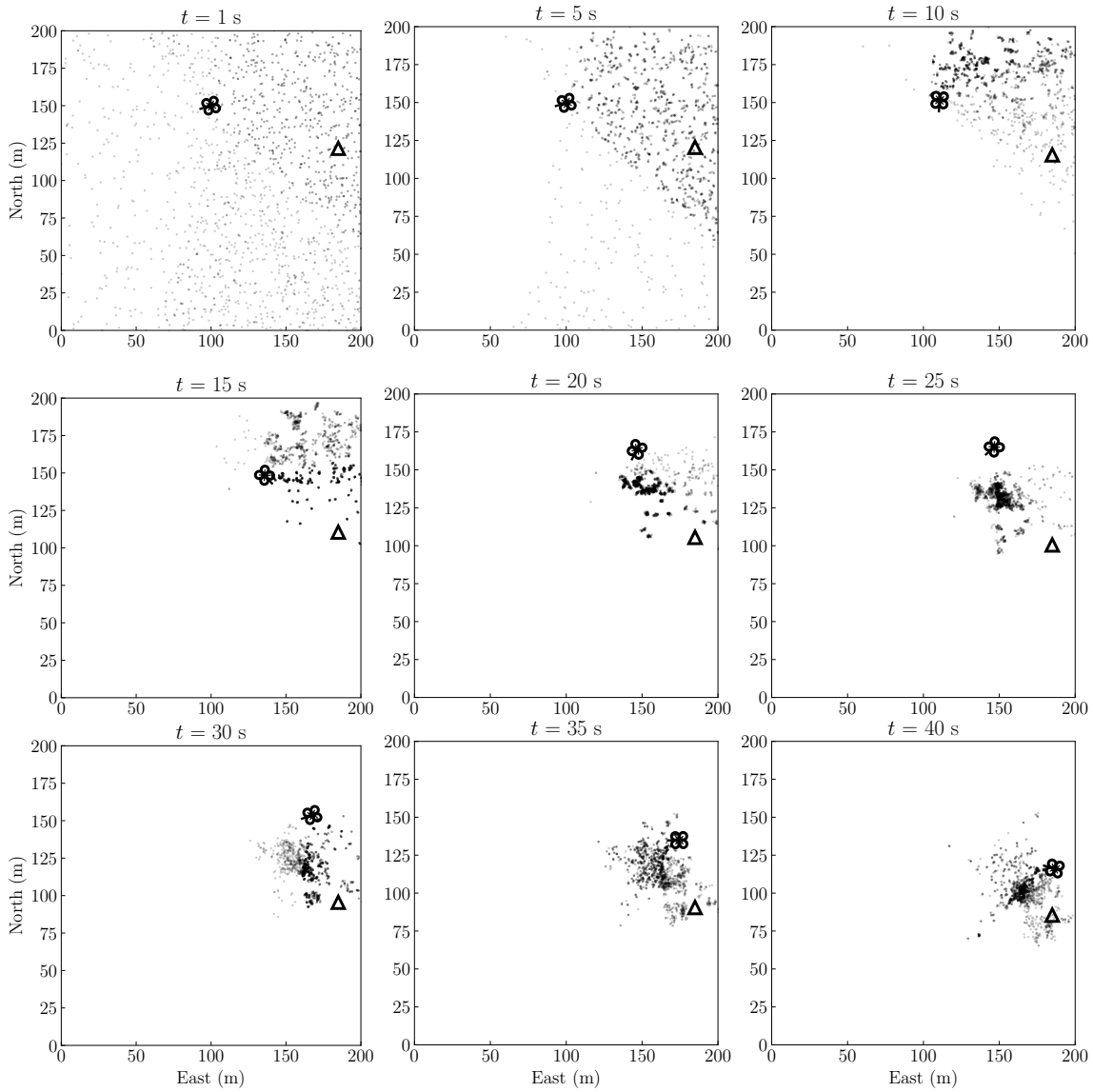

Figure 5.6: Flight test trajectory: the seeker drone tracks the target drone (triangle) as it moves south.

drones, but vision-based solutions struggle to differentiate drones and birds, especially when birds glide [90]. While not all drones emit telemetry, radio tracking is a useful tool to find those that do. Because analyzing drone telemetry signals is difficult, most research focuses on detection but not tracking [91], [92]. This work shows that simple hardware can be used to track a moving drone.

Chapter 6

Ergodic Control for Information Gathering

In the last two chapters, optimal belief-space planning techniques were applied to the problem of drone-based radio localization. While these techniques are principled, they present computational difficulties and are difficult to implement on robots. In robotics, heuristic methods are often used instead of principled methods when the computational cost is excessive. While these methods are not provably optimal or even approximately optimal, they often perform well in practice and are easier to implement on real systems.

Ergodic control is one such heuristic method that has been applied to the challenging problem of information gathering and active sensing. This method is based on the intuitive idea of taking sensor measurements from an area in proportion to the estimated information there. Ergodic control has shown promising experimental results and is generally easier to implement than principled belief space planning techniques.

This chapter presents background information on ergodic control and explores its recent use in the context of information gathering. In addition, conditions are formulated for the optimality of ergodic control for information gathering tasks. Ultimately, these conditions are limited, but they represent the first investigation into analyzing the potential optimality of ergodic control.

6.1 Background

Ergodic theory is a complex mathematical field that studies the long-term average behavior of systems [93]. Typically, averages over time and some state space are measured and compared, and we might call a system ergodic if its time-averaged statistics match some statistics averaged over the state space. Ergodic theory has been applied to statistical mechanics and fluids. For example, if we measured a particle's position over many timesteps, its average position might represent the distribution of all particles; the average position of all particles at an instant should match the long-term average position of a single particle. This level of understanding suffices for our purposes; for a more thorough review and comprehensive list of references, see Chapter 2 of Lauren Miller's thesis [94].

In the context of mobile robot trajectories, ergodicity has been applied to compare a robot's trajectory to some spatial distribution. A trajectory is ergodic with respect to this distribution if its time-averaged statistics match the distribution's spatial statistics; the distribution representing the robot's position should match the spatial distribution. In other words, the robot spends time in a region proportional to the distribution's density in the region. Figure 6.1 compares a trajectory ergodic with a distribution and a trajectory maximizing time spent in high density regions. The bimodal distribution has twice the density in one mode, and an ergodic trajectory spends about twice as much time in the vicinity of that mode as in the other one.

How is ergodic control used for information gathering?

Ergodic control has recently been proposed for designing trajectories for mobile sensors [94]–[96]. This framework can be applied to general, nonlinear systems and has outperformed greedy methods in some experiments [96], [97]. Ergodic control is built on the notion of trajectory ergodicity. A trajectory is ergodic with respect to some distribution if time spent in a state space region is proportional to the distribution's density in that region. When using ergodic control for information gathering, the distribution used is an expected information density, which is a measure of information at a point in the sensor's state space.

Although ergodic control has shown promising experimental results, it has only

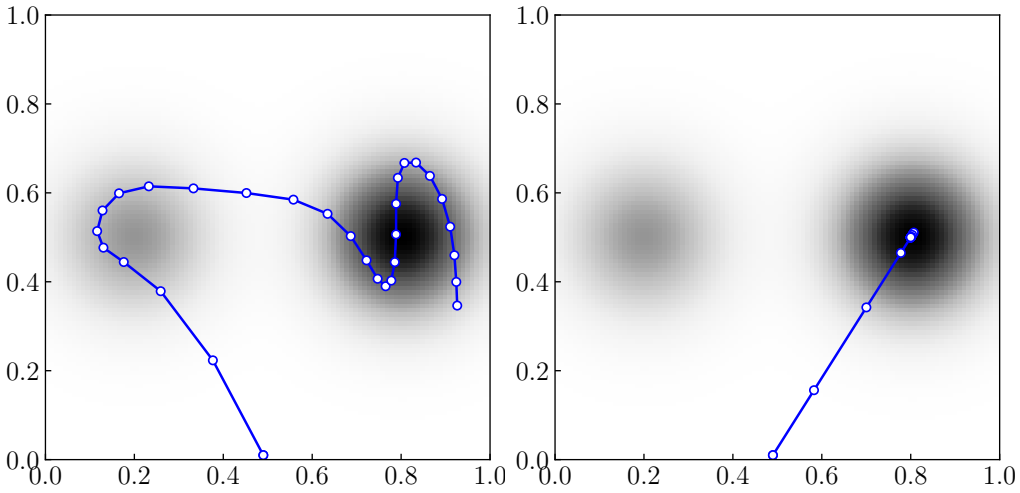

Figure 6.1: An example of trajectory ergodicity (left) and a trajectory that simply moves to the highest density point (right). Both trajectories start from $(0.5, .01)$.

recently been applied to information gathering tasks. It is not understood *why* ergodic control works well. Why does it make sense to spend time in a region proportional to its information density, instead of spending all our time in the most dense region? Selecting the length of an ergodic trajectory is another open research problem [95].

This chapter attempts to provide some insight into these fundamental questions of ergodic control. We present a problem class for which the optimal information gathering trajectory is ergodic. This class assumes measurement submodularity, where successive measurements from a state reduce the information available at that state. Specifically, the class assumes the rate of decay is linear. Under this assumption, selection of the ergodic optimization horizon for many systems becomes trivial. We use simple toy problems to validate these ideas and show the potential suboptimality of ergodic control when the assumptions do not hold. We generate ergodic trajectories for more complex problems to verify the connection between optimal information gathering, information decay, and ergodic trajectories.

6.2 Generating Erogdic Trajectories

Consider a domain $X \subset \mathbb{R}^s$ and a distribution $\phi : X \to \mathbb{R}$ that provides a density $\phi(\mathbf{x})$ at a state $\mathbf{x} \in X$. A trajectory of horizon T is a function $x : [0, T] \to X$. The state at time t according to trajectory x is denoted $x(t)$.

The time-averaged statistics of a trajectory are a distribution c over the state space, where the density at $\mathbf{x}$ is

$$c(\mathbf{x}) = \frac{1}{T} \int_0^T \delta(\mathbf{x} - x(t)) \, \mathrm{d}t, \tag{6.1}$$

where δ is the Dirac delta function. The factor $1/T$ ensures the distribution integrates to 1. Likewise, ϕ must be a valid density that integrates to 1 so that c and ϕ can be compared.

The goal in ergodic control is to drive c to equal ϕ. This goal is made explicit in an ergodic metric that measures the KL divergence between c and ϕ [98]. The KL divergence measures the similarity of two distributions. A different but widely-used metric decomposes c and ϕ into Fourier coefficients and compares the coefficients to each other [99]. The distribution is decomposed into Fourier coefficients $\phi_{\mathbf{k}}$:

$$\phi_{\mathbf{k}} = \int_X \phi(\mathbf{x}) F_{\mathbf{k}}(\mathbf{x}) \, \mathrm{d}\mathbf{x}, \tag{6.2}$$

where $F_{\mathbf{k}}$ is a Fourier basis function and $\mathbf{k} = [k_1, \ldots, k_s]$ is a multi-index used to simplify notation; $\phi_{\mathbf{k}}$ is short for $\phi_{k_1, k_2, \ldots, k_s}$. Each k_i ranges from 0 to K; there are $(K+1)^s$ coefficients in total. The coefficients $c_{\mathbf{k}}$ of trajectory x are

$$c_{\mathbf{k}}(x) = \frac{1}{T} \int_0^T F_{\mathbf{k}}(x(t)) \, \mathrm{d}t. \tag{6.3}$$

The ergodic metric $\mathcal{E}$ is a weighted sum of the squared difference between trajectory and distribution coefficients:

$$\mathcal{E}(x) = \sum_{\mathbf{k}} \Lambda_{\mathbf{k}} \left\| c_{\mathbf{k}}(x) - \phi_{\mathbf{k}} \right\|^2, \tag{6.4}$$

where $\sum_{\mathbf{k}}$ is short for $\sum_{k_1=0}^{K} \dots \sum_{k_s=0}^{K}$ and weights $\Lambda_{\mathbf{k}}$ favor low-frequency features. This metric has been used in feedback laws that drive trajectories toward ergodicity [99].

Strictly speaking, trajectories are only ergodic if $c \to \phi$ as $T \to \infty$ [99]. However, we follow recent work and call trajectory x ergodic if $\mathcal{E}(x)$ is small, even for finite horizons. Projection-based trajectory optimization (PTO) is one way to design ergodic trajectories for a given horizon T [100]. This method can be used for general nonlinear systems and the resulting ergodic trajectories have been used in information gathering tasks [95]. In these tasks, the distribution ϕ is an expected information density (EID) that represents the value of making a measurement from a specific state. The EID can be generated from information-theoretic concepts such as Fisher information or expected entropy reduction.

An ergodic trajectory is open-loop—a trajectory is designed for an EID, but this distribution changes as measurements are made and the belief is updated. To take advantage of this updated information, an MPC framework can be used [95]. First, an ergodic trajectory is generated for planning horizon T. Then some or all of that trajectory is executed, and measurements are collected. The belief and EID are updated, and a new ergodic trajectory is generated for planning horizon T. This approach leverages the ability to plan entire trajectories while incorporating updated information. Because ergodic trajectory generation can be computationally expensive, the execution horizon is often as large as the planning horizon [94]–[96].

It has been claimed that ergodic control effectively balances exploration and exploitation of information—more time is spent at information dense regions, but less dense regions are also explored [96]. Empirically, ergodic control seems like a viable choice for localization tasks. When compared to greedy, information-theoretic methods, ergodic control has slightly underperformed when noise is low, but significantly outperformed in environments with significant unmodeled noise [94]. At extraordinarily high levels of noise, ergodic control has underperformed uniform sweeps of the environment. When noise is so high as to render the model useless, it is reasonable to cover the space uniformly. Although it has slightly underperformed greedy and

uniform methods when noise is very low or very high, ergodic control generally performs well across noise regimes. The ability to adapt to concentrated information (low noise) or diffuse information (high noise) is a benefit of ergodic control.

6.3 Optimality and Submodularity

An optimal information gathering trajectory maximizes $\mathcal{I}(x)$, the information gathered by trajectory x, while adhering to dynamic or time constraints. On the surface, it is not clear why an ergodic trajectory would maximize $\mathcal{I}(x)$. If $\phi(\mathbf{x})$ represents the information at point $\mathbf{x}$, directly maximizing $\int_0^T \phi(x(t))\,\mathrm{d}t$ seems reasonable. This strategy would direct the sensor to the point with highest information density, instead of distributing measurements ergodically. To justify ergodic behavior, we look to submodularity.

6.3.1 Submodularity

In the context of information gathering, measurement submodularity refers to the notion that repeated measurements from a given location are successively less informative [96]. Formally, we say this submodularity is present if

$$\mathcal{I}(x_a + x_b) \leq \mathcal{I}(x_a) + \mathcal{I}(x_b), \tag{6.5}$$

where $x_a + x_b$ is the concatenation of trajectories x_a and x_b [101].

Submodularity is present in many information gathering tasks and must be accounted for to prevent solely and repeatedly sampling the maximally dense point [96]. If the sensor only samples this point, and the information there becomes depleted, the total information gathered along the trajectory might be low. In one information gathering example with a discrete number of states, the planner assumes a state's information is depleted after a single measurement, preventing sensors from staying at the information maxima [101]. Another way to handle submodularity is to plan for a single step. In greedy, one-step trajectory planners, the belief and EID can be

updated after each measurement, thereby incorporating submodularity and preventing a sensor from sampling a point with depleted information. By only planning for the next measurement location, the planner can ignore submodularity induced by an entire trajectory. However, when planning an entire trajectory for an initial EID, we need something to handle the submodularity.

In this context, it seems that ergodic control might be one way to incorporate submodularity into trajectory generation. In ergodic control, a trajectory is generated for an initial EID, which becomes stale as soon as the sensor starts making measurements. It is possible to update the EID and replan with MPC, but this can be computationally expensive. Because previous research uses relatively long execution and planning horizons, we focus on a single ergodic trajectory generated from an initial EID.

Submodularity seems to be a possible justification for ergodic control. We next examine a particular type of submodularity that best justifies ergodic trajectories.

6.3.2 Example and Problem Class

Suppose a sensor is in a domain where information is concentrated at two states. The left state has an information density of 80%, and the right state has a density of 20%. By definition, an ergodic trajectory splits its time proportionally to this ratio, and this falls out of the metric in Equation (6.4). Perfect ergodicity (i.e., $\mathcal{E} = 0$) can be achieved if $c_{\mathbf{k}} = \phi_{\mathbf{k}}$:

$$\frac{1}{T} \sum_{\mathbf{x}_d \in X_d} \tau(\mathbf{x}_d) F_{\mathbf{k}}(\mathbf{x}_d) = \sum_{\mathbf{x}_d \in X_d} \phi(\mathbf{x}_d) F_{\mathbf{k}}(\mathbf{x}_d),$$

where X_d is a discrete set of states with nonzero information, $\tau(\mathbf{x}_d)$ is the time spent in state $\mathbf{x}_d$, and $\phi(\mathbf{x}_d)$ represents the information at $\mathbf{x}_d$. Equality holds when

$$\frac{\tau(\mathbf{x}_d)}{T} = \phi(\mathbf{x}_d).$$

That is, perfect ergodicity is achieved if the proportion of time spent at $\mathbf{x}_d$ is equal to the information at that location. In our example, the sensor spends $0.8T$ in the

left state and $0.2T$ in the right. After spending $0.8T$ at the left state, the ergodic trajectory never returns. One situation where this behavior is optimal is if the state is stripped of information after $0.8T$. Then, the 20% state will contain more information after $0.8T$, and an optimal trajectory will spend the rest of the time there.

Using the above example as a guide, we claim that an ergodic trajectory minimizes the time to gather all available information in a domain if the following model for information collection and submodularity holds:

1. Information is collected (and depleted) from a state when a sensor spends time there.

2. Information is collected from all states at the same rate: $1/T$ per unit time for a continuous trajectory and $1/N$ per time step for a discrete trajectory with N steps.

3. The information available at state $\mathbf{x}$ is equal to $\phi(\mathbf{x})$. In a discrete domain, we assume $\sum_{\mathbf{x}_d \in X_d} \phi(\mathbf{x}_d) = 1$ (the analog to $\int_X \phi(\mathbf{x})\, \mathrm{d}\mathbf{x} = 1$ in the continuous case).

6.3.3 Time Horizon Selection

Our problem class requires a linear collection (and depletion) of information. If we know the rate at which information is collected at, we can choose the ergodic trajectory horizon to efficiently collect the available information.

Assume we have the same two-state example from the previous subsection, where the left and right states have 0.8 and 0.2 units of information, respectively. Assume further that we know the collection rate is 0.1 per step; at each step, the sensor collects 0.1 information units from its current state. There is a cost to switch between the states and the sensor starts in the left state.

The trajectory that minimizes the time and cost to collect all information is 10 steps long. The sensor spends its first eight steps in the left state and its last two in the right state. This perfectly ergodic trajectory collects all information available while minimizing the switching cost.

If we instead generated a 20-step ergodic trajectory minimizing control cost, the resulting trajectory would spend 80% of its time in the left state and 20% in the right—so, 16 steps in the left state followed by four in the right. After its first eight steps in the left state, the sensor would deplete all available information there. It would collect no new information until switching to the right state. Eventually, all information would be collected, but it would have taken roughly twice as long as with the 10-step horizon.

If we picked a shorter horizon, like five steps, a perfectly ergodic trajectory would spend four steps in the left state and then one in the right. However, at the end of this trajectory, the sensor would only have collected half the available information—the left state would still have 0.4 and the right would have 0.1. The sensor could execute another five-step ergodic trajectory starting from the sensor's last position (the right state). This trajectory would spend one step in the right state followed by four in the left. After the two five-step trajectories, all information is collected—just as it was at the end of our single 10-step trajectory. However, the sensor incurs twice the cost by switching states twice, using two sweeps to cover the space. Further, two ergodic trajectories are computed instead of one, which can be expensive.

By selecting a horizon for our ergodic trajectory, we assume a decay rate. If this rate matches the true decay rate, we can minimize the time required to collect all available information. In many dynamical systems, a trajectory with this carefully selected horizon will also minimize the control effort required to gather all information, as it did in our example. However, this is not the case with all dynamical systems. For example, an oscillating system might trade time for energy use. In these systems, an ergodic trajectory might exert extra control effort to drive the sensor to distribute measurements ergodically.

6.3.4 Example Outside the Class

Consider two observation posts on either side of a runway. An observer estimates the distance to an approaching aircraft. From either post, the observer measures the true distance corrupted with zero-mean Gaussian noise. The left post offers the best

view, while the right post is blocked by trees. As a result, the Gaussian noise of the left post has standard deviation σ_{small}, and the noise of the right has $\sigma_{\text{large}} > \sigma_{\text{small}}$.

Both posts have non-zero information density—from either, enough noisy measurements can be stitched together to give a low variance distance estimate. However, more observations are required from the right (noisier) post. The optimal search trajectory makes all measurements from the left post. However, an ergodic trajectory would spend some time in the right post because it has non-zero information density. The linear information decay assumed in our problem class implies all information will be "used up" from the left post after some fraction of the time horizon. As a result, an ergodic trajectory reserves some time for the right post.

The ergodic trajectory is suboptimal because it falls outside of our problem class. The sensor model implies measurements from the left post are *always* more informative than those from the right, regardless of the time spent there.

6.3.5 Analysis of the Ergodic Metric

So far, we have provided intuitive arguments for the connection between submodularity and the optimality of ergodic trajectories. In this section, we provide a theoretical argument using the Fourier-based ergodic metric.

Before proceeding, consider two preliminaries. First, the Fourier transform is linear with respect to distributions. That is, if $z, y \in \mathbb{R}$, and ϕ^1 and ϕ^2 are two distributions, then

$$\phi = z\phi^1 + y\phi^2 \iff \phi_{\mathbf{k}} = z\phi^1_{\mathbf{k}} + y\phi^2_{\mathbf{k}}.$$

Second, when adding two distributions, we add the densities at each point; scaling a distribution scales the density at each point. When adding or scaling distributions, the resulting distributions will not integrate to 1, so care must be taken when performing these operations.

Our argument proceeds as follows. Suppose we desire a trajectory with horizon $T = T_a + T_b$ that is split into two partial trajectories x_a and x_b. Suppose x_a has already been executed for its horizon T_a. This partial trajectory has a spatial distribution c^a and coefficients $c^a_{\mathbf{k}}$, each of which are normalized by horizon T_a. We want to design

the remainder of the trajectory, x_b, for the remaining horizon T_b so that the entire trajectory $x = x_a + x_b$ is ergodic. The coefficients for each partial trajectory are

$$\begin{aligned} c_{\mathbf{k}}^a &= \frac{1}{T_a} \int_0^{T_a} F_{\mathbf{k}}(x(t))\, \mathrm{d}t, \\ c_{\mathbf{k}}^b &= \frac{1}{T_b} \int_{T_a}^{T_a+T_b} F_{\mathbf{k}}(x(t))\, \mathrm{d}t. \end{aligned} \tag{6.6}$$

The coefficients for the entire trajectory are a weighted average of the coefficients for the individual trajectories:

$$\begin{aligned} c_{\mathbf{k}} &= \frac{1}{T_a + T_b} \int_0^{T_a+T_b} F_{\mathbf{k}}(x(t))\, \mathrm{d}t \\ &= \frac{1}{T_a + T_a} \left(T_a c_{\mathbf{k}}^a + T_b c_{\mathbf{k}}^b \right). \end{aligned} \tag{6.7}$$

The objective function then becomes

$$J(x_b) = \sum_{\mathbf{k}} \Lambda_{\mathbf{k}} \left(\frac{T_a c_{\mathbf{k}}^a + T_b c_{\mathbf{k}}^b}{T_a + T_b} - \phi_{\mathbf{k}} \right)^2. \tag{6.8}$$

We can reorder this objective so it becomes

$$J(x_b) = \left(\frac{T_b}{T_a + T_b} \right)^2 \sum_{\mathbf{k}} \Lambda_{\mathbf{k}} \left(c_{\mathbf{k}}^b - \phi'_{\mathbf{k}} \right)^2, \tag{6.9}$$

where

$$\phi'_{\mathbf{k}} = \frac{T_a + T_b}{T_b} \left(\phi_{\mathbf{k}} - \frac{T_a}{T_a + T_b} c_{\mathbf{k}}^a \right). \tag{6.10}$$

We drop the scale factor, yielding the equivalent objective

$$J(x_b) = \sum_{\mathbf{k}} \Lambda_{\mathbf{k}} \left(c_{\mathbf{k}}^b - \phi'_{\mathbf{k}} \right)^2. \tag{6.11}$$

Therefore, designing x_b to minimize Equation (6.8) is equivalent to designing x_b to minimize Equation (6.11). We are effectively designing x_b to be ergodic with respect to a new distribution ϕ', whose coefficients are $\phi'_{\mathbf{k}}$. Because of the linearity of the

Fourier transform, the modified distribution ϕ' is similar to the modified coefficients $\phi'_{\mathbf{k}}$:

$$\phi' = \frac{T_a + T_b}{T_b}\left(\phi - \frac{T_a}{T_a + T_b}c^a\right). \tag{6.12}$$

The distribution ϕ' results from the effect of partial trajectory x_a and its corresponding distribution c^a on the original distribution ϕ. The quantity inside the parentheses of Equation (6.12) is equal to the original distribution minus a scaled version of c^a; the scale factor is equal to the proportion of time spent in trajectory x_a.

However, the distribution in the parentheses of Equation (6.12) is invalid because it does not integrate to 1. If we are designing x_b to be ergodic with respect to spatial distribution ϕ, we normalize c^b and ϕ so we can compare them. The linearity of the Fourier decomposition implies

$$\int_X \left(\phi(\mathbf{x}) - \frac{T_a}{T_a + T_b}c^a(\mathbf{x})\right) \mathrm{d}\mathbf{x} = \frac{T_b}{T_a + T_b}. \tag{6.13}$$

Therefore, we have the normalization term $(T_a + T_b)/T_b$ in Equation (6.12), ensuring ϕ' integrates to 1.

We have shown that the ergodic objective from Equation (6.4) reduces the values of states in which time has already been spent, proportional to the time spent there; this result matches the conditions presented in Section 6.3.2.

These results satisfy an intuitive result: if $T_a = T_b$ and $c^a_{\mathbf{k}} = \phi_{\mathbf{k}}$, then $\phi'_{\mathbf{k}} = \phi_{\mathbf{k}}$. That is, if the partial trajectory x_a is perfectly ergodic, then x_b should be ergodic with respect to the same distribution in order for the whole trajectory to be ergodic. The trajectory x_a collects half the information available at every state, so it makes sense to perform a similar sweep over the domain to retrieve the remaining information.

Consider another intuitive result. From Equation (6.12), $\phi'(\mathbf{x}) < 0$ if

$$\phi(\mathbf{x}) < \frac{T_a}{T_a + T_b}c^a(\mathbf{x}).$$

If this is the case, we have oversampled point $\mathbf{x}$ during partial trajectory x_a and it is impossible to rectify this in the remaining horizon T_b [102]. It is possible to overcome this oversampling by increasing the horizon T_b, which would ensure a smaller scale

applied to $c^a(\mathbf{x})$.

6.3.6 Spatial Correlation

Our intuitive examples used domains where information is concentrated in a discrete set of states so we could observe the effect of sampling from a state. This observation is more difficult in a continuous domain. Even with noiseless dynamics, the agent cannot sample all states in a continuous domain in finite time. In a real scenario with noise, the vehicle will likely never return to the same exact state, so the notion of spending more time in a state is unrealistic.

These problems arise from use of the Dirac delta in the definition of the time-averaged statistics c, which sets the sensing footprint at any time to be a single state. An alternative is to encode a larger sensor footprint into c [98]. For example, if a sensor gathers information from all points within a radius of its current state, the time-averaged statistics c can be defined to reflect this. However, the bulk of existing work uses the Dirac delta, so we use it here.

Although the Dirac delta implies no spatial correlation between measurements, correlation is introduced by the ergodic metric, giving the sensor a footprint larger than a single state. We have assumed a perfect relationship between a spatial distribution ϕ and its coefficients $\phi_{\mathbf{k}}$—that is, decomposing ϕ into coefficients $\phi_{\mathbf{k}}$ and using these coefficients to reconstruct a spatial distribution would lead to ϕ. This interchangeability holds as $K \to \infty$, but real implementations use a finite number of coefficients, yielding a band-limiting effect on the representational power of the Fourier decomposition [94]. It has been posited that this effect can be beneficial as it allows for unmodeled uncertainty in the EID. We build on this idea, suggesting that fewer coefficients can add spatial correlation between vehicle states, as higher-order coefficients are needed to capture fine differences in a distribution or trajectory.

An example of this spatial correlation is shown in Figure 6.2. A discrete ergodic trajectory is generated for a simple Gaussian distribution. This trajectory is decomposed into sets of coefficients $c_{\mathbf{k}}$ for different numbers of coefficients K. These sets of coefficients are used to reconstruct spatial distributions of the trajectory. When

$K = 5$, the resulting spatial distribution of the trajectory looks fairly similar to the original Gaussian distribution. When $K = 30$, the spatial distribution more closely matches the trajectory. When $K = 150$, the spatial distribution is so similar to the trajectory that individual points along the trajectory are discernible. Visually, the coarse $K = 5$ distribution most closely matches the original spatial distribution. Even though a small number of states are visited in the trajectory, much of the state space has positive density because of the spatial correlation introduced by the small number of coefficients. In contrast, there is much less spatial correlation in the $K = 150$ distribution; only states in the near vicinity of the discrete trajectory's points have any density.

This spatial correlation affects the ergodic trajectories generated. Figure 6.3 shows trajectories generated for the same distribution ϕ, but one uses $K = 5$ coefficients and the other uses $K = 100$. The trajectories are generated using PTO until a descent direction threshold is reached [100]. The trajectory generated with fewer coefficients is more spread out, because the coarse decomposition implies greater spatial correlation.

Figure 6.4 shows an example of the partial-trajectory example from the previous subsection. Although the first partial trajectory only coarsely covers the lower-right mode, the modified spatial distribution suggests all information was gathered from the mode.

6.4 Information Gathering Experiments

We use two experiments to test the relationship between ergodicity, submodularity, and information gathering. In each experiment, ergodic trajectories are generated for a mobile sensor and an EID composed of one or two Gaussians. The EID covers the unit square, which is discretized into a 10×10 grid. The information in each cell is obtained from the EID.

At each time step, the sensor collects (and removes) information from the cell it occupies at a specified rate. If there is not enough information in the cell, the sensor collects whatever is left. Discretization implies spatial correlation between measurements, as measurements from any point in a cell affect future measurements

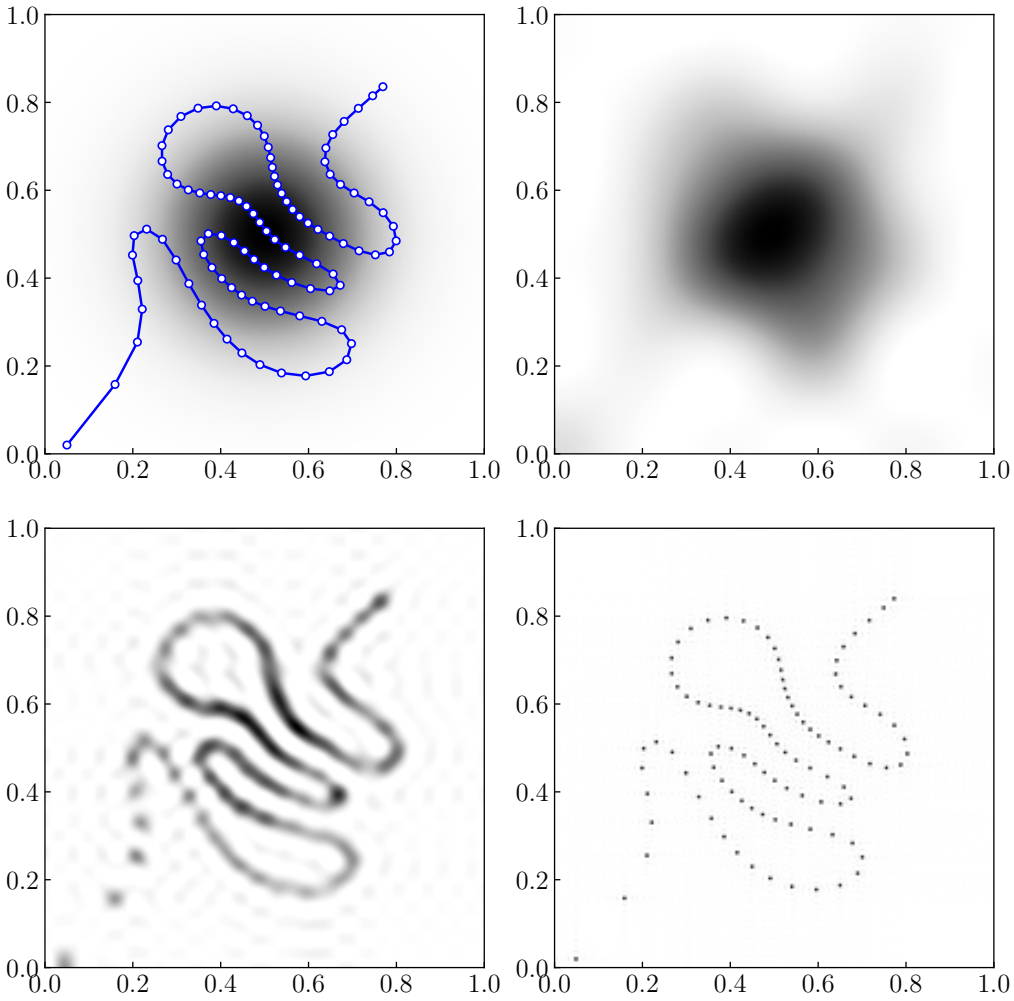

Figure 6.2: In the upper left, the original distribution and a trajectory designed to be ergodic with respect to it. The reconstructed distributions from this trajectory when using $K = 5$, $K = 30$, and $K = 150$ coefficients are shown in the upper right, lower left, and lower right, respectively.

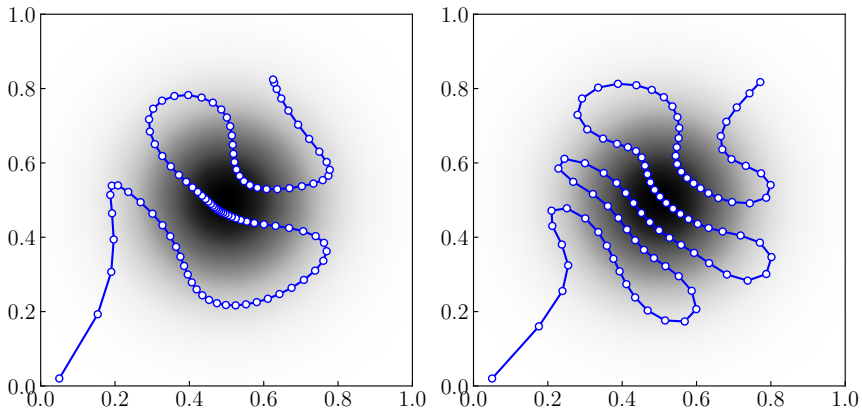

Figure 6.3: Trajectories generated to be ergodic with respect to a Gaussian distribution. The left trajectory was generated with $K = 5$ coefficients, and the right was generated with $K = 100$.

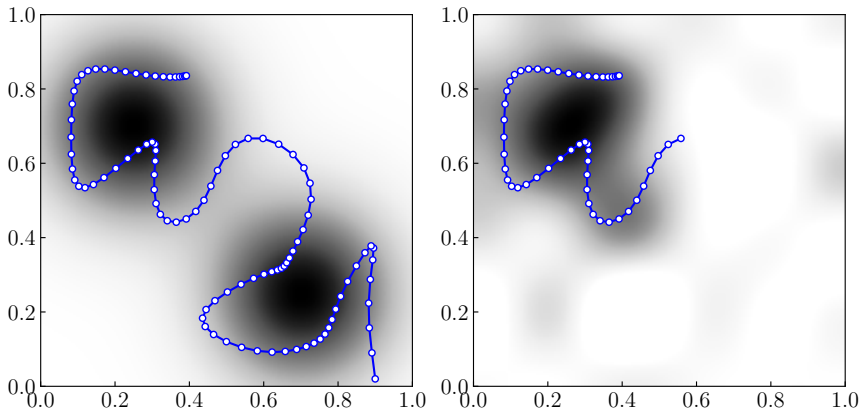

Figure 6.4: On the left, a trajectory ergodic with respect to a bimodal distribution ϕ starts in the lower right corner. On the right, we show the modified spatial distribution according to Equation (6.12) after half the trajectory is executed. The lower right mode is gone because all information was collected after the first half of the trajectory was spent there.

from any point inside the same cell. Note that this spatial correlation does not correspond to the spatial correlation implied by a finite number of coefficients. This domain is simple but not unlike others that have been used in information-gathering research [101].

We use PTO with $K = 50$ to generate discrete trajectories with $N = 100$ points starting from $(0.25, 0.35)$. Single integrator dynamics are used with a time step of 0.5 seconds.

6.4.1 Ergodic Score and Information Collected

We have claimed that perfectly ergodic trajectories are information-optimal under linear information submodularity. If our claim is correct, information gathered should increase as ergodic score improves (i.e., $\mathcal{E}$ decreases).

We terminate PTO at different ergodic scores and record the information collected by each trajectory. We compare against rapdily-exploring information gathering (RIG), a sampling-based motion planner that incorporates information submodularity [101]. We also generate an information-optimal trajectory that moves the sensor to the grid cell with the most information left. Information is collected from the cell, and the process repeats. The resulting trajectories are feasible only because of the simple dynamics; such a technique is not applicable to general systems. Figure 6.5 shows the relationship between ergodic score and information collected. Trajectories are shown in Figure 6.6.

As trajectory ergodicity increases, the information collected increases. The optimal trajectory collects the most information and has the lowest ergodic score, reinforcing the tie between ergodicity and information collection under our model. It is impossible to collect all the information with the finite number of discrete steps. However, if $N \to \infty$, the information-optimal trajectory approaches 100% information gathered and $\mathcal{E} \to 0$. RIG approaches optimality if sufficient points are used, so it performs well.

Both the information-optimal and RIG trajectories are generated with an exact model of the spatial correlation involved. In contrast, the PTO trajectories only

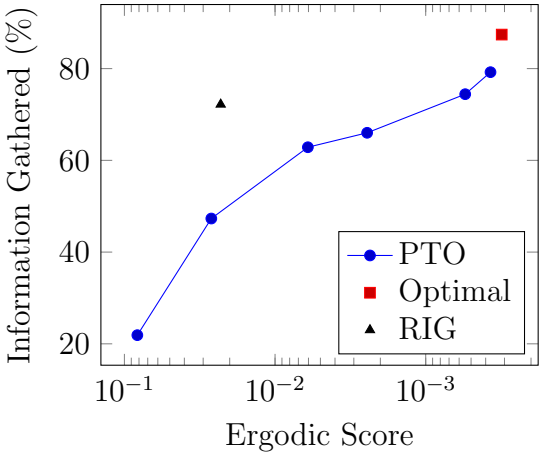

Figure 6.5: Information gathered as a function of ergodic score.

have a sense of the spatial correlation through the finite number of coefficients used. However, these trajectories are still competitive if solved to a low enough ergodic score. This result validates the claim that improved ergodicity leads to more information collected in problems with our model.

6.4.2 Trajectory Horizon and Information Collected

In Section 6.3.3, we claim that knowledge of the information collection (decay) rate informs selection of the trajectory horizon. Suppose we know the information collection rate is $1/N_f$ per time step, where N_f is a positive integer. When selecting a horizon N for our ergodic trajectory, we posit that N should match N_f for the most efficient trajectories. We set $N_f = 100$ and use PTO to generate a trajectory with $N = 100$. We then generate a composite trajectory consisting of two smaller ergodic trajectories, each with $N = 50$. Figure 6.7 shows the trajectories.

Both trajectories collect roughly the same information: 77.8% for the single trajectory and 82.6% for the composite. However, the single trajectory has a control effort ($\sum_{n=1}^{N} u_n$) of 3.6, and the composite has a control effort of 6.8. When the horizon is too short, not enough information is collected and a second pass is needed, increasing the cost. Thus, knowledge of the information collection rate can inform selection of the trajectory horizon.

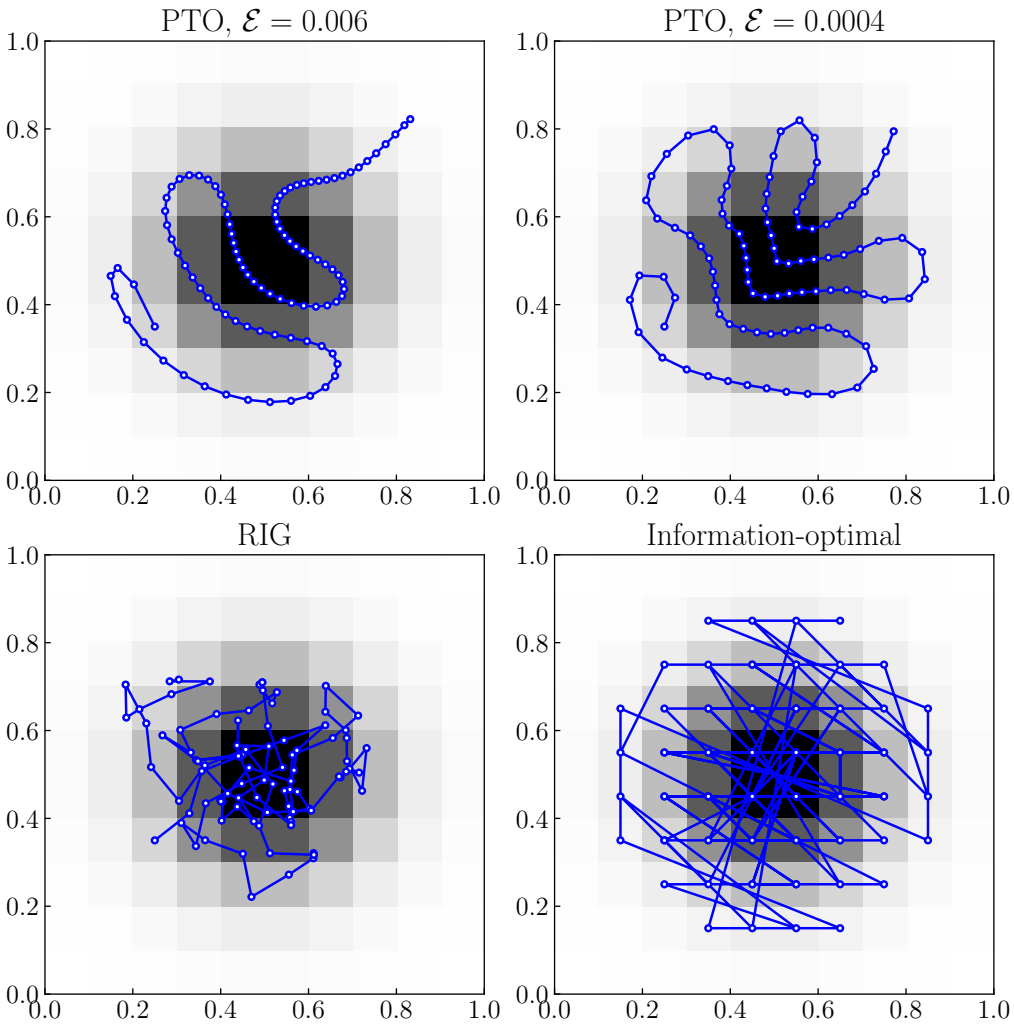

Figure 6.6: Trajectories generated with different methods collecting information in a discrete 10×10 grid.

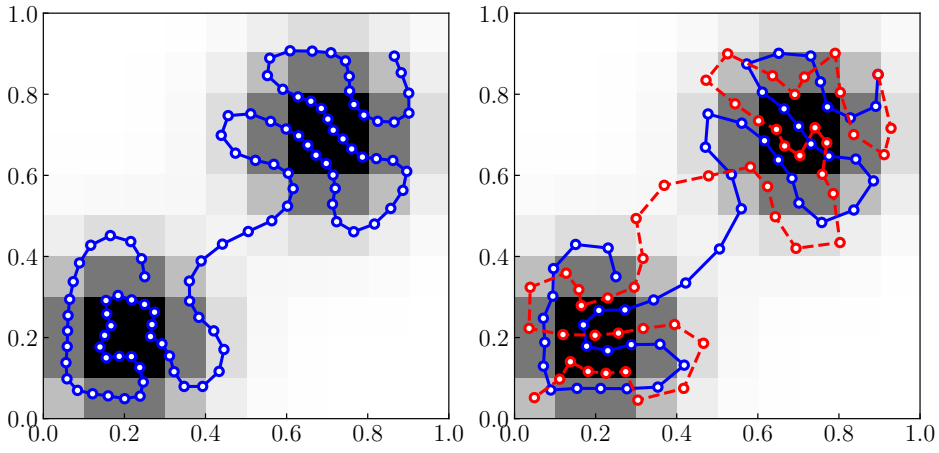

Figure 6.7: PTO ergodic trajectories. On the left, a single trajectory generated for horizon N_f. On the right, a trajectory of horizon N_f is composed of two trajectories each designed for a horizon of $N_f/2$. The first sub-trajectory is the solid, blue line. The second is the red, dashed line. The single trajectory on the left collects roughly the same information with about half the cost.

6.5 Discussion

This chapter introduced ergodic control and its recent use in information gathering tasks. Additionally, this chapter explored the situations under which ergodic trajectories might be optimal. The results suggest ergodic control is the optimal information gathering strategy under a specific model of information collection and submodularity. This model is unrealistic in many sensing tasks for two reasons. First, a measurement made in one location often decreases the information available at other locations. This spatial correlation is not captured by the traditional Dirac delta formulation in ergodic control; the spatial correlation resulting from a finite number of Fourier coefficients is unlikely to match the correlation of a real sensor. Second, this model assumes that while some states might have more total information, information is collected at the same rate from all states. But in real scenarios, measurements from certain states are often more informative than others. Further, the EID is often formulated to represent the value of a single measurement, such as mutual information or the expectation of Fisher information—rather than the total information at

a specific state. Because this proposed model is unrealistic in many tasks, it is likely that ergodic control is suboptimal for general information gathering tasks; although it is possible that other models are optimally solved by ergodic control.

Ergodic control does excel in some tasks, particularly when the proposed model of submodularity holds. Coverage problems in which an agent must surveil or cover an area uniformly are well served by ergodic control [99]. Coverage problems actually adhere to the proposed problem class; there is value to visiting uncovered states, but that value decreases linearly until these states have been as well covered as others. Ergodic control is also suitable for autonomous painting [103], because painting an image is effectively a coverage problem in which the time to spend at a state is governed by the image's darkness at that state. Because ergodic trajectories are distributed over a spatial distribution (rather than seeking out maxima or minima), they are robust to unmodeled sensor noise [94].

The next chapter describes how to reduce the computational complexity of ergodic control using approximations from neural networks. The chapter after that evaluates ergodic control in drone-based radio source localization.

Chapter 7

Generating Information Maps

One of the advantages of ergodic control is that it is not generally intractable like many belief-space planning techniques. However, "not intractable" does not mean "trivial", and it can still be difficult to compute ergodic trajectories in real-time on robotic platforms. This chapter proposes using neural networks to drastically reduce the time needed to generate information maps and their coefficients, key steps to generating ergodic trajectories.

7.1 Introduction

When using ergodic control for localiztion tasks, trajectories are designed to be ergodic with respect to a distribution ϕ. This distribution is an information map (also called an expected information density) describing how information is distributed over the localizing agent's state space.

As the agent executes an ergodic trajectory, it makes measurements and updates the distribution over possible target locations, which changes the information map. Therefore, ergodic control is typically implemented in a model-predictive fashion: ergodic trajectories are generated for a current information map, the agent executes part of this trajectory, and a new trajectory is generated when the information map is updated with a new measurement.

Therefore, implementing ergodic control on real robotic platforms requires that

information maps and ergodic trajectories be updated and computed in real-time. There has been much research into generating ergodic trajectories quickly, leading to algorithms that run in real-time [98], [102], [104], [105]. However, the problem of updating information maps has not been addressed. Information maps are constructed from the belief over target locations and a sensor model. It is often necessary to integrate over the agent, target, and observation spaces, which can be exorbitantly expensive. In practice, this problem is side-stepped by using Fisher information maps, which have lower computational expense. However, this method still struggles to scale and is not appropriate for all sensor models. This chapter proposes using neural networks to generate information maps from a belief over target locations. The resulting maps are generated quickly and are good approximations to the true maps. Neural networks are also used to learn the Fourier decopmoposition of information maps, allowing these coefficients to be quickly approximated from the belief.

The idea of using machine learning to speed up online computation is not new. In an early example, support vector machines determined if a robotic agent could reach other points in the state space [106]. Traditional approaches numerically solved a computationally expensive two-point boundary value problem to determine reachability, but this machine learning approach drastically reduced computation time.

Here, a machine learning approach is also used, but convolutional neural networks are used instead of support vector machines. Convolutional neural networks are a natural choice because the input is a distribution over possible target locations. This input is like an image and there is likely spatial correlation between points in the state space.

Convolutional neural networks have had stunning success classifying and modifying images, in large part because of spatial correlation in images [107], [108]. A convolutional layer is a set of filters that is convolved over the input image. These filters detect repeated features in the input. Typically, a few convolutional layers are stacked until fed into a fully connected layer for the output.

7.2 Model

Because this chapter's purpose is validation of the neural network technique, simplifications are made. These simplifications are for clarity and do not affect the neural network generation of maps and coefficients. The first simplification is that a stationary target is considered, so the target state is $\theta = [\theta^{\mathrm{n}}, \theta^{\mathrm{e}}]^\top$.

Two sensor models are considered in this chapter. The first is a bearing-only modality that provides estimates of the bearing to the target. These measurements might be achieved with a complex method like beam-steering, but this modality is included here because of its simplicity. Because the sensor heading does not affect the measurement, the sensor state space is $X \subset \mathbb{R}^2$ and the sensor state is denoted $x_t = [x_t^{\mathrm{n}}, x_t^{\mathrm{e}}]^\top$. As a result, output information maps are 2D and easier to visually evaluate. Further, this is favorable to the true methods as it is a smaller domain to integrate over. The measurement obtained at time t is

$$z_t = \beta_t + w_t, \tag{7.1}$$

where w_t is zero-mean Gaussian noise with standard deviation σ. Recall that the bearing to the target, measured east of north, is

$$\beta_t = \arctan\left(\frac{\theta^{\mathrm{e}} - x_t^{\mathrm{e}}}{\theta^{\mathrm{n}} - x_t^{\mathrm{n}}}\right). \tag{7.2}$$

The second sensor is the double-Moxon sensor from Section 3.4. In this modality, vehicle heading affects the measurements received, so the agent state space is in SE(2) and the state is denoted $x_t = [x_t^{\mathrm{n}}, x_t^{\mathrm{e}}, x_t^{\mathrm{h}}]^\top$. Recall that the observation set is $Z = \{0, 1\}$ and the probabilistic sensor model is defined

$$P(z_t = 1 \mid x_t, \theta) = \begin{cases} 0.9, & \text{if } \beta_t - x_t^{\mathrm{h}} \in [-60°, 60°] \\ 0.1, & \text{if } \beta_t - x_t^{\mathrm{h}} \in [120°, 240°] \\ 0.5, & \text{otherwise.} \end{cases} \tag{7.3}$$

When using this neural network technique, the target distribution is represented

as a discrete belief. Because a discrete belief is represented as an array, it can be thought of as an image where each cell is a pixel. This representation benefits the neural networks, which typically expect array inputs. If desired, particle filters can still be used for filtering purposes, as long as the particle sets are converted to discrete beliefs before feeding them into the neural networks. This conversion can be done without much computational cost; particles are assigned to the grid cell they fall in [36].

7.3 Generating Information Maps

The information map $\phi : X \to \mathbb{R}$ maps the state space to a measure of information quality or quantity. To feasibly compute the map, information values are computed at a discrete set of points $X_d \subset X$. The information at $x \in X_d$ is computed using quantities like mutual or Fisher information.

Computing information values typically requires integrating over the target space Θ, so this space is also approximated as a discrete set of points Θ_d. It is sometimes necessary to integrate over the measurement space Z. If this space is continuous, it is approximated by a discrete set Z_d. In bearing-only localization, where $Z = [0°, 360°)$, the discrete set is $Z_d = \{0°, 10°, \ldots, 350°\}$.

7.3.1 Mutual Information

Mutual information is often used to guide mobile sensors in localization tasks [16], [24]. The mutual information at a state is equal to the expected reduction in belief entropy resulting from a measurement there. Entropy captures the uncertainty in a distribution or random variable. Because the goal in localization is to reduce uncertainty about the unknown parameter θ, minimizing belief entropy is sensible.

Given two random variables A and B, the mutual information $I(A; B)$ is the amount of information obtained about one variable given the other is known. Equivalently, $I(A; B)$ gives the reduction in uncertainty of A given B is known or the reduction in uncertainty of B given A is known. Mutual information is symmetric so

$I(A;B) = I(B;A)$.

In localization, we are often interested in $I(b_t; z_{t+\Delta t})$, the reduction in uncertainty of b_t given the next measurement is known. Strictly speaking, b_t is a distribution and not a random variable; we abuse notation and use b_t to refer to the random variable describing the value of θ, which has distribution b_t. The measurement at the next timestep, $z_{t+\Delta t}$, is a random variable because it is an unknown quantity.

The value of $I(b_t; z_{t+\Delta t})$ is made explicit in the relation

$$I(b_t; z_{t+\Delta t}) = H(b_t) - H(b_t \mid z_{t+\Delta t}). \tag{7.4}$$

The quantity $H(b_t)$ is the current entropy of θ. The quantity $H(b_t \mid z_{t+\Delta t})$ is the conditional entropy of θ given the next measurement were known. We leverage the symmetry of mutual information:

$$I(z_{t+\Delta t}; b_t) = H(z_{t+\Delta t}) - H(z_{t+\Delta t} \mid b_t). \tag{7.5}$$

In greedy control, Equation (7.5) is evaluated for each $x_{t+\Delta t}$, or possible agent state at the next timestep. The first term is

$$H(z_{t+\Delta t}) = -\sum_{z \in Z_d} P(z_{t+\Delta t} = z) \log P(z_{t+\Delta t} = z), \tag{7.6}$$

where, using total and conditional probability,

$$P(z_{t+\Delta t} = z) = \sum_{\theta_i \in \Theta_d} P(z_{t+\Delta t} = z \mid x_{t+\Delta t}, \theta_i) b_t(\theta_i). \tag{7.7}$$

The second term in Equation (7.5) is

$$H(z_{t+\Delta t} \mid b_t) = \sum_{\theta_i \in \Theta_d} b_t(\theta_i) \sum_{z \in Z_d} P_{zx\theta} \log P_{zx\theta}, \tag{7.8}$$

where $P_{zx\theta} = P(z_{t+\Delta t} = z \mid x_{t+\Delta t}, \theta_i)$ for short.

When generating an information map, Equation (7.5) is evaluated for each $x \in$

X_d instead of $X_{t+\Delta t}$, the list of states that can be reached in the next timestep. Each term in Equation (7.5) is of order $O(|\Theta_d||Z_d|)$ so generating the entire map is $O(|X_d||\Theta_d||Z_d|)$. Each operation requires a call to the sensor model $P(z_t \mid x_t, \theta)$, which can be expensive. For example, a bearing sensing modality requires calls to relatively expensive trigonometric functions. However, these calls can be reduced with caching and memoization.

The main computational concern is that the numbers of sensor and target states are often exponential functions of some other variable. Consider a mobile sensor localizing a target in a square field. We might discretize to n states per dimension, meaning the sensor and target could each occupy any of n^2 states. Generating the information map is of order $O(n^4|Z|)$. Clearly, increasing the discretization or the size of the search area incurs enormous increases in computation.

7.3.2 Fisher Information

Fisher information offers another way to generate information maps. The Fisher information $\mathcal{I}(\alpha)$ describes the amount of information that a random variable carries about unknown parameter α.

In this work, the observable variable is the measurement z and it is conditional on the sensor state x and target state θ. Consider the bearing-only sensor model, where the measurement value is scalar and has Gaussian noise that is constant across the state space. The Fisher information matrix for a specific sensor-target state is

$$\mathcal{I}(x, \theta) = \frac{1}{\sigma^2} \nabla_\theta g(\theta, x) \nabla_\theta g(\theta, x)^\top, \tag{7.9}$$

where the σ is the standard deviation of the Gaussian noise and $\nabla_\theta g(\theta, x)$ is the gradient of the measurement function g with respect to θ. In bearing-only sensing, g is the true bearing in eq. (2.6) and its gradient is

$$\nabla_\theta g(\theta, x) = \frac{1}{\|\theta - x\|^2} \begin{bmatrix} \theta^{\mathrm{e}} - x^{\mathrm{e}} \\ x^{\mathrm{n}} - \theta^{\mathrm{n}} \end{bmatrix}. \tag{7.10}$$

When calculating Fisher information for a point in the sensor's state space, the sensor state is known but the target state is not. Therefore, the current belief is used to take an expectation over sensor states:

$$\Phi(x) = \sum_{\theta_i \in \Theta_d} b_t(\theta_i) \mathcal{I}(x, \theta_i). \tag{7.11}$$

An information map requires a scalar value of information at each point, so the determinant is commonly used [96]:

$$\phi(x) = \det \Phi(x). \tag{7.12}$$

The information map ϕ is built using Equations (7.9) to (7.12) to evaluate the information at each point $x \in X_d$.

The complexity of generating the Fisher information map is $O(|X_d||\Theta_d|)$, better than mutual information by the factor $|Z_d|$. This factor is eliminated in part because of the simplified version of Fisher information in Equation (7.9). In cases with more complex noise models, integration over the measurement space is needed to compute $\mathcal{I}(\theta, x)$. However, $\mathcal{I}(\theta, x)$ can be precomputed offline, so that the Fisher information map complexity is still $O(|X_d||\Theta_d|)$. Further, there are no calls to the log or measurement functions. The low complexity helps explain why Fisher information maps are common in prior work [94], [96], including a real-time implementation on a robot [109].

However, Fisher information is not the best metric for all problems, and picking the exact form of the information map is an open research question. The relationship between Fisher and mutual information is complex [110]. In some problems, it is not even clear how to apply Fisher information. For example, the double-Moxon sensor model defined in Equation (7.3) has just two discrete observations, and the gradient is not well defined. Mutual information might be more appropriate in that case.

7.4 Generating Maps and Coefficients with Neural Networks

A stationary target sits in a $200\,\mathrm{m} \times 200\,\mathrm{m}$ field. The belief is represented with an $n \times n$ discrete grid, where $n = 28$. The weight of each cell in the belief gives the probability the target is in the grid. The belief is initialized to a uniform distribution.

The agent moves through this field while searching for the target. When using the bearing modality, the agent state space is discretized to $n \times n$ points in the search area. When using the double-Moxon modality, the agent state space is $n \times n \times 36$, as agent headings are discretized into 36 points.

By using the sensor models and mutual or Fisher information, information maps over the agent state space can be generated. In the bearing modality, these maps cover $n \times n$ points; in the double-Moxon modality, they cover $n \times n \times 36$ points.

These information maps are also decomposed into Fourier coefficients. For the bearing modality, $K = 5$ is the highest order coefficient, in line with prior work [109]. In the double-Moxon modality, $K = 17$ is used, the smallest value that captured major features in observed information maps.

7.4.1 Neural Network Architectures

Figure 7.1 shows the neural network architectures used for the bearing-only sensing modality. Both networks take in the 28×28 belief. One network outputs the 28×28 information map and the other outputs the 6×6 Fourier coefficients. Recall that $K = 5$ and there are $(K + 1) \times (K + 1)$ coefficients for decompositions in $\mathbb{R}^2$, for a total of 36. The same networks are used for Fisher information maps and coefficients as well as mutual information maps and their coefficients.

All convolutional layers have rectified linear activations. In the information map architecture, the dense layer has a softmax activation function, which ensures that all outputs are between 0 and 1 and sum to 1. This normalization is convenient as information maps are often normalized so they are proper densities. Further, it allows the KL divergence to be used as the loss function during training. In the

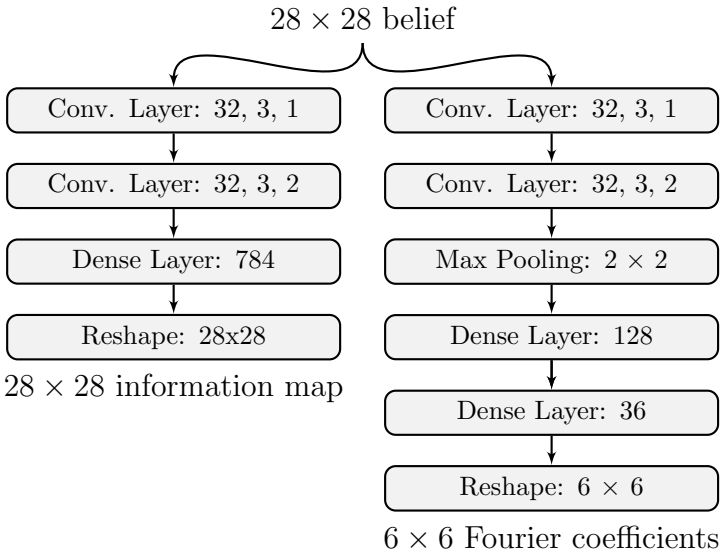

Figure 7.1: Neural network architectures for bearing-only sensing modality. The numbers listed for a convolutional layer are the number of filters, the width of each filter, and the stride size in each dimension.

Fourier coefficient architecture, the first dense layer has a rectified linear activation and the second has a linear activation. Mean absolute error serves as the loss function that compares network outputs to training data.

Figure 7.2 shows the neural network architectures used for the double-Moxon modality. Again, both networks take in the 28×28 belief. However, the seeker drone state is in SE(2), so there is an extra dimension. The output of one network is a $28 \times 28 \times 36$ information map, and the output of the other is $18 \times 18 \times 17$ complex Fourier coefficients. In SE(2), the Fourier coefficients are complex [111], so the coefficient network outputs two real numbers for each coefficient.

7.4.2 Training

The networks are trained on 500 simulations of 20 steps each. In each simulation, the target sits at a random location. The sensing agent selects its control input with a one-step, mutual information optimization. Measurements are made at each step, after which an information map is generated and decomposed into Fourier coefficients.

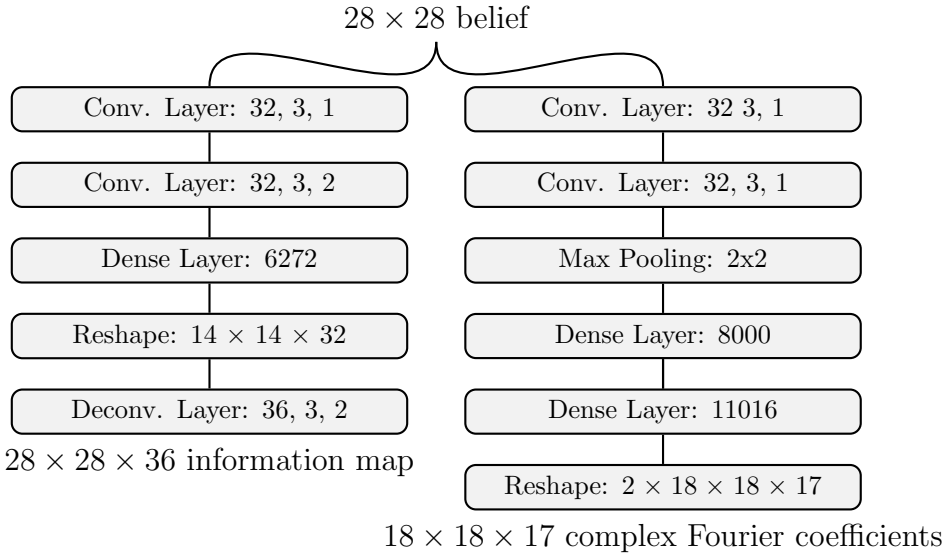

Figure 7.2: Neural network architectures for double-Moxon sensing modality. The numbers listed for a convolutional layer are the number of filters, the width of each filter, and the stride size in each dimension.

The beliefs are used as training inputs, and the resulting maps and coefficients are used as training outputs.

The networks were trained with Keras [112] with the Tensorflow [113] backend. Training was done on a Tesla k40c graphics processing unit (GPU). A GPU is not necessary, but it reduced training time from a few hours to about ten minutes.

Overfitting is always a concern with machine learning. To minimize overfitting, 10% of the training data is separated into a validation set. At each epoch, the loss is evaluated on both the training and validation sets. If the loss diverges, overfitting has likely occurred. This behavior was not observed, but methods such as dropout and regularization can be used if overfitting does occur [114].

7.4.3 Complexity in Evaluation

Before evaluating the trained networks in simulation, we consider the computational complexity of these evaluations. The networks are trained offline, so it does not matter if training is slow. However, a trained network must generate information maps from

beliefs in real-time. The computational complexity of evaluating a convolutional neural network for a new input is

$$O\left(\sum_{l=1}^{d} n_{l-1} \cdot s_l^2 \cdot n_l \cdot m_l^2\right), \tag{7.13}$$

where d is the number of convolutional layers, n_{l-1} is the number of input layers to layer l, s_l is the filter width of layer l, n_l is the number of filters in layer l, and m_l is the width of layer l's output [115]. The input is 2D so the number of input layers is $n_0 = 1$. In the worst case, the stride length is one, and the output is zero-padded so that the output width m_l equals the input width. The input is an $n \times n$ belief, so the output width of a layer is n.

Because convolutions take most of the computation time, this complexity does not include the cost of any pooling or fully connected layers. Prior empirical work suggests these layers account for an additional 5–10% of computation time [115].

If the network structure is held constant except for the input size $m_l = n$, then the asymptotic complexity is $O(m_l^2) = O(n^2)$. Recall that Fisher information was $O(|\Theta_d||X_d|)$; if n^2 points are used for Θ_d and n^2 for $X_d \subset \mathbb{R}^2$, the asymptotic complexity is $O(n^4)$. If $X_d \subset \mathrm{SE}(2)$ is discretized with n^3 points, then the complexity is $O(n^5)$. In theory, neural networks can generate information maps faster than computing them with Fisher or mutual information.

Of course, this result is theoretical and describes the limit as n grows. In reality, other network elements affect computation time. Further, convolutional layers often have nonlinear activation functions at their output, which can be expensive to compute. Finally, it is possible the network structure must implicitly grow with input width n. Perhaps more filters would be needed to capture fine-scale details that appear due to finer discretization of the state space.

7.5 Simulations

Once designed and trained, the networks are evaluated in simulations. After each observation, the belief is updated and information maps are generated along with their Fourier coefficients. These are compared to the neural network outputs. An example is shown in Figure 7.3.

Quantitative results in this section are from 100 20-step simulations with random target locations. As in the data generation, the agent moves according to a myopic entropy minimization. As a result, the beliefs seen in execution are similar to, but not necessarily equal to, those seen in training.

A tilde indicates a distribution was generated from Fourier coefficients, and the superscript n indicates the distribution was generated by a neural network. For example, ϕ is the true information map generated by the equations in Section 7.3; $\widetilde{\phi}$ is the distribution generated from the true Fourier coefficients—that is, coefficients generated from the true distribution. The distribution $\widetilde{\phi}^{\mathrm{n}}$ is generated from the network-produced coefficients and ϕ^{n} is the neural network approximation of the true information map.

7.5.1 Quality of Approximation

Because neural networks are nonlinear function *approximators*, there will be some degradation in the information maps produced. KL divergence is used to evaluate this degradation quantitatively. The KL divergence $D(P\|Q)$ is a measure of how well Q approximates P; the KL divergence is zero when Q equals P.

Table 7.1 shows the average KL divergence after each simulation step. The first quantity, $D(\phi\|\phi^{\mathrm{n}})$, compares the network-produced information maps to the true maps. The second quantity, $D(\widetilde{\phi}\|\widetilde{\phi}^{\mathrm{n}})$, captures the quality of the network-produced coefficients by comparing their reconstructed information map to that reconstructed from the true coefficients. The third quantity, $D(\phi\|\widetilde{\phi})$, compares the map generated from the true coefficients to the true information map. Fourier coefficients introduce band-limiting degradation but are still used to guide mobile sensors [94], [99], [100], [109], [111], so this last value is a useful reference of acceptable quality.

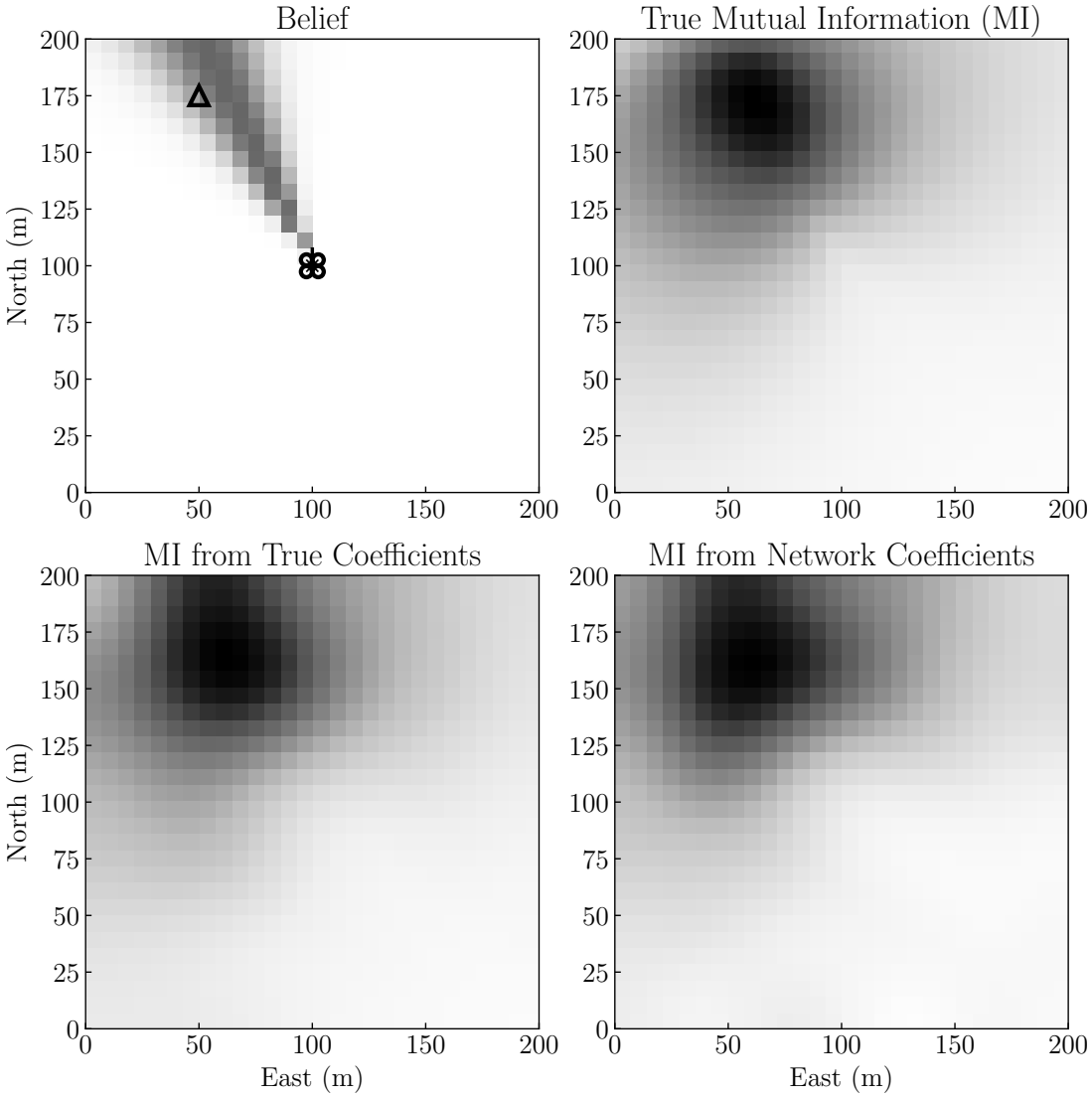

Figure 7.3: The mobile sensor (quadrotor) receives a bearing measurement to a target (triangle) and generates a belief. A mutual information map is then generated (upper right). A Fourier decomposition of this map is generated and the map is regenerated (bottom left). The Fourier coefficients generated by the neural network are also used to generate a map (bottom right).

Table 7.1: Measuring Network Map Quality with KL Divergence.

Modality	Metric	$D(\phi\|\phi^{\mathrm{n}})$	$D(\widetilde{\phi}\|\widetilde{\phi}^{\mathrm{n}})$	$D(\phi\|\widetilde{\phi})$
Bearing	Fisher	0.069	0.00045	2.78
	Mutual	0.036	0.0074	0.049
Double-Moxon	Mutual	0.038	0.010	0.10

Table 7.2: Computation Time for True and Neural Network (NN) Maps.

			Time to Compute (s)	
Modality	Metric	Method	Map	Coefficients
Bearing	Fisher	True	0.0061	0.0061
		NN	0.0031	0.0016
	Mutual	True	0.33	0.33
		NN	0.0021	0.0013
Double-Moxon	Mutual	True	0.76	1.33
		NN	0.0093	0.026

The results suggest the networks accurately capture the information maps. The divergence values between the true double-Moxon maps and the network maps are low. The divergence is only 0.038 when comparing the network map to the true map. In comparison, the divergence is nearly triple that when using the true coefficients to reconstruct the information map, suggesting that more information is lost when approximating with the true coefficients than with the neural network. If the true coefficients can be used in control tasks, then the network output will suffice as well. Figure 7.4 shows the approximations are also visually similar to the true maps.

7.5.2 Computation Time

Table 7.2 shows the mean time to generate maps and coefficients from beliefs. For the true methods, the map is made before decomposing it into coefficients, so the true coefficient time includes the true map generation time.

In the bearing modality, where the information map is a distribution over $\mathbb{R}^2$, the time to compute Fourier coefficients from the map is trivial. Both the domain

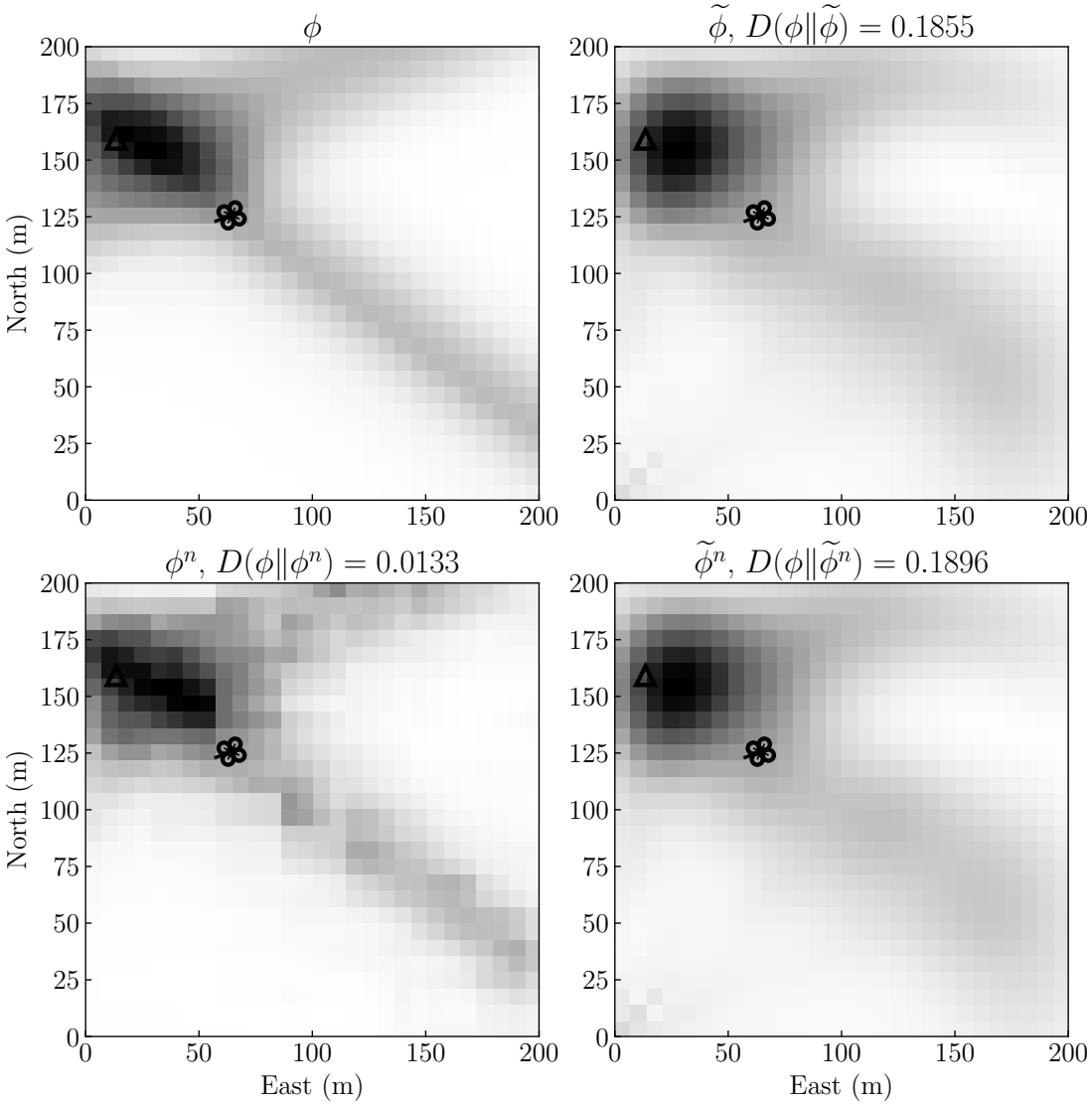

Figure 7.4: Comparison of true mutual information map and approximations during one timestep of double-Moxon simulation. The information map covers SE(2), but a 2D slice at 0° heading is shown here.

and number of coefficients are small, leading to fast computation. Fisher information is also computed rapidly, resulting in computation times comparable to the neural network. Although it was found above that a neural network could be faster than Fisher information in the asymptotic limit, there is not much difference at the map size used in this work. This result helps explain the use of Fisher information in real-time [109]. However, when using mutual information, neural networks generate maps and coefficients roughly two orders of magnitude faster.

In the double-Moxon modality, the speed improvement with neural networks is also about two orders of magnitude. The Fourier decomposition is slower because there is another dimension to integrate over, and more coefficients are needed to faithfully represent the distribution. When using mutual information, the time to generate information maps is almost a second, which is too slow for real-time use. In contrast, the neural network could comfortably be used at rates greater than 20 Hz, allowing real-time use.

Simulations were performed on a laptop computer with an i7 processor and 8 GB RAM. Neural network evaluations were performed on the CPU (instead of the GPU) for a fair comparison. Care was also taken to reduce the computation time of mutual information and its Fourier coefficients. Julia, a high-level language whose performance approaches C, was used. Caching and memoization were used to eliminate calls to measurement functions or the complex functions used in SE(2) Fourier decomposition. Vectors were ordered to match Julia's column-major ordering and prevent cache misses. Nonetheless, the neural network generated maps much more quickly.

7.6 Discussion

This chapter shows that convolutional neural networks can generate high-fidelity information maps in real-time, allowing mobile sensors to update maps as new observations are made. These maps are critical to ergodic control, and this improvement enables real-time recalculation of ergodic trajectories for some modalities, like the double-Moxon method, where Fisher information is not appropriate.

The neural network method for approximating information maps is easy to implement on real robots. The flight tests in the next chapter include this approximation technique, as the neural networks are implemented on the onboard computer (DJI Manifold) of the localizing drone. Because the Manifold is a full computer running the Ubuntu operating system, Tensorflow can be used to easily implement the neural network model. However, neural networks can be easily integrated onto less advanced systems, and they have been implemented on flight computers with extremely limited computational power [116].

One might be concerned with the simple dynamics used in this chapter and what effect more complicated dynamics might have on neural network performance. However, the neural networks only learn mappings from beliefs to information distributions, so it does not matter if the sensing agent's dynamic model is complicated. What *does* matter is the dimensionality of the target space. Because image analysis is the prime use of convolutional neural networks, most available infrastructure does not handle inputs with more than three dimensions. For example, Tensorflow currently allows convolutions over three dimensions or less. Fortunately, target spaces are often 2D or 3D, as they typically cover possible target locations.

Another possible concern is unmodeled sensor noise, which obliquely degrades neural network performance. On one hand, unmodeled sensor noise also affects true map generation techniques; maps generated using Fisher or mutual information will be inaccurate, so the neural network, which learns these mappings, will also be inaccurate. In this sense, neural networks are no worse at handling unmodeled noise. However, unmodeled sensor noise can specifically degrade neural network performance if the noise leads to unexpected observations, which lead to beliefs not seen in training. Likewise, if trajectories used during execution differ from those used in training, new beliefs might be seen which the network has not seen during training, leading to network-generated maps with lower fidelity. To mitigate the effect of unmodeled sensor noise, noise can be added during network training. To mitigate the effect of unseen trajectories, networks should be trained on trajectories to be used in execution.

Chapter 8

Evaluating Ergodic Control in Localization

The previous two chapters described how to generate ergodic trajectories for a mobile robot. The last chapter showed how to reduce the computational complexity of generating information maps, a critical step when applying ergodic control.

In this chapter, ergodic control is applied to the drone-based localization task. Simulations are run to compare its performance against simple, greedy localization algorithms. In addition, significant unmodeled noise is introduced for some of the simulations. This noise is added to test the claim that ergodic control performs well in environments with unmodeled noise.

8.1 Background

Section 6.2 introduced the Fourier-based ergodic metric used to generate trajectories. Once the trajectory and information distribution are decomposed into Fourier coefficients, these coefficients are compared. The trajectory is then modified until its coefficients match those of the information distribution.

There are different methods to generate trajectories with coeffients matching those of the information distribution. Spectral multiscale coverage (SMC) was one of the first methods proposed [99]. This method is a feedback law for single or double

integrators in $\mathbb{R}^2$ or $\mathbb{R}^3$. This feedback law greedily minimizes the ergodic metric. This greedy behavior is the reason "spectral multiscale" appears in the title. SMC guides an agent to travel to regions with highest information density first, as this most quickly reduces the ergodic metric. As time goes on, SMC then guides an agent to regions with lower density. In this way, SMC captures the most prominent spectral features first. SMC has mostly been applied to coverage problems.

Another method for designing ergodic trajectories is based on projection-based trajectory optimization (PTO) [100]. In this method, an initial trajectory is perturbed in the direction that minimizes the gradient of the ergodic metric. This iterative process continues until a local minimum in the ergodic metric is reached. One advantage of PTO over SMC is that it can handle nonlinear system dynamics. Another advantage is trajectory efficiency; because PTO does not greedily reduce the ergodic metric, it can plan a path that is ergodic over an entire horizon, and instead visit regions in an ordering that requires less control effort. In contrast, SMC always visits highest density regions first, even if they are far away.

PTO has been tested in localization tasks. In one of these tasks, a robotic fish localed a stationary electric target [96]. PTO performed similarly to greedy methods in the nominal environment but could outperform greedy methods in environments with significant unmodeled noise. In another work, a range sensor was controlled to localize a target in the presence of unmodeled distractor objects [97]. Similar conclusions were found regarding the performance of PTO and greedy methods; the presence of distractor objects led greedy methods to be "fooled" and fail. In contrast, PTO distributed the sensor's measurements over more of the state space, For this reason, it has been said that ergodic methods balance exploration and exploitation [94]; a mobile sensor will spend more time in regions with high information density, while still spending time in other regions. Another work used ergodic control to guide a drone localizing a target with bearing measurements [109]; although this work used another method to generate ergodic trajectories, it is related to the topic in this thesis. While this work showed the feasibility of ergodic control for this task, ergodic control was not compared against greedy methods.

This chapter tests PTO and SMC to generate ergodic trajectories for a simple

drone-based localization task. These trajectories are compared against the standard greedy method used throughout this work. It is assumed that the drone carries a sensor capable of making bearing measurements, instead of the double-Moxon sensor presented earlier in the work. A bearing sensor allows the robot state space to be in $\mathbb{R}^2$ instead of SE(2), allowing SMC to be used as well. Using multiple methods to generate ergodic trajectories allows us to understand whether performance is due solely to ergodicity or the method used. Additionally, it is easier to visualize the information distributions over $\mathbb{R}^2$ and analyze resulting trajectories. Because the double-Moxon returns measurements that are bearing-like, it is assumed that the results will hold for that modality.

8.2 Nominal Conditions

To test the localization performance of ergodic control, localization trials were simulated. In each simulation, a stationary radio source is at a random location in a 400 m by 400 m search area. The localizing drone moves with integrator dynamics and gets instantaneous bearing measurements at 1 Hz. The source is considered localized when 50% of the belief is concentrated in one of 625 16 m by 16 m cells.

Ergodic trajectories with horizons of 50 seconds were generated with the PTO and SMC methods. These trajectories were recalculated after executing parts of them, in a model-predictive fashion. In some policies, ergodic trajectories were recalculated after only executing 2% of the trajectory; in others, 100% of the 50-step ergodic trajectory is executed before recalculating. These ergodic trajectories were compared to the standard greedy method. Table 8.1 shows the results.

The PTO ergodic trajectories perform poorly. Not only do they take more time to localize the radio source, they travel much more distance. PTO does not allow control saturation [103], so the drone's maximum speed (5 m/s) is not enforced. To prevent PTO from excessively exceeding the speed limit, a quadratic penalty on control effort is applied. However, the average speed still exceeded the drone's maximum speed. Even with the relaxed speed constraint, PTO took more time to localize the source.

A surprising result is the poor performance of rapid replanning with PTO; the

Table 8.1: Evaluating localization performance of ergodic control with nominal noise. The percent of the trajectory executed before replanning is shown in parentheses.

Policy	Localization Time (s)	Distance Traveled (m)
Greedy	33.7	161.1
PTO (2%)	70.6	335.7
PTO (10%)	51.3	407.8
PTO (20%)	38.4	426.3
PTO (50%)	35.8	497.5
PTO (100%)	34.7	516.2
SMC (2%)	30.0	150.2
SMC (10%)	29.8	148.9
SMC (20%)	29.9	149.8
SMC (50%)	31.2	156.1
SMC (100%)	34.2	170.8

worst performance comes when the execution horizon is only 2% of the planning horizon. The result is surprising because replanning incorporates the information from measurements made since the last plan was made.

Visually examining PTO trajectories shows what the problem is. Figure 8.1 shows the ergodic trajectory planned after a single measurement. PTO does not greedily minimize the ergodic metric; it simply designs trajectories that have low overall scores. To PTO, two trajectories that have the same ergodic score are equivalent; it does not matter if one trajectory first visits the highly informative regions, as long as they are eventually visited. Because the most informative regions are closer to the radio source than thedrone, PTO is content to design trajectories that reach the informative regions by the end of its planning horizon. But if rapid replanning is performed, the drone never reaches that part of the trajectory.

In contrast to PTO, SMC produced ergodic trajectories that performed much better, even outperforming the greedy method. SMC also appears to be insensitive to changes in the execution horizon. SMC's robustness and good performance is likely due to the way it greedily reduces the ergodic metric. Because states near the radio source have the most information, the fastest way to make the drone's trajectory

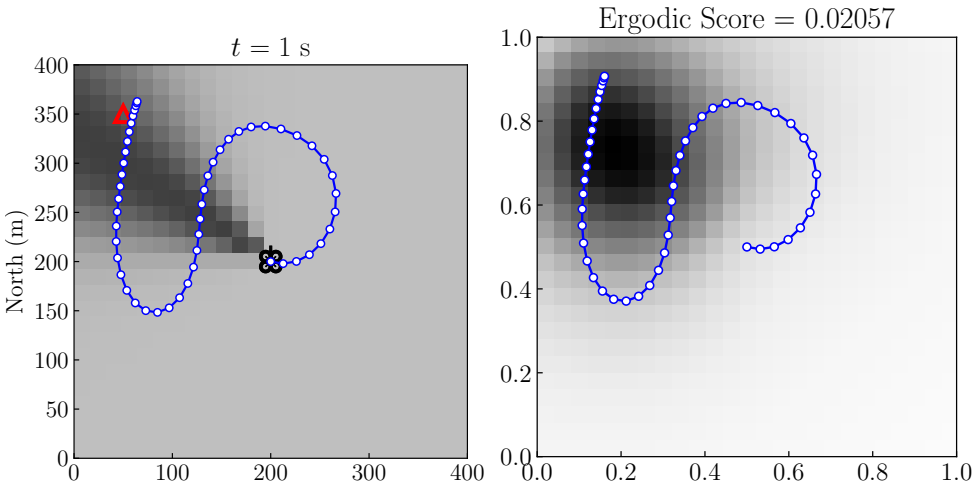

Figure 8.1: On the left, beliefs. On the right, the planned ergodic trajectories are plotted over information.

match the information distribution is to immediately spend time in these high information states. Therefore, the drone moves to these points first. Normally, SMC would then move the drone to other states with less information, to more accurately match its trajectory with the distribution. But by the time the drone has visited the highly informative states, good measurements have been obtained and the radio source is localized.

Figure 8.2 shows example trajectories from a single simulation. The SMC trajectory looks more like the greedy trajectory than the PTO trajectory. SMC's good performance seems to be unrelated to its being ergodic; its trajectories have higher ergodic scores than those produced by PTO. Instead, its good performance comes from the way in which it achieves ergodic trajectories. In this way, it is similar to the greedy method but less myopic. Whereas the greedy method picks the immediate action leading to more information gain over a single step, SMC plans a trajectory that leads the drone to the most informative regions in the state space. For this reason, SMC slightly outperforms this particular greedy method in this scenario.

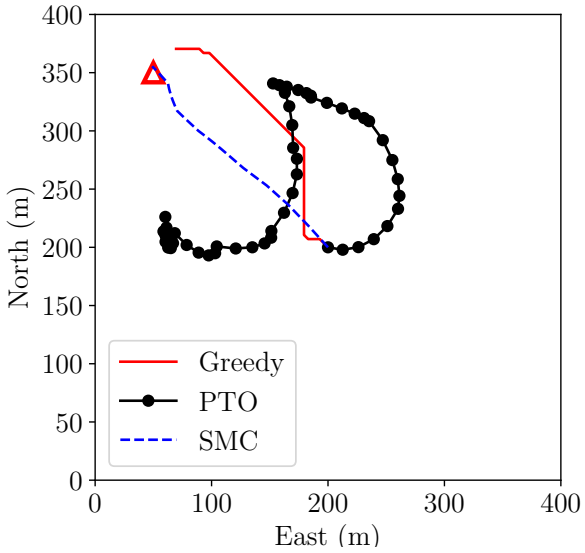

Figure 8.2: Example trajectories starting from (200, 200). The triangle is the target.

8.3 Unmodeled Noise

There are many sources of unmodeled noise affecting radio measurements. However, the modalities presented in Chapter 3 are designed to be robust to many of these noise sources. By comparing the strength measured by two antennas simultaneously, sources of noise affecting both antennas are canceled out.

However, multipath is a source of noise that is not accounted for. Multipath occurs when radio waves reflect off different surfaces and arrive at the receiving antenna. The receiving antenna not only sees the radio waves coming directly from the transmitter, but also these reflections. As a result, the receiving antenna sees radio waves coming from different paths, which is why this phenomenon is called multipath.

Multipath noise is particularly problematic for radio localization, including the modalities presented in Chapter 3. These modalities get a rough bearing estimate by comparing signal strengths measured by antennas pointing in different directions. If radio waves arrive from multiple directions, it might be difficult to determine the bearing to the transmitter. It is difficult to predict how strength measurements will be affected; radio waves suffer constructive and destructive interference, so one cannot

simply add the strengths of all waves at a point to estimate the power measured there.

Unfortunately, there is relatively little work involving multipath in mobile robot localization. Because radio modeling is so difficult in the first place, most work assumes operation in open environments, with no buildings to cause multipath. For example, that is the approach taken so far in this thesis. However, multipath might play a role when operating in urban environments or near airports.

In one work, multipath reflections from the ground and ceiling are considered for a robot operating indoors [117]. However, the contributions from these reflections are simply added to the line-of-sight path, which is not necessarily correct, as mentioned above. This model only slightly improves accuracy when predicting measured signal strength (see Figure 8 in that work). The large experimental deviations from their model might be the result of destructive interference. In another work, ray-tracing is used to find all multipath signals in a cluttered environment [118]. Again, these signals are treated individually rather than considering their cumulative effect.

In this chapter, a simplified multipath model is included as unmodeled noise. We make the following assumption: one of the incoming waves will dominate the others and the bearing measured by a sensing modality will be in the direction of the strongest wave. If the receiving antenna has a clear path to the receiver, the true bearing is estimated. But if the direct path is occluded, incoming direction of the shortest multipath reflection is estimated to be the bearing. Recall the equation defining bearing between the seeker drone position x_t and target location θ:

$$\beta_t(x_t, \theta) = \arctan\left(\frac{\theta^{\mathrm{e}} - x_t^{\mathrm{e}}}{\theta^{\mathrm{n}} - x_t^{\mathrm{n}}}\right). \tag{8.1}$$

We define θ^R as the starting point of the last straight-line segment in the shortest reflected path from the target to the seeker drone. When there is an occlusion preventing a line of sight path, the bearing to θ^R is used as the bearing measured by the the drone.

$$\beta_t^R(x_t, \theta) = \begin{cases} \beta_t(x_t, \theta), & \text{if line-of-sight between } \theta \text{ and } x_t \text{ is clear.} \\ \beta_t(x_t, \theta^R) & \text{otherwise.} \end{cases} \tag{8.2}$$

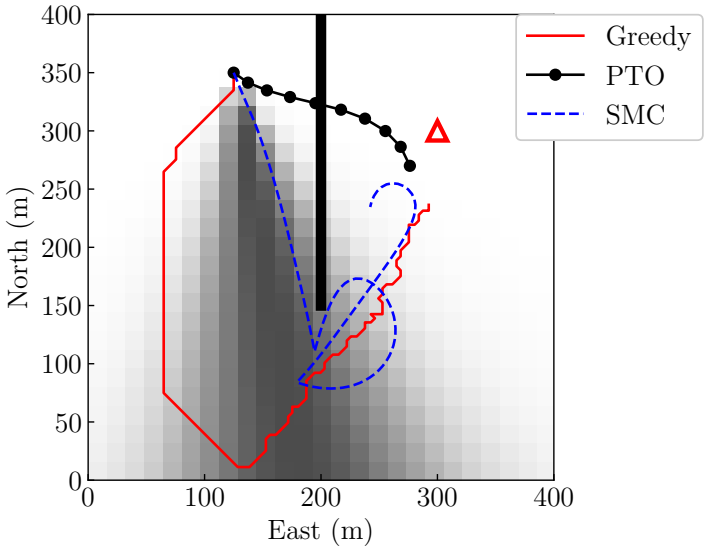

Figure 8.3: Localizing a target occluded by a wall. The belief shown is after a single step. The PTO trajectory flies over the wall and quickly localizes the radio source, while the other methods are fooled by the reflection.

Of course, the model proposed here is a gross simplification and no claims are made about its accuracy. The purpose of this section is simply to find a somewhat-plausible source of unmodeled noise and evaluate ergodic control under these conditions. It is left to future work to determine more realitic multipath models.

The scenario envisioned has the drone initially separated from the radio source by a large building or wall cutting through the middle of the search area. Additionally, radio signals reflect off the square boundary of the search area. Figure 8.3 shows resulting trajectories solved for this scenario. PTO performs well, moving the drone over the wall, where it makes good measurements and correctly localizes the target in 11 s. In contrast, SMC and the greedy method first move towards the reflecting wall, as this is the most informative region according to the model. Of course, this model is flawed and does not account for the dividing wall and reflections, so these methods could not converge to the proper target estimate even after 200 s.

8.4 Discussion

The results in this chapter largely confirm those found in prior work. Specifically, ergodic control performs roughly as well as greedy methods in nominal conditions, but it can perform especially well in environments with unmodeled noise. These results held in the kind of multipath a drone might face when localizing a radio target in a cluttred, urban environment.

Another interesting result is that the performance of ergodic control depends heavily on the method used to generate ergodic trajectories. There seems to be a fundamental tradeoff. A method that prioritizes the most informative regions of the state space (as SMC does) will perform better in nominal conditions but is more likely to fail in the face of significant unmodeled noise. A method like PTO does not prioritize more informative regions as long as the overall trajectory is ergodic. These methods will perform better in environments with unmodeled noise, but will perform less well in nominal conditions.

Chapter 9

Conclusion

9.1 Summary and Contributions

Because this thesis took a holistic approach to drone-based radio source localization, it considered both the hardware and algorithms needed. Hardware contributions resulted in improved sensing modalities. Algorithmic contributions centered around two main approaches: classic non-myopic belief space planning using the Markov decision process framework, and ergodic control for information gathering.

Hardware

1. **Presenting and evaluating two sensing modalities:** The directional-omni (Section 3.3) and double-Moxon (Section 3.4) modalities were introduced and evaluated. In Section 3.5.1 (specifically, Table 3.3 and Figure 3.15), these modalities were shown to greatly outperfom the widely used "rotate-for-bearing" modality [15], [22], [23]. While the double-Moxon and directional-omni modalities are less informative than methods that provide instantaneous bearing measurements, they are much simpler; complex methods like beam-steering have only rarely been implemented on drones [32]. Flight tests revealed the double-Moxon modality to be more robust than the directional-omni modality, allowing it to be used for more frequencies. Ultimately, the double-Moxon modality was

used to find a moving drone by its telemetry radio (Section 5.4).

2. **Simple, low-cost implementation:** Not only was the performance analyzed, but it was shown how to physically realize these modalities without much user effort or resources. A comparison of commercially available software defined radios (SDRs) was presented in Section 3.2.2. The double-Moxon modality can be implemented for under $50 USD and one hour of effort when using two of the cheaper SDRs and the antenna construction technique detailed in Section 3.4.1.

3. **Localization of novel radio sources:** The double-Moxon modality was used to localize a ham radio, a wildlife-radio tag, a cell phone, and the telemetry radio of a moving drone. Localizing a drone contributes to the nascent field of counter-UAS which will likely play a critical role in protecting national infrastructure.

Non-myopic belief-space planning

1. **Improving offline POMDP solvers for problems with belief-dependent rewards:** Because localization can be cast as a POMDP with belief-dependent rewards, this thesis investigated recent work extending POMDP solvers to use belief-dependent rewards. The key contribution of Chapter 4 is an improved lower bound that greatly sped up offline solvers that incorporate belief-dependent rewards. In Section 4.4, this lower bound was shown to reduce computation by a factor of about 20 on simple toy problems. In Section 4.5, an offline solver incorporating this bound was evaluated in a simplified drone-based localization task. But even after solving for 12 hours, the resulting policies offered meager improvements over simple methods like greedy solvers. While offline solvers did not prove fruitful in this work, the contributions in Chapter 4 can be built upon by future researchers.

2. **Analyzing online methods for drone-based radio source localization:** In Chapter 5, Monte Carlo tree search (MCTS) was used to guide a drone using the double-Moxon modality to localize a moving drone by its telemetry radio. A particle filter was used for estimating the state of the target drone. It was

found that many particles were needed to maintain good estimates, but that MCTS, which requires propagating the particle sets, was too slow with such large particle sets. The resulting compromise maintained a large particle set for localization purposes, while a downsampled particle set was fed to MCTS. Simulations showed that this compromise outperformed a greedy solution, which was the prior state of the art for drone-based radio source localization. Downsampling made the computation feasible on a real drone, and a flight test was shown in Section 5.4. There does not appear to be other research in which a drone has localized another by its telemetry emissions.

Ergodic Control

1. **Analysis of optimality:** Chapter 6 analyzed the optimality of ergodic control for information gathering problems (localization is effectively an information gathering problem). By analyzing an ergodic metric based on Fourier decomposition, a problem class was found under which ergodic trajectories optimally gather information. This result was validated empirically. Unfortunately, the class is very limited, and drone-based radio source localization is outside it. Combined with evaluations in Chapter 8, it seems likely that ergodic control is suboptimal for drone-based radio source localization. However, I do not know of any other attempts at analyzing the optimality of ergodic control, and I hope others will build upon this contribution to further our understanding of when ergodic control should be used.

2. **Neural networks generating information maps:** In ergodic control for information gathering, trajectories are generated such that they are ergodic with respect to an information map, or a distribution of information over the state space. This information map is generated from the sensor model and the belief over target states. Unfortunately, generating this map can be slow. The key contribution of Chapter 7 is introducing and evaluating a neural network architecture that can generate information maps from beliefs. The resulting information maps are high-fidelity approximations that are produced orders of

magnitude faster.

3. **Evaluation in drone-based radio source localziation:** Finally, ergodic control was evaluated in drone-based radio source localization. The simulations in Chapter 8 suggest that, in nominal conditions, ergodic control underperforms greedy methods. However, it was shown that under a simple multipath model, ergodic control could outperform greedy methods.

9.2 Further Work

This thesis is not the last word on localizing radio sources with an autonomous drone. There are various areas for improvement by future researchers.

9.2.1 Improved Planning

The belief-space planning literature is vast, and other forms of belief-space planning should be considered. A promising option is sequential action control (SAC) [105]. Ironically, this method was first applied to generating ergodic trajectories. But recent work directly applies SAC to belief-space planning for active sensing, with promising results [119]. Future work should test this and other algorithms on drone-based radio localization and tracking.

Neural networks offer another promising avenue for future planners. Their motivating advantage is that they can be trained offline with efficient online performance. Improving online performance, so that algorithms can be run in real-time on real platforms, has been a major theme of this work. While this work has used neural networks to generate information maps for an ergodic planner, neural networks could theoretically be used to directly map target beliefs to seeker actions. These mappings could learn to imitate the actions produced by some MDP solver, like MCTS, in simulations run offline [116]. Alternatively, deep reinforcement learning can be used to estimate the values of each action directly [120]. Because these simulations are run offline, increasingly complex noise and dynamic models can be used without affecting

online evaluations. Therefore it might be easier to incorporate more realistic and non-deterministic motion models for the seeker drone.

9.2.2 Miniaturization

The sensing modalities proposed and analyzed in this work are simple and lightweight, but some applications, especially those with radios at lower frequencies, will need smaller options. Recall that antenna sizes scale with wavelength, which scales inversely with frequency. The lowest frequency used in this work, 217 MHz, required Moxon antennas half a meter wide. While these antennas are not heavy, their width approaches a limit after which it is unwieldy to mount on small consumer drones. An antenna for 100 MHz would be about twice as wide, which is likely infeasible for many drones. Minitaturization would allow a wider array of radio sources to be localized, and might allow for nano quadcopters like the Crazyflie to be used [121].

One option is to use the time difference of arrival (TDOA) of a radio signal arriving at different antenna elements. While this work stayed away from TDOA methods because of their complexity, certain applications might require it. Further, hardware complexities might be reduced with recent advances in commodity SDRs. For example, the HackRF One radios have a clock input and output port; theoretically, connecting the output of one radio to the input of another would allow for precise timing of incoming signals [122]. The same company that made the RTL-SDR radios has announced a future product called the KerberosSDR, which will have four antenna inputs and is being marketed for direction finding applications [123]. Future work should investigate whether these advances and TDOA can result in smaller sensor systems.

9.2.3 Multiple Radio Sources

While this work has examined localization of different types of radio sources, it assumes that only one radio source is active during localization. Localizing multiple radio sources at once is a challenging hardware problem and it is not clear that the sensing modalities presented herein would be able to separate emissions from multiple

radios. This problem is not unique to these modalities but seems to be a problem for other modalities as well [24]. While some researchers claim they can find multiple radio sources, they often rely on using wildlife radio-tags that operate at slightly different frequencies, making it easy to separate the transmissions for each one [18].

More advanced radio frequency analysis might help in certain applications. For example, drone telemetry radios often have different frequency hopping patterns so they can operate concurrently. Therefore, the radios are likely to be radiating at different frequencies at any given moment. By looking at a wide spectrum, as done in this work with the HackRF One radios, emissions from multiple radios can be captured and analyzed separately. Of course, this leads to a data association problem for subsequent measurements—which emissions correspond to which target? One option is to learn the hopping pattern of each target radio [124], but it is not clear how to do so once all targets start transmitting. Specialized filters such as the probability hypothesis density (PHD) filter and its derivatives can also help with the data association problem [125].

However, if multiple radio sources operate in a narrow frequency range, their emissions will be difficult to separate. Incoming radio waves will suffer constructive and destructive interference, so signal strengths will not simply add. A simple algorithmic option might be to isolate each target radio in sequence. If one target radio is much closer than the others, gains on the seeker drone's radios could be tuned down until only emissions from the closest radio are captured. This "divide and conquer" approach is likely suboptimal and might not even be feasible. Clearly, the problem of hunting multiple radio sources at the same frequency requires further investigation.

Bibliography

[1] M. Geyer and R. Frazier, "FAA GPS RFI mitigation program," in *International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS)*, 1999 (cit. on p. 2).

[2] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*. Ganga-Jamuna Press, 2006 (cit. on p. 2).

[3] S. Pullen, G. Gao, C. Tedeschi, and J. Warburton, "The impact of uninformed RF interference on GBAS and potential mitigations," in *Proceedings of the 2012 International Technical Meeting of the Institute of Navigation (ION ITM 2012), Newport Beach, CA*, 2012 (cit. on p. 3).

[4] GPS World, *The hunt for RFI: Unjamming a coast harbor*, Accessed: 2018-06-14, Jan. 2003. [Online]. Available: `http://gpsworld/the-hunt-rfi` (visited on 09/30/2010) (cit. on p. 3).

[5] The Guardian, *Car thieves using GPS 'jammers'*, Accessed: 2018-06-15. [Online]. Available: `https://www.theguardian.com/technology/2010/feb/22/car-thieves-using-gps-jammers` (cit. on p. 3).

[6] Septentrio, *The chirp jammer: A GPS hit and run*, Accessed: 2018-08-14, Aug. 2016. [Online]. Available: `https://www.septentrio.com/insights/chirp-jammer-gps-hit-and-run` (visited on 08/14/2018) (cit. on p. 3).

[7] Septentrio, *GNSS jamming and road tolling*, Accessed: 2018-08-16, Aug. 2018. [Online]. Available: `https://www.septentrio.com/insights/gnss-jamming-and-road-tolling` (visited on 08/16/2018) (cit. on p. 3).

[8] J. Grabowski, "Field observations of personal privacy devices," in *Proceedings of the 2012 International Technical Meeting of The Institute of Navigation*, 2001 (cit. on p. 3).

[9] T. Kraus, R. Bauernfeind, and B. Eissfeller, "Survey of in-car jammers-analysis and modeling of the RF signals and IF samples (suitable for active signal cancelation)," *Proceedings of ION GNSS 2011*, pp. 20–23, 2011 (cit. on p. 3).

[10] R. H. Mitch, R. C. Dougherty, M. L. Psiaki, S. P. Powell, B. W. OHanlon, J. A. Bhatti, and T. E. Humphreys, "Signal characteristics of civil GPS jammers," in *Proceedings of ION GNSS*, 2011 (cit. on p. 3).

[11] Inside GNSS, *FCC fines operator of GPS jammer that affected Newark airport GBAS*, Accessed: 2016-01-19, Aug. 2013. [Online]. Available: `http://insidegnss.com/node/3676` (visited on 09/30/2010) (cit. on pp. 3, 14).

[12] United States Department of Transportation, *UAS sightings report*, Accessed: 2018-05-08. [Online]. Available: `https://www.faa.gov/uas/resources/uas_sightings_report/` (cit. on p. 3).

[13] UK Airprox Board, *Current drone Airprox count and information*, Accessed: 2018-05-10. [Online]. Available: `https://www.airproxboard.org.uk/Topical-issues-and-themes/Drones/` (cit. on p. 3).

[14] FAA, *Wildfires and drones don't mix*, Accessed: 2018-08-21, 2017. [Online]. Available: `https://www.faa.gov/uas/where_to_fly/airspace_restrictions/wildfires/` (visited on 08/21/2018) (cit. on p. 3).

[15] O. M. Cliff, R. Fitch, S. Sukkarieh, D. L. Saunders, and R. Heinsohn, "Online localization of radio-tagged wildlife with an autonomous aerial robot system," in *Robotics: Science and Systems*, 2015 (cit. on pp. 4–6, 13, 16, 23, 33, 44, 71, 128).

[16] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Distributed cooperative search using information-theoretic costs for particle filters, with quadrotor applications," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2006 (cit. on pp. 4, 5, 18, 19, 53, 71, 105).

[17] P. J. Seddon and R. Maloney, *Tracking Wildlife Radio-Tag Signals by Light Fixed-Wing Aircraft*. Department of Conservation Wellington, New Zealand, 2004 (cit. on p. 4).

[18] F. Körner, R. Speck, A. H. Göktogan, and S. Sukkarieh, "Autonomous airborne wildlife tracking using radio signal strength," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2010 (cit. on pp. 4, 5, 22, 133).

[19] E. M. Geyer, B. M. Wirier, and R. A. Frazier, "Airborne GPS RFI localization algorithms," in *International Technical Meeting of the Satellite Division of The Institute of Navigation*, 1997 (cit. on p. 4).

[20] U.S. Marines (@USMC), *Marine rifle squads are becoming more lethal, agile and adaptable.* Accessed: 2018-06-15. [Online]. Available: `https://twitter.com/USMC/status/995997159186149378/` (cit. on p. 4).

[21] G. M. Hoffmann, "Autonomy for sensor-rich vehicles: Interaction between sensing and control actions," PhD thesis, Stanford University, 2008 (cit. on p. 5).

[22] A. Perkins, L. Dressel, S. Lo, and P. Enge, "Antenna characterization for UAV based GPS jammer localization," in *Institute of Navigation (ION) GNSS+*, 2015 (cit. on pp. 5, 11, 13, 16, 23, 33, 43, 128).

[23] A. Perkins, L. Dressel, S. Lo, and P. Enge, "Demonstration of UAV-based GPS jammer localization during a live interference exercise," in *Institute of Navigation (ION) GNSS+*, 2016 (cit. on pp. 5, 13, 14, 16, 23, 128).

[24] L. Dressel and M. J. Kochenderfer, "Efficient and low-cost localization of radio signals with a multirotor UAV," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2018 (cit. on pp. 5, 71, 105, 133).

[25] L. Dressel and M. J. Kochenderfer, "Pseudo-bearing measurements for improved localization of radio sources with multirotor UAVs," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018 (cit. on pp. 5, 71).

[26] A. Posch and S. Sukkarieh, "UAV based search for a radio tagged animal using particle filters," in *Australasian Conference on Robotics and Automation (ACRA)*, 2009 (cit. on pp. 5, 22).

[27] P. Soriano, F. Caballero, and A. Ollero, "RF-based particle filter localization for wildlife tracking by using an UAV," in *International Symposium of Robotics*, Barcelona, España, 2009 (cit. on pp. 5, 22).

[28] S. Venkateswaran, J. T. Isaacs, K. Fregene, R. Ratmansky, B. M. Sadler, J. P. Hespanha, and U. Madhow, "RF source-seeking by a micro aerial vehicle using rotation-based angle of arrival estimates," in *American Control Conference (ACC)*, 2013 (cit. on pp. 5, 23).

[29] J. T. Isaacs, F. Quitin, L. R. G. Carrillo, U. Madhow, and J. P. Hespanha, "Quadrotor control for RF source localization and tracking," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014 (cit. on pp. 5, 23).

[30] K. VonEhr, S. Hilaski, B. E. Dunne, and J. Ward, "Software defined radio for direction-finding in UAV wildlife tracking," in *IEEE International Conference on Electro Information Technology (EIT)*, 2016 (cit. on pp. 6, 23, 44).

[31] H. Bayram, N. Stefas, K. S. Engin, and V. Isler, "Tracking wildlife with multiple UAVs: System design, safety and field experiments," in *Multi-Robot and Multi-Agent Systems (MRS), 2017 International Symposium on*, 2017 (cit. on pp. 6, 23).

[32] A. Perkins, S. Lo, and P. Enge, "Development of a three-element beam steering antenna for bearing determination onboard a UAV capable of GNSS RFI localization," in *Institute of Navigation (ION) GNSS+*, 2017 (cit. on pp. 6, 24, 128).

[33] H. Bayram, K. Doddapaneni, N. Stefas, and V. Isler, "Active localization of VHF collared animals with aerial robots," in *IEEE Conference on Automation Science and Engineering (CASE)*, 2016 (cit. on pp. 11, 13, 23).

[34] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: An open-source robot operating system," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009 (cit. on p. 11).

[35] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960 (cit. on p. 16).

[36] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005 (cit. on pp. 16, 52, 53, 105).

[37] M. J. Kochenderfer, *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015 (cit. on p. 16).

[38] J. A. Thomas and T. M. Cover, *Elements of Information Theory*. John Wiley & Sons, 2006 (cit. on p. 19).

[39] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2000 (cit. on pp. 21, 22).

[40] T. R. Consi, J. R. Patzer, B. Moe, S. A. Bingham, and K. Rockey, "An unmanned aerial vehicle for localization of radio-tagged sturgeon: Design and first test results," in *OCEANS'15 MTS/IEEE Washington*, 2015 (cit. on p. 22).

[41] J. Hightower, R. Want, and G. Borriello, "SpotON: An indoor 3D location sensing technology based on RF signal strength," 2000 (cit. on p. 22).

[42] J. Graefenstein and M. E. Bouzouraa, "Robust method for outdoor localization of a mobile robot using received signal strength in low power wireless networks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008 (cit. on p. 22).

[43] D. Song, J. Yi, and Z. Goodwin, "Localization of unknown networked radio sources using a mobile robot with a directional antenna," in *American Control Conference (ACC)*, 2007 (cit. on p. 22).

[44] D. Song, C.-Y. Kim, and J. Yi, "Simultaneous localization of multiple unknown and transient radio sources using a mobile robot," *IEEE Transactions on Robotics*, vol. 28, no. 3, p. 668, 2012 (cit. on p. 22).

[45] P. Scerri, R. Glinton, S. Owens, D. Scerri, and K. Sycara, "Geolocation of RF emitters by many UAVs," in *AIAA Infotech@Aerospace Conference*, 2007 (cit. on p. 22).

[46] M. Hasanzade, Ö. Herekoğlu, R. Yeniçeri, E. Koyuncu, and G. İnalhan, "RF source localization using unmanned aerial vehicle with particle filter," in *IEEE International Conference on Mechanical and Aerospace Engineering (ICMAE)*, 2018 (cit. on p. 22).

[47] J. Derenick, J. Fink, and V. Kumar, "Localization using ambiguous bearings from radio signal strength," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011 (cit. on p. 22).

[48] K. Dantu, P. Goyal, and G. S. Sukhatme, "Relative bearing estimation from commodity radios," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2009 (cit. on p. 23).

[49] C. J. Lowrance and A. P. Lauf, "Direction of arrival estimation for robots using radio signal strength and mobility," in *IEEE Workshop on Positioning, Navigation, and Communcations (WPNC)*, 2016 (cit. on p. 23).

[50] J. Graefenstein, A. Albert, P. Biber, and A. Schilling, "Wireless node localization based on RSSI using a rotating antenna on a mobile robot," in *IEEE Workshop on Positioning, Navigation, and Communcations (WPNC)*, 2009 (cit. on p. 23).

[51] B. N. Hood and P. Barooah, "Estimating DoA from radio-frequency RSSI measurements using an actuated reflector," *IEEE Sensors Journal*, vol. 11, no. 2, pp. 413–417, 2011 (cit. on pp. 23, 24).

[52] M. A. Ibrahim, "Signal source searching and tracking via antenna array beam steering on fixed wing UAVs for communication link enhancement," Master's

thesis, King Fahd University of Petroleum and Minerals (Saudi Arabia), 2012 (cit. on p. 24).

[53] N. Malhotra, M. Krasniewski, C Yang, S. Bagchi, and W. Chappell, "Location estimation in ad hoc networks with directional antennas," in *IEEE International Conference on Distributed Computing Systems (ICDCS)*, 2005 (cit. on p. 24).

[54] N. Deshpande, E. Grant, M. Draelos, and T. C. Henderson, "Received signal strength based bearing-only robot navigation in a sensor network field," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014 (cit. on p. 24).

[55] B. R. Huber, "Radio determination on mini-UAV platforms: Tracking and locating radio transmitters," Master's thesis, 2009 (cit. on p. 24).

[56] M. A. Magers, "Geolocation of RF emitters using a low-cost UAV-based approach," Master's thesis, Air Force Institute of Techonology (United States), 2016 (cit. on p. 24).

[57] A. Nika, Z. Zhang, X. Zhou, B. Y. Zhao, and H. Zheng, "Towards commoditized real-time spectrum monitoring," in *ACM Workshop on Hot Topics in Wireless*, 2014 (cit. on p. 28).

[58] T. Zhang, A. Patro, N. Leng, and S. Banerjee, "A wireless spectrum analyzer in your pocket," in *International Workshop on Mobile Computing Systems and Applications*, 2015 (cit. on p. 28).

[59] Y. Chen, Z. Liu, X. Fu, B. Liu, and W. Zhao, "Theory underlying measurement of AOA with a rotating directional antenna," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2013 (cit. on p. 29).

[60] L. A. Moxon, *HF Antennas for all Locations*, 2nd. Radio Society of Great Britian, 1993 (cit. on p. 35).

[61] AB1JX, *Moxon rectangle calculator*, Accessed: 2017-01-28. [Online]. Available: `http://ab1jx.1apps.com/ham/calcs/moxon/` (cit. on p. 36).

[62] G. J. Burke, A. Poggio, J. Logan, and J. Rockway, "Numerical electromagnetic code (NEC)," in *IEEE International Symposium on Electromagnetic Compatibility*, 1979 (cit. on p. 36).

[63] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1, pp. 99–134, 1998 (cit. on p. 49).

[64] C. H. Papadimitriou and J. N. Tsitsiklis, "The complexity of Markov decision processes," *Mathematics of Operations Research*, vol. 12, no. 3, pp. 441–450, 1987 (cit. on p. 49).

[65] M. Araya, O. Buffet, V. Thomas, and F. Charpillet, "A POMDP extension with belief-dependent rewards," in *Advances in Neural Information Processing Systems (NIPS)*, 2010 (cit. on pp. 50, 53, 71).

[66] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces.," in *Robotics: Science and Systems*, 2008 (cit. on pp. 50, 52).

[67] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973 (cit. on p. 51).

[68] J. Pineau, G. Gordon, S. Thrun, et al., "Point-based value iteration: An anytime algorithm for POMDPs," in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2003 (cit. on p. 52).

[69] D. Hsu, W. S. Lee, and N. Rong, "A point-based POMDP planner for target tracking," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008 (cit. on pp. 52, 71).

[70] L. Dressel and M. J. Kochenderfer, "Signal source localization using partially observable Markov decision processes," in *AIAA Infotech@Aerospace Conference*, Kissimmee, FL, 2015 (cit. on pp. 52, 53).

[71] N. Roy, W. Burgard, D. Fox, and S. Thrun, "Coastal navigation-mobile robot navigation with uncertainty in dynamic environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1999 (cit. on p. 52).

[72] N. Roy, G. J. Gordon, and S. Thrun, "Finding approximate POMDP solutions through belief compression," *Journal of Artificial Intelligence Research*, vol. 23, pp. 1–40, 2005 (cit. on p. 52).

[73] M. T. Spaan, T. S. Veiga, and P. U. Lima, "Decision-theoretic planning under uncertainty with information rewards for active cooperative perception," *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pp. 1–29, 2014 (cit. on pp. 53–55).

[74] Y. Satsangi, S. Whiteson, and M. T. J. Spaan, "An analysis of piecewise-linear and convex value functions for active perception POMDPs," Informatics Institute, University of Amsterdam, Tech. Rep. IAS-UVA-15-01, 2015 (cit. on pp. 53, 55, 56).

[75] A. Eck and L.-K. Soh, "Evaluating POMDP rewards for active perception," in *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2012 (cit. on pp. 54, 55).

[76] M. Araya, "Des algorithmes presque optimaux pour les problèmes de décision séquentielle à des fins de collecte d'information," PhD thesis, Université de Lorraine, 2013 (cit. on pp. 54, 63).

[77] G. Shani, J. Pineau, and R. Kaplow, "A survey of point-based POMDP solvers," *Autonomous Agents and Multi-Agent Systems*, vol. 27, no. 1, pp. 1–51, 2013 (cit. on p. 57).

[78] M. Hauskrecht, "Value-function approximations for partially observable Markov decision processes," *Journal of Artificial Intelligence Research*, pp. 33–94, 2000 (cit. on p. 57).

[79] M. Hauskrecht, "Incremental methods for computing bounds in partially observable Markov decision processes," in *AAAI Conference on Artificial Intelligence (AAAI)*, 1997 (cit. on p. 59).

[80] T. Smith and R. Simmons, "Heuristic search value iteration for POMDPs," in *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2004 (cit. on p. 62).

[81] M. T. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of artificial intelligence research*, pp. 195–220, 2005 (cit. on p. 63).

[82] M. Lauri, N. Atanasov, G. J. Pappas, and R. Ritala, "Myopic policy bounds for information acquisition POMDPs," *arXiv*, 2016. eprint: `arXiv:1601.07279` (cit. on p. 68).

[83] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in Neural Information Processing Systems (NIPS)*, 2010 (cit. on p. 69).

[84] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "DESPOT: Online POMDP planning with regularization," in *Advances in Neural Information Processing Systems (NIPS)*, 2013 (cit. on p. 69).

[85] Z. N. Sunberg and M. J. Kochenderfer, "Online algorithms for POMDPs with continuous state, action, and observation spaces," in *International Conference on Automated Planning and Scheduling (ICAPS)*, Delft, 2018. [Online]. Available: `https://arxiv.org/abs/1709.06196` (cit. on pp. 69, 73).

[86] L. Dressel and M. J. Kochenderfer, "Efficient decision-theoretic target localization," in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2017 (cit. on p. 71).

[87] L. Kocsis and C. Szepesvári, "Bandit based Monte-Carlo planning," in *European Conference on Machine Learning (ECML)*, 2006 (cit. on p. 72).

[88] D. P. Bertsekas, *Dynamic programming and optimal control*, 3. Athena Scientific Belmont, MA, 2005, vol. 1 (cit. on p. 75).

[89] R. J. Wallace and J. M. Loffi, "Examining unmanned aerial system threats & defenses: A conceptual analysis," *International Journal of Aviation, Aeronautics, and Aerospace*, vol. 2, no. 4, p. 1, 2015 (cit. on p. 78).

[90] S. R. Ganti and Y. Kim, "Implementation of detection and tracking mechanism for small UAS," 2016 (cit. on p. 80).

[91] P. Nguyen, M. Ravindranatha, A. Nguyen, R. Han, and T. Vu, "Investigating cost-effective RF-based detection of drones," in *Workshop on Micro Aerial Vehicle Networks, Systems, and Applications for Civilian Use*, 2016 (cit. on p. 80).

[92] X. Shi, C. Yang, W. Xie, C. Liang, Z. Shi, and J. Chen, "Anti-drone system with multiple surveillance technologies: Architecture, implementation, and challenges," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 68–74, 2018 (cit. on p. 80).

[93] K. Petersen, *Ergodic Theory*. Cambridge University Press, 1983 (cit. on p. 82).

[94] L. Miller, "Optimal ergodic control for active search and information acquisition," PhD thesis, Northwestern University, 2015 (cit. on pp. 82, 85, 93, 101, 108, 113, 120).

[95] Y. Silverman, L. M. Miller, M. A. MacIver, and T. D. Murphey, "Optimal planning for information acquisition," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013 (cit. on pp. 82, 83, 85).

[96] L. M. Miller, Y. Silverman, M. A. MacIver, and T. D. Murphey, "Ergodic exploration of distributed information," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 36–52, Feb. 2016 (cit. on pp. 82, 85, 86, 108, 120).

[97] L. M. Miller and T. D. Murphey, "Optimal planning for target localization and coverage using range sensing," in *IEEE Conference on Automation Science and Engineering (CASE)*, 2015 (cit. on pp. 82, 120).

[98] E. Ayvali, H. Salman, and H. Choset, "Ergodic coverage in constrained environments using stochastic trajectory optimization," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017 (cit. on pp. 84, 93, 103).

[99] G. Mathew and I. Mezić, "Metrics for ergodicity and design of ergodic dynamics for multi-agent systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 4, pp. 432–442, 2011 (cit. on pp. 84, 85, 101, 113, 119).

[100] L. M. Miller and T. D. Murphey, "Trajectory optimization for continuous ergodic exploration," in *American Control Conference (ACC)*, 2013 (cit. on pp. 85, 94, 113, 120).

[101] G. A. Hollinger and G. S. Sukhatme, "Sampling-based robotic information gathering algorithms," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1271–1287, 2014 (cit. on pp. 86, 97).

[102] S. Ivić, B. Crnković, and I. Mezić, "Ergodicity-based cooperative multiagent area coverage via a potential field," *IEEE Transactions on Cybernetics*, vol. 47, no. 8, pp. 1983–1993, 2017 (cit. on pp. 92, 103).

[103] A. Prabhakar, A. Mavrommati, J. Schultz, and T. D. Murphey, "Autonomous visual rendering using physical motion," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2016 (cit. on pp. 101, 121).

[104] A. Prabhakar, K. Fla, T. D. Murphey, et al., "Symplectic integration for optimal ergodic control," in *IEEE Conference on Decision and Control (CDC)*, 2015 (cit. on p. 103).

[105] A. R. Ansari and T. D. Murphey, "Sequential action control: Closed-form optimal control for nonlinear and nonsmooth systems," vol. 32, no. 5, pp. 1196–1214, 2016 (cit. on pp. 103, 131).

[106] R. E. Allen, A. A. Clark, J. A. Starek, and M. Pavone, "A machine learning approach for real-time reachability analysis," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014 (cit. on p. 103).

[107] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems (NIPS)*, 2012 (cit. on p. 103).

[108] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016 (cit. on p. 103).

[109] A. Mavrommati, E. Tzorakoleftherakis, I. Abraham, and T. D. Murphey, "Real-time area coverage and target localization using receding-horizon ergodic exploration," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 62–80, 2018 (cit. on pp. 108, 109, 113, 117, 120).

[110] X.-X. Wei and A. A. Stocker, "Mutual information, Fisher information, and efficient coding," *Neural Computation*, vol. 28, no. 2, pp. 305–326, 2016 (cit. on p. 108).

[111] L. M. Miller and T. D. Murphey, "Trajectory optimization for continuous ergodic exploration on the motion group SE(2)," in *IEEE Conference on Decision and Control (CDC)*, 2013 (cit. on pp. 110, 113).

[112] F. Chollet, *Keras*, `https://github.com/fchollet/keras`, 2015 (cit. on p. 111).

[113] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., "Tensorflow: A system for large-scale machine learning.," in *OSDI*, 2016 (cit. on p. 111).

[114] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014 (cit. on p. 111).

[115] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015 (cit. on p. 112).

[116] K. D. Julian and M. J. Kochenderfer, "Neural network guidance for UAVs," in *AIAA Guidance, Navigation, and Control Conference*, 2017 (cit. on pp. 118, 131).

[117] T. Deyle, C. C. Kemp, and M. S. Reynolds, "Probabilistic UHF RFID tag pose estimation with multiple antennas and a multipath RF propagation model," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008 (cit. on p. 125).

[118] T. Ward, E. L. Pasiliao, J. M. Shea, and T. F. Wong, "Autonomous navigation to an RF source in multipath environments," in *IEEE Military Communications Conference*, 2016 (cit. on p. 125).

[119] H. Nishimura and M. Schwager, "SACBP: Belief space planning for continuous-time dynamical systems via stochastic sequential action control," in *Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2018 (cit. on p. 131).

[120] K. D. Julian and M. J. Kochenderfer, "Autonomous distributed wildfire surveillance using deep reinforcement learning," in *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2018 (cit. on p. 131).

[121] W. Giernacki, M. Skwierczyński, W. Witwicki, P. Wroński, and P. Kozierski, "Crazyflie 2.0 quadrotor as a platform for research and education in robotics and control engineering," in *Methods and Models in Automation and Robotics (MMAR), 2017 22nd International Conference on*, 2017 (cit. on p. 132).

[122] M. Bartolucci, J. A. del Peral-Rosado, R. Estatuet-Castillo, J. A. García-Molina, M. Crisci, and G. E. Corazza, "Synchronisation of low-cost open source SDRs for navigation applications," in *ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing (NAVITEC)*, 2016 (cit. on p. 132).

[123] RTL-SDR.com, *Kerberossdr preview: A 4x coherent RTL-SDR for direction finding, passive radar and more*, Aug. 2013. [Online]. Available: `https://www.rtl-sdr.com/hydrasdr-preview-a-4x-coherent-rtl-sdr-for-direction-finding-passive-radar-and-more/` (cit. on p. 132).

[124] H. Shin, K. Choi, Y. Park, J. Choi, and Y. Kim, "Security analysis of FHSS-type drone controller," in *International Workshop on Information Security Applications*, 2015 (cit. on p. 133).

[125] R. P. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2003 (cit. on p. 133).