INFORMATION-THEORETIC ACTIVE PERCEPTION FOR MULTI-ROBOT TEAMS

Benjamin Charrow

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the
Degree of Doctor of Philosophy

2015

Vijay Kumar, Supervisor of Dissertation
UPS Foundation Professor of Mechanical Engineering and Applied Mechanics

---

Nathan Michael, Co-Supervisor of Dissertation
Assistant Research Professor of Robotics

Lyle Ungar, Graduate Group Chairperson
Professor, Computer and Information Science

Dissertation Committee

Vijay Kumar, Professor of Mechanical Engineering and Applied Mechanics
Nathan Michael, Assistant Research Professor of Robotics
Camillo J. Taylor, Professor of Computer and Information Science
Daniel D. Lee, Professor of Electrical and Systems Engineering
Alejandro Ribeiro, Associate Professor of Electrical and Systems Engineering
Mac Schwager, Assistant Professor of Mechanical Engineering

Acknowledgements

First and foremost I want to thank my advisors, Vijay Kumar and Nathan Michael. Throughout my entire Ph.D., they gave me a great deal of freedom to pick projects that I was excited about and then provided enough concrete guidance for me to produce results. They were also invaluable in teaching me how to present ideas in papers, presentations, and conversations, so that others would find our work as interesting as I do.

My entire committee also provided valuable feedback on all of my work and pushed me to make it better.

Many of the projects in this thesis were done in close collaboration with several talented people. In particular, I want to thank Sikang Liu, who helped immensely with all of the quadrotor and mapping experiments. Greg Kahn, Sachin Patil, Ken Goldberg, and Pieter Abbeel also deserve special mention for all their ideas and expertise on trajectory optimization. It was a lot of fun writing a paper and running experiments with people on the other coast. I also can't count how many conversations I had with Philip Dames and Nikolay Atanasov about information theory and robotics; this thesis is definitely much better for having known them. Pratap Tokekar also has my thanks for his help in formalizing ideas on getting air and ground robots to cooperate and for his endless knowledge of combinatorial optimization. I don't have space to list all of their names, but grad school was substantially more enjoyable due to everyone in MRSL and GRASP. They all have my heartfelt thanks.

I also want to thank my entire family, especially Caroline. She has constantly supported me and kept my spirits high, even when my robots could only crash into walls. There is no way that I could have finished this without her.

ABSTRACT

INFORMATION-THEORETIC ACTIVE PERCEPTION FOR MULTI-ROBOT TEAMS

Benjamin Charrow

Vijay Kumar

Nathan Michael

Multi-robot teams that intelligently gather information have the potential to transform industries as diverse as agriculture, space exploration, mining, environmental monitoring, search and rescue, and construction. Despite large amounts of research effort on active perception problems, there still remain significant challenges. In this thesis, we present a variety of information-theoretic control policies that enable teams of robots to efficiently estimate different quantities of interest. Although these policies are intractable in general, we develop a series of approximations that make them suitable for real time use.

We begin by presenting a unified estimation and control scheme based on Shannon's mutual information that lets small teams of robots equipped with range-only sensors track a single static target. By creating approximate representations, we substantially reduce the complexity of this approach, letting the team track a mobile target. We then scale this approach to larger teams that need to localize a large and unknown number of targets.

We also examine information-theoretic control policies to autonomously construct 3D maps with ground and aerial robots. By using Cauchy-Schwarz quadratic mutual information, we show substantial computational improvements over similar information-theoretic measures. To map environments faster, we adopt a hierarchical planning approach which incorporates trajectory optimization so that robots can quickly determine feasible and locally optimal trajectories. Finally, we present a high-level planning algorithm that enables heterogeneous robots to cooperatively construct maps.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Gathering information quickly and efficiently is a fundamental task for many real-world applications of robotics. Inspecting ship hulls for structural defects [36], building 3D models of damaged buildings [83], determining which crops need to be irrigated [2], and localizing devices in smart buildings [107] are all examples of information gathering tasks with broad commercial applicability. At their core, they are "active perception" problems which require robots that can both estimate quantities of interest and autonomously take actions to improve those estimates [4].

Teams of robots are particularly appealing for these types of problems. Unlike individual robots, they can obtain multiple sensor measurements from different places at the same point in time. When each robot is only equipped with sensors that provide limited information like RF antennas, teams are able to estimate quantities that an individual robot cannot. Even when using information rich sensors like cameras and laser range finders, the time it takes to gather information can decrease super-linearly with the size of the team [134]. Different types of robots also often have substantially different capabilities in terms of available sensors, where they can move, and how long they can be active before they run out of power. Consequently, teams comprised of heterogeneous robots are more capable than any individual robot.

However, designing teams of robots that can effectively gather information is a chal-

lenging problem that encompasses many different areas of robotics research. For example, robots must be capable of autonomously navigating to various parts of an environment. This requires robots that are capable of planning paths that do not collide with obstacles or other robots and using controllers that can generate inputs to actuators such that robots follow those paths. Navigation also requires that each robot is capable of determining where it is in space, a task known as localization. Depending on whether or not the robots are operating in an environment whose essential structure is known, robots may also need to solve the mapping problem, which requires them to build a detailed model of the environment they are in. In addition to all of these tasks, active perception problems require robots to estimate external quantities such as the location of a lost piece of medical equipment or the exact geometry of a building. As much as possible, a team of robots must effectively coordinate its actions.

Although each of these subproblems is challenging in its own right, the primary focus of this thesis is on principled and computationally efficient information gathering strategies for multi-robot teams. In the remainder of this chapter, we discuss some of the primary research challenges in designing such strategies. We then summarize the remainder of this thesis which describes a series of active perception problems and corresponding strategies to solve them.

1.1 Research Problems

The ultimate goal of an active perception problem is to estimate some unknown quantity of interest. Most modern perception approaches are probabilistic; instead of forming a single concrete guess of what something is, they determine a probability distribution over possible values that it could be. Consequently, the goal of an active perception strategy is to reduce the uncertainty of the probabilistic estimate as quickly as possible. To formally define this goal of "uncertainty reduction," we use information theory, a branch of mathematics that provides several tools for reasoning about the relative uncertainty of different distributions.

Given the goal of reducing uncertainty, there are a number of high-level characteristics

that we want any strategy to exhibit:

1. *General.* Given the wide variety of active perception problems, it is desirable to develop algorithms that generalize across different sensing modalities and enable robots with different mobility constraints to reduce an estimate's uncertainty as much as possible.
2. *Clarifying.* The faster robots can obtain low uncertainty estimates the better. This means that an approach should take actions that are likely to reduce an estimate's uncertainty.
3. *Online.* Robots continuously gather measurements as they take actions, enabling them to gradually form more certain estimates. An active perception approach should be able to take advantage of this increased information and adapt its actions to exploit everything that it knows.
4. *Collaborative.* An estimate's uncertainty can often be reduced faster by effectively coordinating multiple robots. Strategies that can achieve this are preferable to those that can't.

Developing a control policy that achieves all of these goals is difficult. Strategies that use approximations and simplified models may be quick to compute, but the selected actions may not substantially reduce the uncertainty of the estimate. Conversely, robots may be able to use detailed probabilistic models to predict how their actions are likely to improve the estimate. Unfortunately, given the inherent uncertainty in these predictions, computing these policies can take a long time. This issue is particularly important in time-critical scenarios such as search and rescue.

Although there are many different types of active perception problems, there are two broad classes that can be addressed: estimating low-dimensional state with low-information sensors and estimating high-dimensional state with high-information sensors.

A characteristic example of the low-dimension low-information problem is range-only target localization, where robots must estimate the position of a target only using measurements of their relative range (i.e., measurements contain no information about the relative

angle between a robot and the target). In addition to being a challenging active perception problem, range-only target localization can also be used to solve real world problems. For example, in the near future, automated buildings will use a large number of devices for a variety of services including power and water monitoring, building security, and indoor localization for smart phones [107]. Effectively using and maintaining this many devices will require knowing where each of them is located. A cost-effective way of getting this information would be to equip each device with RF or audio based range-only sensors [74, 99]. This approach could even work when sensors are embedded in a building's walls [43]. Because range-only sensors only provide limited information about a device's location and having humans localize all devices would be a time consuming and error prone task, it is natural to try and automate the process using a team of robots.

On the opposite extreme of active perception are problems like active mapping which require robots to build a complete and low uncertainty model of the environment that they are operating in. These maps typically consist of probabilistic estimates of the presence and location of obstacles and require the use of sensors like laser-range finders or 3D cameras like the Microsoft Kinect. Active mapping is then a problem that involves estimating high-dimensional state with high-information sensors. Being able to autonomously construct these maps has a large number of potential applications like inspecting ship hulls for defects or mines [36], exploring the surface of other planets [30], and assisting in disaster relief by giving safety workers a clear idea of how buildings are damaged [83].

Studying both range-only localizing and active mapping is useful for understanding approaches to active perception. A methodology that can address both types of problems can be considered general, because each problem requires robots to exhibit different types of behavior. To reduce uncertainty in range-only localization, robots must account for the lack of angular information their sensors provide and gather complementary measurements to the same target. In contrast, sensors used for active mapping provide information about a robots' immediate surroundings and they primarily need to gather measurements throughout the environment. These differences also mean that teams of robots must exploit different

collaboration strategies. In range-only localization teams can reduce uncertainty faster by gathering multiple measurements to the same target, but in active mapping they can often reduce uncertainty faster by observing different parts of the map.

1.2 Thesis Overview

The primary goal of this work is to develop computationally tractable information-theoretic control laws that enable multi-robot teams to effectively coordinate their actions to gather measurements that reduce the uncertainty of estimates. This thesis makes a variety of contributions that make substantial progress towards solving the research problems outlined in the previous section:

1. Ch. 3 introduces a basic form of the range-only localization problem, in which a small team of robots must localize a single static target in a known environment. We develop an accurate estimation scheme for a commercially available RF range-only sensor as well as a mutual information based control law.
2. Ch. 4 develops techniques for approximating mutual information, enabling it to be calculated substantially faster. This enables the approach from Ch. 3 to track a mobile target, by enabling the team to consider the impact of multiple measurements over time while planning, going beyond a greedy one-step maximization. Importantly, we prove that the approximations introduce bounded error. This proof also bounds the difference in mutual information between different control inputs, which makes it easier to generate appropriate actions.
3. Ch. 5 extends the range-only problem to the case where a team of robots must localize an unknown number of targets throughout an environment. It also addresses the exponential complexity inherent in multi-robot planning problems, enabling larger teams of robots to effectively work together. The computational complexity of the resulting control strategy is polynomial in all relevant variables.
4. Ch. 6 examines the use of alternative information-theoretic objectives, like Cauchy-Schwarz quadratic mutual information, for an active mapping problem. In contrast

to many other approaches, we explicitly model the dependence of separate measurements over multiple time steps to develop online algorithms that search over multiple, multi-step actions. This chapter also examines how information-theoretic control laws generalize across different platforms by conducting experiments where individual air and ground robots map different environments.

5. Ch. 7 shows how a hierarchical planning framework combined with trajectory optimization can substantially improve the performance of active mapping strategies, by enabling robots to quickly compute feasible and locally optimal information-theoretic trajectories.
6. Ch. 8 introduces a method for generating high-level plans for heterogeneous robots that are exploring and mapping an environment. The goal of this work is to enable air and ground robots to cooperatively build better maps than either could individually.

All of these contributions build on a large body of research in robotics and information-theory. Ch. 2 discusses this work, and also compares the information-theoretic approach to other active perception strategies. To conclude the thesis, we summarize all contributions in Ch. 9 and discuss some potential directions for future research.

Chapter 2

Background and Related Work

This chapter introduces the basic concepts of information theory and how it can be used to develop an active perception control law. It also surveys recent research in active perception that is relevant to this thesis.

2.1 Information Theory

Information theory was developed in the 1940's by Shannon [115] and others. Shannon's work was motivated by designing ways to encode, transmit, and decode messages without them being corrupted by noise.

One fundamental contribution of the theory is *Shannon's entropy*, a general measure of the uncertainty of a random variable. For a discrete random variable X, that can take values in the set $\mathcal{X} = \{x_1, \ldots, x_N\}$, Shannon's entropy is defined as:

$$\mathrm{H}[X] = \mathbb{E}_{p(X)}\left[-\log_2 p(X = x)\right] = -\sum_{x \in \mathcal{X}} p(X = x) \log_2 p(X = x) \tag{2.1}$$

In general, $\mathrm{H}[X]$ is non-negative and 0 if and only if X is not random (i.e., takes a value with probability 1). The units of Shannon's entropy are bits when the base of the logarithm is 2 and nats when the natural logarithm is used.

To gain some intuitive understanding of entropy, consider the outcome of a fair coin toss, F. Because $p(F = Heads) = p(F = Tails) = 0.5$, F's entropy is $\mathrm{H}[F] = 1$ bit. This

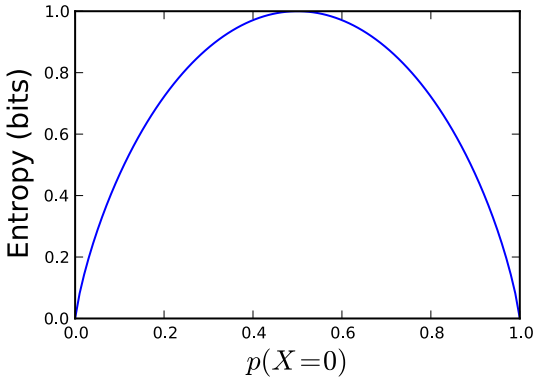


Figure 2.1: Entropy of a Bernoulli random variable X.

value is sensible, as a single bit can encode two different values, and both outcomes are equally likely. For comparison, suppose B is a biased coin where $p(B = Heads) = 0.99$ and $p(B = Tails) = 0.01$. B's entropy is $\mathrm{H}[B] = 0.08$ bits. This is also a sensible value as the outcome of B is quite certain; we would be surprised if we got a tails. Fig. 2.1 shows that the entropy of a generic Bernoulli random variable. As one would hope for a measure of uncertainty, entropy is highest when the outcome is most uncertain, and gradually decreases to 0 as the outcome becomes more certain.

Shannon's entropy can be extended by analogy to continuous distributions. If X is a continuous random variable, then its *differential entropy* is:

$$\mathrm{H}[X] = \mathbb{E}_{p(X)}\left[-\log p(X = x)\right] = -\int p(X = x) \log p(X = x)\, dx \tag{2.2}$$

Unlike discrete entropy, differential entropy can be negative. While it has a similar form to discrete entropy, it is not the limit of discrete entropy, and in fact differs by an infinite offset [27]. Despite this, differential entropy is similar conceptually, and we will use the two concepts interchangeably. As many quantities in robotics are continuous, most of the work in later chapters uses differential entropy.

Another important question in information theory is "what is the uncertainty of a random variable given the outcome of a different random variable?" In other words, suppose

we have two random variables X and Z, and we want to know the entropy of X given Z. This would be straightforward if Z were known; it would simply be the entropy of the distribution $p(X \mid Z = z)$. However, Z is also a random variable, so we take an expectation over it:

$$\mathrm{H}[X \mid Z] = \mathbb{E}_{p(Z)}\left[\mathrm{H}[X \mid Z = z]\right] \tag{2.3}$$

$$= \sum_{z \in \mathcal{Z}} p(Z = z) \left(- \sum_{x \in \mathcal{X}} p(X = x \mid Z = z) \log_2 p(X = x \mid Z = z) \right) \tag{2.4}$$

Similar to differential entropy, conditional differential entropy can be defined for continuous random variables by replacing the appropriate sums with integrals.

Together, entropy and conditional entropy can be used to measure the likely decrease in uncertainty of a random variable. This quantity is known as mutual information:

$$\mathrm{I}_{\mathrm{MI}}[X; Z] = \mathrm{H}[X] - \mathrm{H}[X \mid Z] \tag{2.5}$$

$$= \mathrm{H}[Z] - \mathrm{H}[Z \mid X] \tag{2.6}$$

Note that for both discrete and continuous distributions, mutual information is always non-negative and 0 if and only if the two random variables are independent. The non-negativity has an interesting interpretation that "information never hurts" [27]; learning the outcome of a random variable cannot increase uncertainty.

One important property of mutual information is that it can be expressed in two different ways. In other words, each random variable contains the same amount of information about the other. This is helpful, because when Z is the measurement and X is the state, $p(Z \mid X)$ is the measurement model, which can often be modeled in robotics. This makes (2.6) easier to calculate than (2.5), as (2.5) requires calculating $p(X \mid Z)$ which is either the posterior distribution or an approximate inverse measurement model [134].

It is important to know that Shannon's entropy is not an ad-hoc measure. Shannon [115] derived the expression by specifying axioms that any measure of uncertainty must

satisfy. For example, one requirement is that the uncertainty of independent events must be the sum of the uncertainty of the individual events. Shannon described five axioms and proved that (2.1) is the *only* definition of uncertainty that satisfies all of them.

However, there are alternative axiomizations of uncertainty, notably by Renyi [104]. These measures only satisfy some of Shannon's axioms, but still provide useful characterizations of uncertainty. Renyi's α-entropy for continuous random variables is defined as:

$$\mathrm{H}_\alpha[X] = \frac{1}{1-\alpha} \log \int p^\alpha(X = x)\, dx \tag{2.7}$$

Any $\alpha > 1$ including $\alpha \to \infty$ results in a valid entropy. In the limit as α goes to 1, Renyi's entropy converges to Shannon's entropy [100].

Our primary motivation for studying alternative forms of uncertainty is that Shannon's entropy and mutual information are hard to compute, except in certain special cases like Gaussian models. The source of this difficulty is the expectation over the logarithm in (2.1), which frequently prevents the integral from being evaluated analytically. For this reason, Renyi's quadratic entropy (RQE), also known as the collision entropy, is particularly interesting:

$$\mathrm{H}_2[X] = -\log \int p^2(X = x)\, dx \tag{2.8}$$

As noted by Principe [100], this form is computationally useful because 1) the logarithm appears outside the integral and 2) when $p(X)$ is a Gaussian or a Gaussian mixture model, the integral can be calculated analytically, because all of the integrals are the products of Gaussians.

There are also computational advantages in considering concepts that are similar to mutual information. One way of generating these is by re-expressing mutual information in terms of the Kullback-Leibler divergence (KL divergence or KLD). KL divergence is one way of measuring the "distance" between two densities. For two densities f and g, it is

defined as:

$$\mathrm{D}_{\mathrm{KL}}[f \mid\mid g] = \mathbb{E}_f \left[\log f/g\right] = \int f(x) \log \frac{f(x)}{g(x)}\, dx \tag{2.9}$$

KL divergence is non-negative and 0 if and only if f and g are equal almost everywhere. In general, it is not symmetric (i.e., $\mathrm{D}_{\mathrm{KL}}[f \mid\mid g] \neq \mathrm{D}_{\mathrm{KL}}[g \mid\mid f]$) and does not obey the triangle inequality, making it a pseudo-metric [27]. Interestingly, the mutual information of two variables X and Z, can be expressed as the KL divergence between their joint distribution, $p(X, Z)$, and the product of their marginal distributions, $p(X)p(Z)$:

$$\mathrm{I}_{\mathrm{MI}}[X; Z] = \mathrm{D}_{\mathrm{KL}}[p(X, Z) \mid\mid p(X)p(Z)] \tag{2.10}$$

Because $p(X, Z) = p(X)p(Z)$ if and only if X and Z are independent, mutual information can be viewed as a measure of how dependent X and Z are. If they are independent, then there is no difference between the product of the marginals and the joint distribution, so their mutual information is 0. But if X and Z are very dependent – the outcome of one random variable changes likely outcomes of the other – then their mutual information will be high.

Viewing information as a measure of dependence between random variables makes it easier to define alternative measures of information. Specifically, we can define a new notion of information by replacing KL divergence in (2.10) with a different probability measure. Gibbs and Su [44] provide a good survey of different metrics and how they are related. Principe [100] also discusses several different options. Of particular interest is the Cauchy-Schwarz divergence (CS divergence), which for two densities f and g is defined as:

$$\mathrm{D}_{\mathrm{CS}}[f \mid\mid g] = -\log \frac{\left(\int f(x)g(x)\, dx\right)^2}{\left(\int f^2(x)\, dx\right)\left(\int g^2(x)\, dx\right)} \tag{2.11}$$

The CS divergence's name comes from the Cauchy-Schwarz inequality, which is used to prove

that it is non-negative and 0 if and only if the densities are equal almost everywhere [100]. It can be used to define the Cauchy-Schwarz quadratic mutual information (CSQMI) between random variables:

$$\mathrm{I}_{\mathrm{CS}}[X; Z] = \mathrm{D}_{\mathrm{CS}}[p(X, Z) \mid\mid p(X)p(Z)] \tag{2.12}$$

There is a connection between CS divergence and Renyi's quadratic entropy that is quite similar to the relationship between KL divergence and Shannon's entropy. This similarity then links mutual information and CSQMI. KL divergence can be written in terms of Shannon's cross entropy and Shannon's entropy:

$$\mathrm{D}_{\mathrm{KL}}[f \mid\mid g] = -\mathbb{E}_f [\log g] - \mathrm{H}[f] \tag{2.13}$$

The cross entropy, $\mathbb{E}_f [-\log g]$, measures the efficiency of encoding a message using a distribution given by g rather than the true distribution f. Similarly, Rao [102] showed that CS divergence can be expressed as:

$$\mathrm{D}_{\mathrm{CS}}[f \mid\mid g] = (-\log \mathbb{E}_f [g] - \mathrm{H}_2[f]) + (-\log \mathbb{E}_g [f] - \mathrm{H}_2[g]) \tag{2.14}$$

$\mathrm{H}_2[\cdot]$ is Renyi's quadratic entropy, and $-\log \mathbb{E}_f [g]$ can be viewed as Renyi's quadratic cross entropy. The grouped terms are thus completely analogous to KL divergence, suggesting that CSQMI and mutual information express similar concepts.

As a brief comment on notation, the observant reader may notice that (2.13) and (2.14) take the entropy of a probability density function, but we defined entropy (2.2) as a function of a random variable. This difference is not significant, as each random variable has a corresponding density function. We use both conventions throughout this thesis. Additionally, in this section we have used capital letters for random variables and lower case letters to denote their instantiations. As is common in most robotics papers, the remainder of this thesis will use lower case letters for both quantities, as it will often be clear from context

what is intended.

We defined all of the concepts in this section in terms of random variables, but the definitions for random vectors are nearly identical. The only change is that the domain and dimensionality of the probability mass and density functions increases. This is useful as it gives us a consistent notation for handling all cases. When it makes the presentation clearer, we will use boldface to denote vector quantities.

2.2 Information Based Control for Robotics

Given its basis as a theory of communication, information theory may not seem connected to active perception. However, the fundamental goal of active perception is to reduce uncertainty of an estimate, and information theory provides many ways to quantify this intuitive notion. In this section, we use a simple range-only localization problem to explain how information theory can be applied to active perception. The work presented in later chapters will formalize and expand this approach, but the fundamental ideas will remain the same.

A generic estimation problem is to estimate some state, x, given a sequence of measurements taken from time 1 to time t: $[z_1, \ldots, z_t] = z_{1:t}$. The posterior distribution over the state is $p(x \mid z_{1:t})$. At time t, the robot must take an action to obtain a better estimate of x. If the state of the robot is c, then the measurements that it gets should depend on the true state as well as the robot's position; the measurement model – distribution over measurements – can be specified as $p(z \mid x, c)$. After the robot takes an action, it will get a new measurement, update its estimate, repeat the process. One reasonable control policy is to select an action that will likely result in the smallest entropy (i.e., uncertainty) of the posterior estimate at time $t+1$. Formally, given a set of possible actions $A = \{a_1, \ldots, a_N\}$, this control policy would be:

$$a^* = \arg\min_{a \in A} \mathrm{H}[x \mid z_{t+1}, z_{1:t}, a] \tag{2.15}$$

Note that measurements $z_{1:t}$ have already been collected (i.e., are instantiations of random

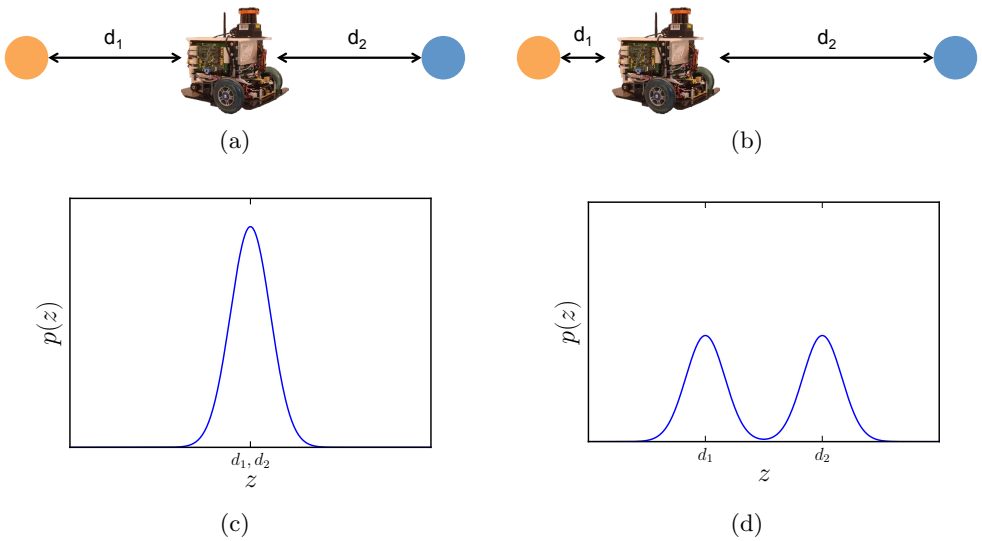


Figure 2.2: 1D active perception problem. (a) Orange and blue dots show two possible locations for a target. When the robot is equidistant from both, $d_1 = d_2$, the robot can predict that future measurements, (c), will not help distinguish between the two hypotheses. (b) and (d) show that the robot can predict that moving closer to one hypotheses is a better choice, because the received measurement depends on the target's true location.

variables), whereas x and z_{t+1} are unknown as they're in the future.

Minimizing the conditional entropy of the future estimate, (2.15), is equivalent to maximizing mutual information. Using (2.5):

$$\arg\min_{a \in A} \mathrm{H}[x \mid z_{t+1}, z_{1:t}, a] = \arg\min_{a \in A} \mathrm{H}[x \mid z_{1:t}] - \mathrm{I}_{\mathrm{MI}}[x; z_{t+1} \mid z_{1:t}, a] \tag{2.16}$$

$$= \arg\max_{a \in A} \mathrm{I}_{\mathrm{MI}}[x; z_{t+1} \mid z_{1:t}, a] \tag{2.17}$$

This equivalence serves as the primary motivation for maximizing mutual information and other quantities that measure the dependence between random variables.

To understand this type of policy more concretely, consider the simple 1D problem shown in Fig. 2.2. Here, there are two possible locations along the line where the target can be, and a robot is equipped with a sensor that measures the absolute value of the distance to the target, perturbed by some small amount of noise. Using its current estimate of where the target is, and the measurement model, $p(z \mid x, a)$, the robot can calculate

the distribution over measurements that it is likely to receive using the marginalization rule of probability: $p(z \mid a) = \int p(x)p(z \mid x, a)\, dx$. Fig. 2.2c and Fig. 2.2d show the distribution over measurements that the robot can predict at the current time t. Due to the symmetry of the target's position in Fig. 2.2c, the uncertainty of the measurements, $\mathrm{H}[z]$, is due to the uncertainty inherent in the sensor itself. As this uncertainty is given by $\mathrm{H}[z \mid x]$, $\mathrm{H}[z] = \mathrm{H}[z \mid x]$, meaning the mutual information is close to 0. In contrast, the uncertainty of measurements in Fig. 2.2d is substantially higher, due to the fact that it's unclear whether the robot is close to or far from the target. In this case $\mathrm{H}[z] >> \mathrm{H}[z \mid x]$, so mutual information will be high. Consequently, an information based strategy will pick the action that brings it closer to the target, which is the better choice.

The simple 1D example illustrates the benefits of using an information based control policy. A primary advantage is that because this approach uses the same probability models as the estimation process, it naturally accounts for sensor limitations like noise or limited field of view. Additionally, maximizing information naturally extends to multiple time steps, robots, and sensing modalities, as each of these have well defined effects on the posterior distribution.

The 1D example also illustrates some of the drawbacks of information. While the expressions are quite general, they are difficult to compute, as they require integrating over the joint state and measurement spaces. Except in more restricted settings like linear-Gaussian models, the requisite integrals are difficult to evaluate. Developing appropriate approximations and measures to avoid these issues is a central focus of this thesis.

2.3 Limited Information Active Perception

There is extensive work on estimation and active perception for limited sensing domains. Two widely studied problems are range-only localization and bearing-only localization.

Olson et al. [95], Djugash et al. [32, 33] both considered situations where a single mobile robot localizes itself and the nodes in a sensor network with many nodes using range measurements. Jourdan et al. [62] and Prorok and Martinoli [101] examined methods for

handling disturbances to Ultra-Wideband (UWB) radios that are caused by line-of-sight conditions and different construction materials. Researchers have also examined situations where the target is moving. For example, Hollinger et al. [54] studied how to estimate a mobile target with fixed and uncontrollable range radios. While the preceding approaches used a Bayesian filter to obtain estimates, researchers have also formulated range-only estimation as a non-linear optimization problem [88] or a spectral learning problem [9]. For active perception, using a recursive Bayesian filter for estimation has the advantage of providing an up to date belief at each point in time. All of these works used a human operator to manually drive a robot to gather measurements.

Given that estimation often produces probabilistic estimates, many researchers have formulated the active perception problem as an optimization of some information-theoretic quantity. Grocholsky [49] and Stump et al. [132] designed controllers which maximized the rate of change of the Fisher information matrix (FIM) to localize a stationary target. As the FIM lower bounds the covariance (i.e., uncertainty) of any unbiased estimator [27], minimizing its inverse enables better estimates. However, this work assumed that the belief is Gaussian, which is not the case for typical range measurements. More recently, Levine et al. [77] looked at how to combine the FIM with rapidly-exploring random trees to plan informative paths for robots with non-trivial dynamics. Martínez and Bullo [81] also examined using the determinant of the FIM for target tracking.

Of particular relevance to this thesis are works that use mutual information as their objective function. Hoffmann and Tomlin [52] showed how particle filters can be used in obstacle-free environments with non-linear sensor models to calculate a mutual information based control law. They limited coordination between robots to keep the observation space small. Ryan and Hedrick [110] investigated information-theoretic control and developed a receding horizon controller for a single mobile robot to track a mobile target. Their approach considers the impact of multiple measurements over time, and they approximate mutual information using a sampling based algorithm. However, their approach is computationally expensive. Most of the work in this thesis uses dynamically simpler ground robots, but our

approximations enable us to calculate control inputs for multiple robots in real-time.

In probabilistic estimation problems, it is often difficult to obtain theoretical guarantees about the performance of an active control policy. One exception is Vander Hook et al. [137], who developed an approach for a single target bearing-only sensor problem. They proved that the time it takes their approach to decrease the covariance of the estimate by any given factor is constant factor competitive with any active control policy. While this result is limited to bearing-only sensors and open environments where a robot can move freely, it is still an interesting guarantee, particularly as the bound is with respect to time. The focus of this thesis is different, as we develop algorithms for teams of robots with a variety of sensors that take intelligent actions even when their movement is constrained.

Any active control policy faces the question of how to parameterize the actions the robots can take. One popular approach is to use motion primitives, which has the advantage that the objective only needs to be computed over a discrete set and can be used to incorporate other properties such as obstacle avoidance. However, it is often possible to locally optimize an initial action for the robots by considering the gradient of an objective. Spletzer and Taylor describe a sampling based method for evaluating the gradient of several different types of objectives [126, 125] while Schwager et al. [112] give an explicit characterization of the gradient of mutual information with respect to the configuration of the team.

There are alternative formulations of active perception problems, in particular those that build on geometry. The book by Bullo et al. [11] describes a wide variety of geometric coverage and deployment problems. In contrast to the information based strategies, the geometric approaches often use a set based "bounded uncertainty" model. In this setting, it is often possible to theoretically analyze the performance of a system, which is typically difficult with a probabilistic formulation. For example, Spletzer and Taylor [127] examined a cooperative localization problem where a team must take actions to estimate the position of its own members. They model measurements as having bounded uncertainty, which means the set of possible team configurations that are consistent with received measurements are a convex polytope within the team's full configuration space. This geometric formaliza-

tion allows them to provide guarantees on the rate of convergence for their active search procedure. Geometric and probabilistic strategies are not mutually exclusive. For example, Schwager et al. [113] examined different ways of combining geometric and probabilistic objectives.

A number of researchers have studied issues relating to communication. Kassir et al. [68] developed a distributed information-based control law that explicitly modeled the cost of sending data between agents. In a range-only tracking problem, they showed how their algorithm reduced the amount of inter-agent communication in a Decentralized Decision Making algorithm without significantly affecting the system's overall performance. In related work, Gil et al. [45] developed a controller to position a team of aerial robots to create a locally optimal communication infrastructure for a team of ground robots. Julian et al. [64] developed distributed algorithms for calculating mutual information to estimate the state of static environments using multi-robot teams.

In many scenarios, the number of targets that needs to be localized is unknown a priori, so there has also been research on strategies to discover and precisely localize multiple targets. Dames and Kumar [29] considered a scenario where a team of robots must perform these tasks simultaneously using range-only devices. They assumed measurements have an unknown association, which for computational reasons limits their control law to only consider whether or not a future measurement will be received (i.e., the possible value of the measurement is ignored). All of the range-only work in this thesis assumes measurements have a known association, but as discussed in Ch. 5, this is reasonable as RF-based range sensors typically have a unique identifier like a MAC address. Carpin et al. [13] also localized multiple targets, using a variable resolution binary filter to discover targets throughout an environment and maximize mutual information to control an individual robot. Pursuit-evasion games where a team must find or maintain visibility to targets [25] are also quite relevant.

2.4 Information Rich Active Perception

Information rich problems are those where a robot must estimate high dimensional state. A prototypical example of this problem is active mapping, for which there is a considerable body of prior work [133, 128]. One key question for any active mapping approach is how to represent a static map of the environment. Such representations include, but not limited to, topological maps, landmark-based representations, elevation grids, point clouds, meshes, and occupancy grids [134].

Given a map representation, strategies for mapping and exploration can be primarily grouped into two categories: (i) Frontier-based, and (ii) Information-gain based strategies.

Frontier-based strategies [142] are primarily geometric in nature and travel to the discrete boundary between the free and unknown regions in the map. Extensions of this strategy have been successfully used for building maps of unknown environments in 2D [47, 12]. Holz et al. [56] provide a comprehensive evaluation of frontier-based strategies in 2D environments. Direct extensions to voxel grid representations of 3D environments using frontier voids have also been proposed [42, 35]. Shade and Newman proposed a combination of a frontier-based and a local vector-field based strategy to generate shorter exploration trajectories [114]. Although they do not compute frontiers directly, Shen et al. [117] used a similar strategy with a particle-based representation of free space for computational tractability. While frontiers are somewhat difficult to use in 3D environments [117], they do provide a useful way to generate potential actions for information-theoretic objectives [129].

Information-gain based mapping strategies optimize an information-theoretic measure for exploration. Prior work has investigated minimization of the map entropy, which is a measure of uncertainty associated with all grid cells [84]. Several methods have been proposed that choose to maximize mutual information to reduce map uncertainty [1, 112]. Of particular relevance to the work in this thesis is the recent approach by Julian et al. [66], where a beam model is employed to calculate the mutual information between a laser range finder and a 2D occupancy grid. They show that a mutual information based control law will eventually cause a robot to drive towards unknown space. However, these approaches

use a greedy controller with a one-time step look ahead. Soatto [123] suggests planning over multiple time steps to deal with the issue of a greedy explorer getting stuck in local minima. To address this issue, researchers have proposed considering a discrete set of actions and executing the one that maximizes information gain [10, 40]. Rocha et al. [105] use the gradient of map entropy to select promising frontiers for exploration. These strategies do not locally optimize the robot trajectory with an intent to gain more information as the robot is traveling to the next selected frontier, which might result in inefficient mapping behavior.

Prior work has used trajectory optimization over a finite horizon to optimize information-theoretic criteria [90, 24, 110, 80]. However, these approaches have been primarily limited to continuous representations of the belief and measurements.

One drawback to many frontier-based and information-based strategies is that they only plan over a finite horizon. Particularly when a robot must travel to every destination in an environment, this can result in the robot repeatedly traversing the same parts of an environment in order to observe everything. When an uncertain but complete prior estimate is available, coverage strategies, which are geometric in nature, avoid this issue by planning paths that will observe every portion of the environment. Englot and Hover [36] developed such a strategy to build a detailed model of a ship's hull. Kollar and Roy [71] adopted a similar approach and framed exploration as a constrained optimization problem, but learned to predict how trajectories would reduce the robot and map's uncertainty using reinforcement learning.

Another consideration for any exploration strategy is whether or not to model the uncertainty of the robot itself. Throughout this entire thesis, we assume that the robot is capable of localizing itself using a simultaneous localization and mapping (SLAM) [134, 93, 75] system. Modern SLAM systems [93], which account for uncertainty in the map and the robot's state, are now reasonably robust, and in many situations it is still possible to achieve good performance without considering this uncertainty in the control policy. However, in cases where reliable localization is not possible, it is important to plan trajectories that actively

reduce the uncertainty of the robot's state and the map [10, 129, 136, 69, 60]. Stachniss et al. [129] also considered the active SLAM problem. Using a Rao-Blackwellized particle filter to estimate the joint distribution over maps and the robot's trajectory, they proposed selecting actions that maximize the expected reduction in the filter's entropy, accounting for uncertainty in the robot's pose as well as the uncertainty of the map. However, this approach is expensive, as it requires calculating entropy by repeatedly copying the filter and updating it with a randomly sampled measurement.

For planning approaches that seek to maximize mutual information, one important question is how close to optimal they are. Although these problems are typically NP-hard, building on work in combinatorial optimization by Nemhauser et al. [87], Krause and Guestrin [72] derived approximation guarantees for greedy maximizations of mutual information and other submodular set functions. These results were applied to mobile robot environmental monitoring problems by Singh et al. [119] and Binney et al. [7]. These guarantees only hold in offline settings where teams do not update their actions based on measurements they receive. Golovin and Krause [46] generalized these ideas to online settings, deriving performance guarantees for adaptive submodular functions. Unfortunately, these guarantees require all actions to be available to all robots at all times, making them not directly applicable to the problems discussed in this thesis. Hollinger and Sukhatme [53] develop a sampling based strategy for maximizing a variety of information metrics with asymptotic optimality guarantees. However, they assume that information is additive across multiple measurements (i.e., measurements are independent). This assumption limits cooperation in multi-robot settings (Ch. 3) and can lead to overconfidence when considering multiple measurements of the same quantity (Ch. 6). Unlike the guarantee provided by Vander Hook et al. [137], these guarantees are all with respect to an objective like mutual information and not a performance metric like time to reduce uncertainty by a given amount.

Besides active mapping, information-theoretic objectives have also been used for planning and control in robotics for related information rich tasks involving uncertainty such as inspection [55], environment modeling [120], extrinsic calibration of LIDAR sensors [79],

visual servoing [28], and active object modeling [140]. They have also been used for applications outside of active perception. For example, Kretzschmar and Stachniss [73] use mutual information as a criterion for storing a minimal number of laser scans toward map reconstruction.

For additional related work focused on active vision, see the survey article by Chen et al. [22].

Chapter 3

Estimation and Control for Localizing a Single Static Target

Having robotic teams that are capable of quickly localizing a target in a variety of environments is beneficial in several different scenarios. Such a team can be used in search and rescue situations where a person or object must be located quickly. Cooperative localization can also facilitate localization of robots within a team towards tasks like cooperative mapping or surveillance. Given these multiple applications, it is reasonable to equip the robots with additional sensors to support localization. In this chapter, we focus on the case where each robot is equipped with a range-only radio frequency (RF) sensor. These sensors provide limited information about the state of the target, necessitating the development of an active control strategy to localize it.

This chapter makes two primary contributions. The first is an experimental model for radio-based time-of-flight range sensors. The second is a general framework for active control which maximizes the mutual information between the robot's measurements and their current belief of the target position. Both of these pieces serve as building blocks for Ch. 4 and Ch. 5, which extend the approach to consider the effect of multiple measurements, larger teams, and to scenarios with mobile or multiple targets.

Both the estimation and control strategies presented are fully centralized and require

communication throughout the team. This limitation is typically not significant in the scenarios we consider as the team's only sensor is RF-based. If the team was in an environment where they could not communicate, they could not gather measurements either.

In the rest of this chapter we detail the estimation and control algorithms that enable a team of robots to successfully localize a single stationary target in non-convex environments. Our primary focus is on a series of experiments that we designed to test the approach across different indoor environments and a variety of initial conditions. Overall, we are able to repeatedly localize a target with an error between $0.7-1.6$ m using only two robots equipped with commercially available RF range sensors. A highlight of this work is that despite the limited information that the team has when they are at any fixed position, they are able to effectively coordinate and use their mobility so that the estimate of the target rapidly converges.

This chapter originally appeared in [14, 17].

3.1 Technical Approach

In our approach, the robotic team maintains a distribution over possible locations of the target's location using a particle filter. The team constantly seeks to maximize the mutual information between the current estimate of the target's state and expected future measurements. Both the filter and the control directions are computed in a centralized manner as illustrated in Fig. 3.1. This approach could be made tolerant to some types of communication limitations similar to the system presented by Dames and Kumar [29].

3.1.1 Measurement Model

We consider the case where the team's only way of sensing the target is through a *range*-only sensor. A standard approach for building these sensors is to use the round trip time (RTT) of a signal between two nodes. Because of this, any delay in the signal's propagation results in an over-estimate of the distance. Conversely, an overestimate of the signal's speed results in an under-estimation of the distance. We use the nanoPAN 5375 – a RTT RF sensor which operates in the 2.4GHz spectrum – and show that it exhibits both positive and negative

Table 3.1: Notation.

Symbol	Description
$\mathbf{x}$	Mobile target's 2D position
$\mathbf{z}$	Range measurements
t	Current time
σ_m^2	Measurement noise
σ_p^2	Process noise
$\mathbf{c}$	Destinations for the team
T	Length of future time interval
R	Size of team
M	Num. of particles for target prediction

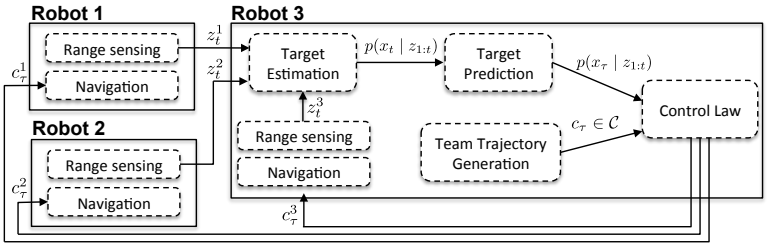


Figure 3.1: System overview with three robots. In this example, Robot 3 aggregates range measurements and estimates the location of the target. This estimate is used to predict the target's future locations over several time steps. The control law selects the trajectory that maximizes the mutual information between the target's predicted location and measurements the team expects to make. This trajectory is sent to the other robots which follow it.

biases, which are primarily a function of whether or not the sensors are in line of sight (LOS) or non line of sight (NLOS). Our data also show that the magnitude of the bias and variance increase with the true distance between the sensors.

There is significant empirical and theoretical support for treating LOS and NLOS measurements differently [103, 99]. Fig. 3.2 shows a histogram of biases which demonstrates this. RTT methods work best when radios are not obstructed by obstacles and have clear LOS conditions. However, when radios do not have LOS they are more likely to be affected by scattering, fading, and self-interference, causing non-trivial positive biases. We model the error of the measurement conditioned on the true state as:

$$p(z \mid \mathbf{x}) = \begin{cases} \mathcal{N}(z; \alpha_0 + r\alpha, r\sigma_L^2) & \text{LOS} \\ \mathcal{N}(z; \beta_0 + r\beta, r\sigma_N^2) & \text{NLOS} \end{cases} \tag{3.1}$$

where r is the true distance between the sensors and $\mathcal{N}(z; \mu, \sigma^2)$ is a normal random variable with mean μ and variance σ^2. α_0 and β_0 are the biases, while α and β determine how the biases change with distance. It is straightforward to calculate the maximum likelihood estimate (MLE) of these parameters for this model with labeled data. Throughout this paper, we assume that measurements are conditionally independent of each other given the

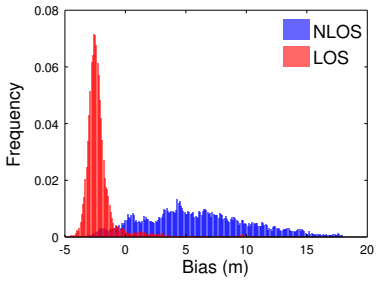


Figure 3.2: Normalized histograms of LOS and NLOS measurements. Both histograms have 20,000 measurements. LOS measurements typically underestimate distance (negative bias), while NLOS overestimate it (positive bias).

true state (i.e., $p(z_1, z_2 \mid \mathbf{x}) = p(z_1 \mid \mathbf{x})p(z_2 \mid \mathbf{x})$).

Models for TDoA sensors typically use a biased Gaussian with constant or time-varying variance [111, 62], whereas (3.1) is more similar to those used in power based range models. We have chosen this approach because, as Patwari et al. noted, the Gaussian model does not perform as well at large distances as the tails of the distribution become heavy [99]. Letting the variance increase with distance ensures that the model handles large deviations from the truth without using a Gaussian mixture model, which would increase the computational complexity of the control as we discuss later.

Because the error statistics of LOS and NLOS measurements are so different, it is important to correctly classify measurements. Using the wrong model to incorporate measurements can result in the filter converging to an incorrect location of the target.

One approach – assuming the robots have a map of the environment – is to use *geometric LOS* and check if a raycast between a hypothesized target location and the measuring robot intersects any walls. This works well with particle filters as measurements can be classified on a per particle basis.

The primary drawback to the geometric approach is that geometric LOS can be different from *RF LOS*. As we show experimentally, measurements can have a negative bias and relatively low variance – indicating RF LOS – even when the target and robot do not have geometric LOS.

We address this issue by using a Hidden Markov Model (HMM) to classify sequences of

measurements based on their observed mean and variance. By using the statistics of the measurements, the HMM typically differentiates RF LOS and RF NLOS more accurately than the geometric method because it does not rely on the assumption that geometric LOS predicts RF interference. Concretely, we model noise as a two state HMM with non-distance dependent Gaussian emissions. To classify a group of measurements, we subtract their mean and find the maximum likelihood sequence of hidden states using the Viterbi algorithm [8]. We train the HMM using the EM algorithm and a collection of unlabeled range measurement errors. Unlike the geometric method, the HMM approach uses the same measurement model to update all particles in the filter. In practice we have found this works better than classifying measurements on a per particle basis using (3.1) as the emission density.

Hidden Markov Models (HMMs) are widely used in several fields. They are a latent variable model that make the same conditional independence assumptions as the Kalman filter. The latent (i.e., unobserved) random variables are discrete and determine the emission (i.e., observed) random variables which can be continuous or discrete. Most relevant to what we present here, Morelli et al. [85] used an HMM to simultaneously localize a mobile beacon with static nodes and classify the line of sight (LOS) conditions of range measurements online. We only use an HMM to classify the LOS conditions, focus on how to coordinate mobile nodes, and report experimental results.

More sophisticated models for predicting RF interfence exist [143], but they often require detailed knowledge of the environment (e.g., the permittivity and conductivity of walls) or an extensive training period. However, in this paper we are mainly interested in accurate estimation and not in building models of RF propagation. Using an HMM is simpler, and still enables accurate estimation.

3.1.2 Estimation

Range measurements are a non-linear function of the target state and can easily lead to non-trivial multi-hypothesis belief distributions as well as rings and crescents. For these reasons, we use a particle filter for the estimation. Formally, let $\mathbf{x}_t = [x, y]$ be the state

of the target at time t and $\mathbf{c}_t^i$ be the 2D position of the i^{th} member of the team. The full configuration of the team is $\mathbf{c}_t = [\mathbf{c}_t^1, \ldots, \mathbf{c}_t^n]$. Each robot makes a 1D range measurement z_t^i as they move around the environment. Aggregating these measurements produces a vector $\mathbf{z}_t = [z_t^1, z_t^2, \ldots, z_t^n]$. Where appropriate, we will condition $\mathbf{z}_t$ on $\mathbf{c}_t$ to emphasize that the measurements depend on the configuration of the team.

The belief at time t is the distribution of the state conditioned on all measurements up to time t. A typical Bayesian filter incorporates measurements over time recursively and a particle filter approximates this as a weighted sum of Dirac delta functions [134]:

$$bel(\mathbf{x}_t) = \eta \; p(\mathbf{z}_t \mid \mathbf{x}_t) \int bel(\mathbf{x}_{t-1}) p(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{u}_t) \, d\mathbf{x}_{t-1} \approx \sum_j w_j \delta(\mathbf{x}_t - \tilde{\mathbf{x}}_j) \tag{3.2}$$

where η is a normalization constant, $\tilde{\mathbf{x}}_j$ is the location of the j^{th} particle and w_j is its weight. While this equation is standard, we wish to emphasize that the approximation is *discrete*. As we show in Sect. 3.1.3, this enables approximations of mutual information.

While the standard particle filter equations allow for the incorporation of non-linear control inputs, $\mathbf{u}_t$, in our scenario the target is stationary. Despite this, during experiments we injected noise into the system to avoid particle degeneracy problems. Specifically, we perturbed the polar coordinates of each particle in the local frame of the robot that is making the measurement with samples from a zero-mean Gaussian distribution. We also used a low variance resampler when the number of effective particles dropped below a certain threshold. All of these techniques can be found in standard books on estimation [134].

Assuming that the team has a map of the environment, they could sometimes improve their estimate of the target's location by rejecting particles that fall in occupied space. We do not take this approach because in some situations the target may be in a region that the team considers occupied (e.g., the target is on top of a desk or inside a room not on the team's map).

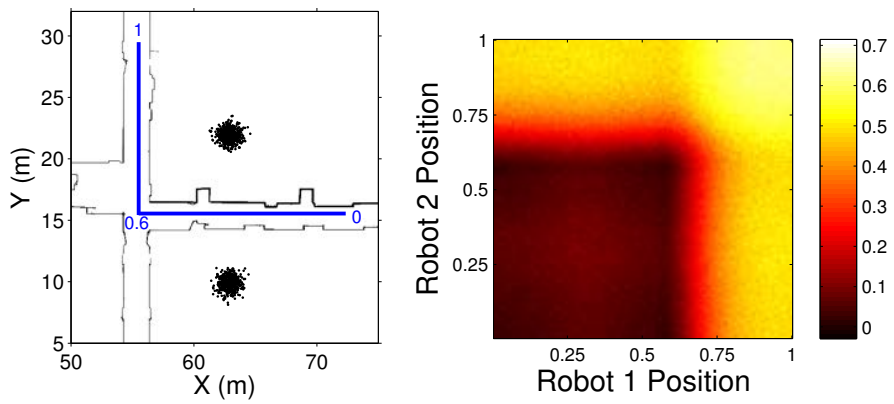
Particle filters are expressive, but can also be computationally expensive, requiring many samples to accurately estimate a distribution. While we use KLD sampling [41] to

keep the number of particles small, it is reasonable to ask whether the target's position should be estimated with a parametric filter such as the Extended Kalman Filter (EKF) or Unscented Kalman Filter (UKF). However, the EKF and UKF are only appropriate when the belief can be well approximated as a single Gaussian. This could be helpful once the team's estimate has converged, but at that point in time the particle filter should be able to represent the belief using a few hundred particles, making it fast for both estimation and control. Using a filter like the EKF also has its drawbacks. Hoffmann and Tomlin [52] describe how linearizing the measurement model introduces error into the calculation of mutual information, potentially hurting the team's overall performance.

3.1.3 Control

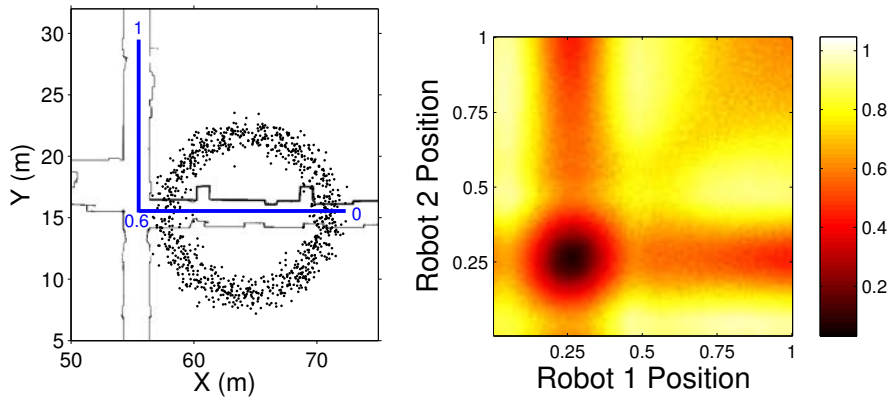
Our control strategy is designed to drive the team so that they obtain measurements which lead to a reduction in the uncertainty of the target estimate. The mutual information between the current belief of the target's state and expected future measurements captures this intuitive notion. The advantage of this approach is that it incorporates the current belief of the target's state along with the measurement model to determine how potential future measurements will impact the state. By design the team will move in directions where their combined measurements will be useful. This is particularly important when using sensors which provide limited information about the state of the target.

To motivate the choice of mutual information as an objective function for range-only localization, we analyze its behavior in two typical scenarios. Fig. 3.3a shows a belief distribution made up of two distinct hypotheses, a distribution that often arises when using range-only measurements. We consider the mutual information gained by two robots when their position is restricted to the blue line. Each robot's position is parameterized by a number in the interval $[0, 1]$. Intuitively, if both robots are within $[0, 0.6]$ then their estimate will not improve because any range measurement they make will support both hypotheses. Alternatively, measurements made in the interval $[0.6, 1.0]$ will eliminate one of the hypotheses. Fig. 3.3b shows that mutual information captures this intuition. It is close to its lower bound of 0 when the robots are within $[0, 0.6]$ and much higher once they



(a) Two hypotheses

(b) Mutual information; two hypotheses



(c) Circular belief

(d) Mutual information; circular belief

Figure 3.3: Mutual information cost surface. (a) and (c) show two separate belief distributions as well as a line parameterized from $[0, 1]$ where two robots can move. (b) and (d) show the value of mutual information as a function of the position of the robots on the line. Overall, mutual information is higher in places where robots will make useful measurements.

leave that area. Fig. 3.3c and Fig. 3.3d show similar results when the belief is circular. In this case, mutual information agrees with the intuition that as robots move away from the center of the belief – 0.3 on the line – they will make more useful measurements.

Accordingly, we formalize our control law as follows. We select the next configuration for the robotic team using the objective function:

$$\mathbf{c}_{t+1} = \arg\max_{\mathbf{c}\in\mathcal{C}} \ \mathrm{I}_{\mathrm{MI}}[\mathbf{x}_t; \mathbf{z} \mid \mathbf{c}] = \arg\max_{\mathbf{c}\in\mathcal{C}} \ \mathrm{H}[\mathbf{z} \mid \mathbf{c}] - \mathrm{H}[\mathbf{z} \mid \mathbf{x}_t, \mathbf{c}] \tag{3.3}$$

The domain of the optimization problem is the configuration space of the team, $\mathcal{C}$, and

all other terms are defined in Ch. 2 (see (2.6)). Here we are concerned with the mutual information between random vectors (i.e., the target state and measurements) as opposed to random variables. Also, note that the team's configuration $\mathbf{c}$ has a concrete value, while the target's state $\mathbf{x}$ is uncertain.

Determining Locations

Typical indoor environments are non-convex, meaning we must maximize mutual information over a non-convex set. To do this, we propose searching over a discrete set of configurations of the team. In our experiments, we do this by creating a connectivity graph along with an embedding into the environment. At each time step, robots find candidate locations by performing breadth first search and taking nodes within specified range intervals. Short ranges allow a robot to continue gathering useful measurements where it is, while long ranges enable it to explore new locations.

To create the connectivity graph we perform a Delaunay triangulation of the environment and define the incenters of the triangles as nodes. This requires a polygonal representation of the obstacles which can be created from an occupancy grid map. For edges, we connect nodes from adjacent triangles as well as their transitive closure. Fig. 3.5 shows two examples of this approach, which we used in our experiments.

Calculating The Objective

Hoffmann and Tomlin [52] developed an approach for calculating mutual information with particle filters that we use here. In particular, they showed that by using the particle filter's discrete approximation of the belief, (3.2), the entropies can be expressed as:

$$\mathrm{H}[\mathbf{z}] \approx -\int_{\mathbf{z}} \left(\sum_i w_i \; p(\mathbf{z} \mid \mathbf{x} = \tilde{\mathbf{x}}_i) \right) \log \left(\sum_i w_i \; p(\mathbf{z} \mid \mathbf{x} = \tilde{\mathbf{x}}_i) \right) d\mathbf{z} \tag{3.4}$$

$$\mathrm{H}[\mathbf{z} \mid \mathbf{x}] \approx -\int_{\mathbf{z}} \sum_i w_i \; p(\mathbf{z} \mid \mathbf{x} = \tilde{\mathbf{x}}_i) \log p(\mathbf{z} \mid \mathbf{x} = \tilde{\mathbf{x}}_i) d\mathbf{z} \tag{3.5}$$

Where we have dropped the measurement's dependence on the configuration of the team for brevity. The approximate equalities are due to the particle filter's approximation of the

belief.

It is necessary to predict each robot's LOS conditions to select a measurement model when calculating the entropies. In contrast to estimation, the geometric method's predictions are more accurate than the HMM's. This is because the HMM treats LOS conditions as a Bernoulli distribution that evolves over time according to a doubly stochastic matrix that is independent of the environment. Consequently, it predicts the same LOS conditions for all future measurements regardless of where the team goes. The geometric method does not have this issue, which is why we use it for prediction.

In our work, we exploit the assumed conditional independence of the measurements given the state when calculating the conditional entropy. By exchanging summation and integration, we can rewrite (3.5) as

$$\mathrm{H}[\mathbf{z} \mid \mathbf{x}] \approx \sum_i w_i \sum_j \mathrm{H}[z^j \mid \mathbf{x} = \tilde{\mathbf{x}}_i]$$

This reduces the conditional entropy to be multiple *separate* integrals over a single measurement space – which can often be done analytically – as opposed to one integral over the joint space of all measurements the team makes.

The entropy of the measurement distribution (3.4) is harder to compute. However, the particle filter transforms the distribution into a finite dimensional mixture model, which enables new approximation techniques.

Approximating Mutual Information

Unfortunately, performing the mutual information calculations in real time for teams with more than 3 or 4 robots is computationally impractical: evaluating the mutual information of a single configuration requires numerically integrating over the full measurement space to calculate the measurement entropy. Rather than performing numerical integration over subsets of the space [52], we view the measurement distribution $p(\mathbf{z})$ as a mixture model. The i^{th} component is $p(\mathbf{z} \mid \mathbf{x} = \tilde{\mathbf{x}}_i, \mathbf{c})$ with weight w_i, both of which are determined by the i^{th} particle. Because our measurement model is Gaussian, we can use a deterministic

approximation algorithm for evaluating the entropy of Gaussian mixture models [59].

The algorithm is based on a Taylor series expansion of the logarithmic term in the integral. This replaces the log term by a sum. Exchanging the order of integration and summation, the entropy can be expressed as a weighted sum of the Gaussians' central moments. The integrals can be calculated analytically, and only the weighting terms need to be computed online.

We only use the 0^{th}-order term in the Taylor series expansion:

$$\mathrm{H}[\mathbf{z}] \approx -\int_{\mathbf{z}} \sum_k w_k \mathcal{N}(\mathbf{z}; \mu_k, \Sigma_k) \log g(\mu_k)\, d\mathbf{z} = -\sum_k w_k \log g(\mu_k)$$

$g(\mu_k)$ is the likelihood of the mixture model evaluated at the mean of the k^{th} component. The computational complexity of this approximation is $\Theta(n^2 l)$ where n is the number of particles and l is the number of robots. The time is linear in l because the conditional independence assumption in the measurement model results in the covariance matrix of the measurements being diagonal. Table 3.3 summarizes the computational complexity of the entire approach.

This approach also works when the measurement model is itself a mixture of Gaussians. However, this would result in one mixture component for every separate combination of mixture components from all robots, leading to exponential growth. This is partly why we use a Gaussian measurement model with distance dependent variance rather than a Gaussian mixture model.

3.2 Experiments

3.2.1 Experimental Design

There are four primary questions that we seek to answer with our experiments: 1) does our measurement model result in an accurate estimate of the target's location? 2) does classifying measurements with an HMM do better than the geometric approach? 3) does our approximation of mutual information result in trajectories that reduce the uncertainty

of the team's estimate? 4) are our results repeatable across different environments and with various starting conditions?

We design five separate experiments to answer these questions. All of the experiments have two robots trying to locate a third stationary robot. To assess repeatability, we run several independent trials of each experiment. For each trial, we let the robots explore the environment without interacting with them, and only stop them once the filter reaches a stable estimate.

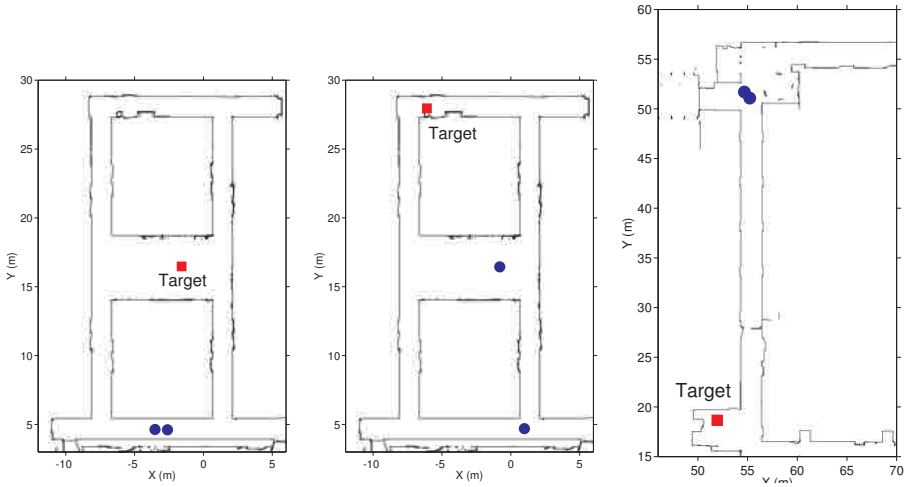
To evaluate the performance of each trial, we calculate the empirical mean of the filter's distribution as well as the volume of its covariance matrix (i.e., its determinant). These statistics are not always an accurate reflection of the filter's performance (e.g., the average of two distinct hypotheses may be far from either hypothesis), but will show whether the filter accurately converges to a single estimate over time.

We use a qualitative approach to evaluate the trajectories of the robots, and manually assess whether or not they are reasonable. As a baseline comparison, in open environments with no prior knowledge of the target's location, it is best to move two range sensors orthogonally to one another as is seen in other approaches [49, 132, 95].

In Experiment 1, we place two robots within 0.5m of each other and put the target in NLOS conditions approximately 16m away. A good control strategy for this experiment will result in the robots moving in complementary directions rather than staying tightly clustered together.

For Experiment 2, we place the robots far away from each other to see if they still move in complementary directions. The separation also tests whether the measurement model consistently combines measurements from different modalities; if the robots follow good trajectories, they will achieve LOS to the target at different times.

In Experiment 3, we place the robots within 0.5m of each other with the target more than 30m away. Experiment 3 also takes place in a different environment. Experiments 1 and 2 take place in Levine Hall at the University of Pennsylvania, which was constructed primarily in 2003 with modern constructions materials – its walls are primarily made up of



(a) Expt. 1 (Levine) (b) Expt. 2 (Levine) (c) Expt. 3 (Towne)

(d) Expt. 4 (Towne) (e) Expt. 5 (Skirkanich)

Figure 3.4: Starting configurations of robots. Blue dots show the starting location of the two mobile robots and the red square shows the location of the target.

wood or metal framing with drywall. Experiments 3 and 4 take place in Towne Building which was built in 1906 – its walls are typically made of brick or concrete.

For Experiment 4, we again place the robots close to each other with the target 20m along a hallway that has a slight bend. The robots and the target do not have geometric LOS, but the wall between them just barely obstructs their view, making it ambiguous whether or not the RF signal will exhibit LOS or NLOS behavior.

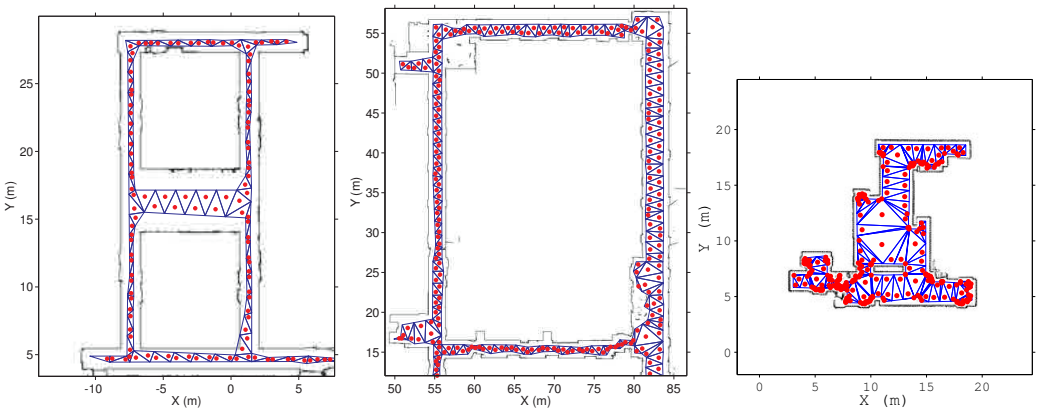
In Experiment 5, we examine how the robots move when they are in a more open environment. The robots start next to each other and have more control options to select from than in the first four experiments since they are not constrained to move along a hallway. This experiment takes place in Skirkanich Hall which was completed in 2006 and also features modern construction materials.

Fig. 3.4 shows the starting location of the target and robots for each experiment.

3.2.2 Equipment and Configuration

We use the 3rd generation of the Scarab mobile robot Sect. A.1. The Scarabs are simple differential drive robots equipped with laser scanners for localization and 802.11s wireless mesh cards for communication [82]. The experimental software is developed in C++ and interfaced via Robot Operating System [106]. The range sensor is commercially available as part of the nanoPAN 5375 Development Kit [86]. The target remains stationary for the duration of each trial while the Scarabs are limited to a maximum speed of 0.2 m/s. The particle filter and mutual information calculations are performed on a laptop with 4GB RAM and an Intel Core Duo processor. To prevent inter-robot collisions, we use the Optimal Reciprocal Collision Avoidance algorithm provided in the RVO2 library [122, 109].

For localization, planning, and determining LOS conditions we use a known occupancy grid map of the environment with a resolution of 0.05 m. The results of the triangulation algorithm in Sect. 3.1.3 are shown in Fig. 3.5. Robots consider nodes within $1.0 - 2.0$ m or $8.0 - 9.0$ m of their current location for the control update, which typically result in 7-12 locations per robot. We chose these distance cutoffs because they generate few enough options to compute everything in real time, while still enabling the team to evaluate several



(a) Levine (b) Towne (c) Skirkanich

Figure 3.5: Graphs for candidate locations. Delaunay triangles are shown along with their incenters which form the vertices of the graph.

Table 3.2: Measurement Model Parameters.

	LOS	NLOS
α_0, β_0	2.51	-1.34
α, β	0.20	-0.24
σ_L^2, σ_N^2	7.00	14.01

distinct choices. In Ch. 4 we develop a theoretical justification for these parameters by proving that mutual information cannot change substantially when the robot's position change doesn't change substantially more than the standard deviation of individual range measurements.

We uniformly sample 2500 locations throughout the environment to initialize the particle filter and run a low variance resampler when the proportion of effective particles drops below 0.3. We calculate the MLE of the measurement model's parameters, (3.1), using a separate dataset gathered in Levine. During the experiments, we double the MLE variances to prevent the filter from prematurely converging without affecting the location of the convergence. Specifically, if $\sigma_{L,\text{MLE}}$ and $\sigma_{N,\text{MLE}}$ are the MLE values of the variance for LOS and NLOS conditions, we set $\sigma_L = 2\sigma_{L,\text{MLE}}$ for LOS and $\sigma_N = 2\sigma_{N,\text{MLE}}$ for NLOS. Parameter values for the experiments are listed in Table 3.2. We emphasize that we used the same parameters for the Towne, Levine, and Skirkanich experiments.

Table 3.3: Computational complexity of control law. (3.3) with n particles, l robots, and d potential locations per robot.

Task	Cost
Single Evaluation	$\Theta(n^2 l)$
Solving Objective	$\Theta(n^2 l d^l)$

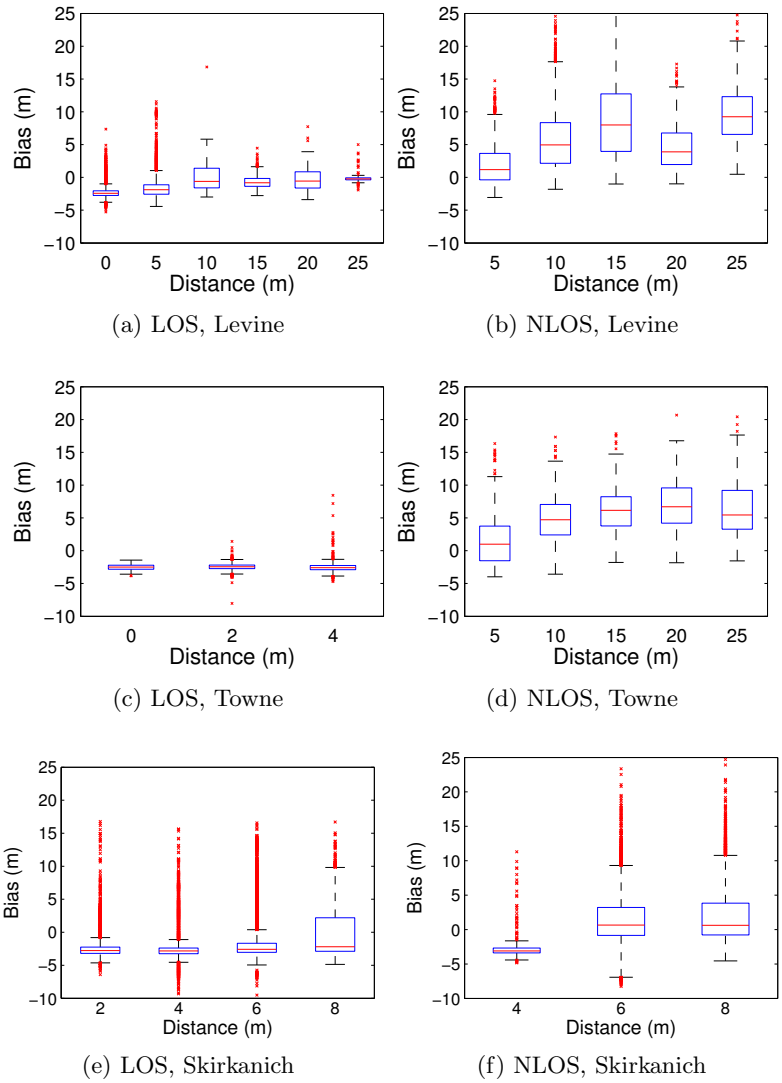
3.2.3 Results

Fig. 3.6 shows the error of the nanoPAN 5375's measurements as a function of distance. At short distances in LOS conditions the nanoPAN provides consistent measurements with an error larger than 2.0m. In NLOS conditions there is significant variability in both the mean and variance of the measurements as the distance between source and receiver increases. This data helps justify our choice of measurement model as it is clear that LOS and NLOS conditions have different sensor measurement biases and variances.

Overall, our approximation of mutual information resulted in good trajectories for the robots. Fig. 3.7 shows that the robots moved to gain complementary measurements; they moved orthogonally to one another when possible, and once they had localized the source, they maintained LOS. Fig. 3.8 shows this process in more detail. Initially, both robots moved away from each other. Next, the robot on the left moved up the vertical hallway, while the other robot moved laterally; an orthogonal movement pattern. Once measurements along the horizontal hallway were no longer helpful due to the symmetry of the distribution, Fig. 3.8b, both robots moved up the vertical hallway, which caused the filter to converge. We stress that these behaviors arose organically from our objective.

Fig. 3.8 also serves as a good example of the types of distributions that typical parametric approaches that assume a unimodal distribution cannot reliably track as there are clearly multiple equally valid hypotheses.

Figures 3.9 and 3.10 show the mean and covariance of the estimate over time across all trials and experiments when using an HMM to classify measurements. Across all experiments, the mean converged to the true state of the target, ultimately surpassing the



(a) LOS, Levine (b) NLOS, Levine

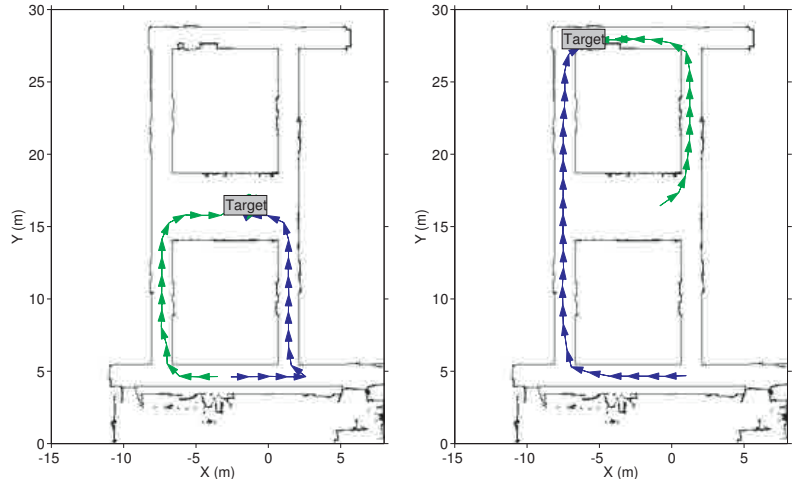
(c) LOS, Towne (d) NLOS, Towne

(e) LOS, Skirkanich (f) NLOS, Skirkanich

Figure 3.6: How geometric LOS and NLOS affects nanoPAN range measurements. The subfigures show standard box-plots with outliers marked as ×'s. Each plot contains approximately 10,000 data points. NLOS conditions are significantly noisier than LOS conditions with statistics that vary with distance.

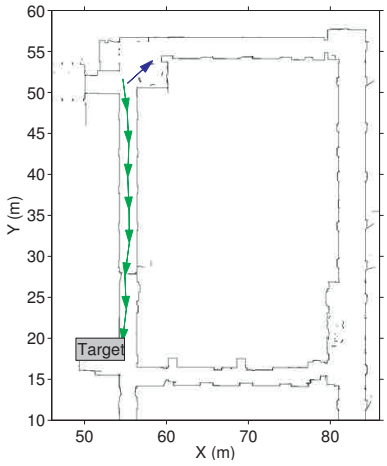
Table 3.4: Mean and std. dev. of RMSE of filter's error over last 30 seconds for each experiment. The error is lower when using an HMM, particularly in Expt. 4.

	Expt. 1	Expt. 2	Expt. 3	Expt. 4	Expt. 5
Geometric	0.73 m (0.25)	0.94 m (0.33)	2.13 m (0.45)	**5.53** m **(0.09)**	2.14 m (0.94)
HMM	0.70 m (0.24)	0.72 m (0.23)	1.61 m (0.63)	**1.51** m **(0.35)**	1.11 m (0.40)

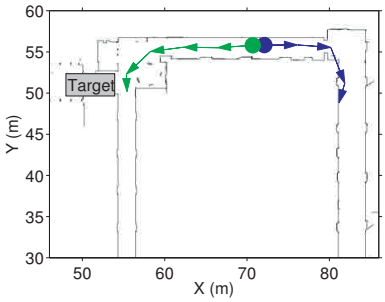


(a) Expt. 1 (Levine)

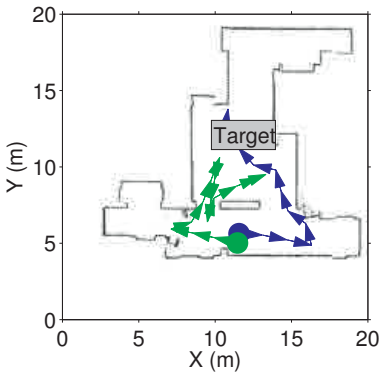
(b) Expt. 2 (Levine)



(c) Expt. 3 (Towne)

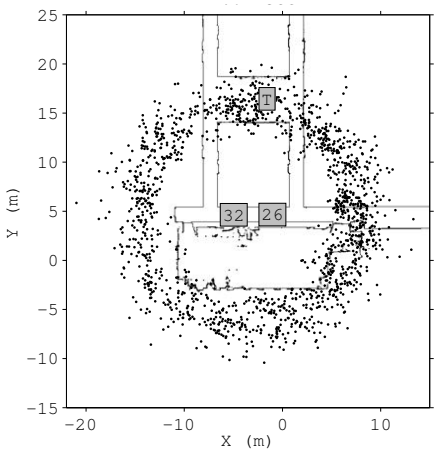


(d) Expt. 4 (Towne)

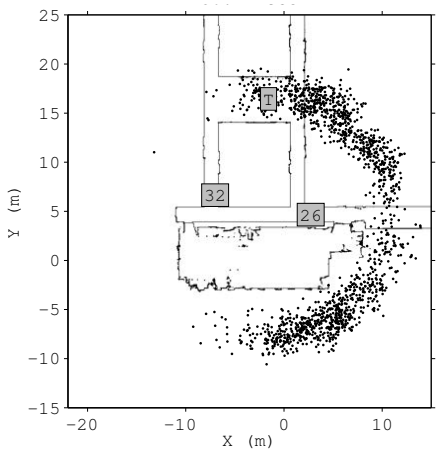


(e) Expt. 5 (Skirkanich)

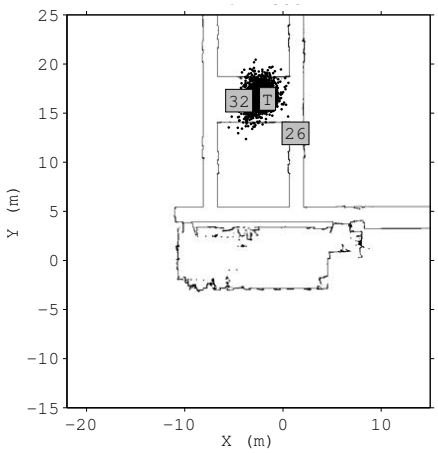
Figure 3.7: Trajectories from maximizing mutual information. Each subfigure is a typical trajectory from each experiment. The robots move to gather complementary measurements, causing the estimate of the target to converge.



(a) $t = 11.02$ s



(b) $t = 30.14$ s



(c) $t = 112.27$ s

Figure 3.8: Evolution of the particle filter and the robots' movement in Experiment 1. Particles are black dots, the target's position is shown by a box with a "T" and each robot's position is shown by a box with a number. a) Early range measurements cause a ring. b) Both robots move laterally, generating two hypotheses. c) The robots move up, and the filter converges.

baseline accuracy of an individual measurement. Table 3.4 provides the resulting root mean square error (RMSE) of the converged filter. The slightly increased error in Experiments 3, 4 and 5 suggests that the parameters learned from data collected in Levine do not perfectly transfer to Towne and Skirkanich. However, the overall error is still low. The fact that the covariance decreases and ends in the range $[-3, 0]$ on a natural logarithmic scale, shows that filter consistently converges to a single hypothesis. For reference, the natural logarithm of the determinant of the identity matrix (i.e., 1 meter variance in only x and y) has a value of 0. The consistent trends across all trials and experiments demonstrate that the fundamental approach is robust to various starting conditions and changes in the sensor across environments.

As shown in Tab. 3.4, the results for using the geometric LOS approach result in similar accuracies except in Experiment 4. For that experiment, we ran 3 separate trials and in each case the filter converged to a single estimate that was 5.7 m away from the target's true location. The essential problem is that at the start of the experiment the robots and the target do not have geometric LOS, but the RF interference is low. As Fig. 3.11a shows, the measurements exhibit negative bias and relatively low variance. Comparing these results to the HMM's classification, Fig. 3.11d it is clear that the HMM classifies measurements more accurately, enabling the filter to accurately correct for the bias. It is worth noting that the robots still followed a reasonable trajectory when using geometric LOS; the control did *not* fail.

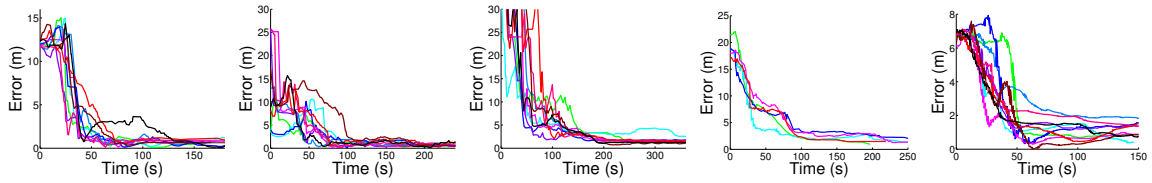
Fig. 3.11 shows two additional datasets from Levine which exhibit the same phenomena. Fig. 3.11b and Fig. 3.11e show measurement classifications from an experiment where a robot drives up and down a long hallway with a target at one end (Fig. 3.12a). There are no obstructions; the robot and target always have geometric LOS. Fig. 3.11c and Fig. 3.11f show classifications from another experiment where a robot drives along a short hallway with a target just around the corner (Fig. 3.12b). The robot only has geometric LOS to the target at one end of its path. As in Experiment 4, these results show that geometric LOS is not always equivalent to RF LOS; there are times when the bias and variance of

the measurements suggest one measurement regime, but the geometric method predicts another. Because the HMM uses the statistics of the measurements, it classifies them more accurately.

3.3 Conclusion

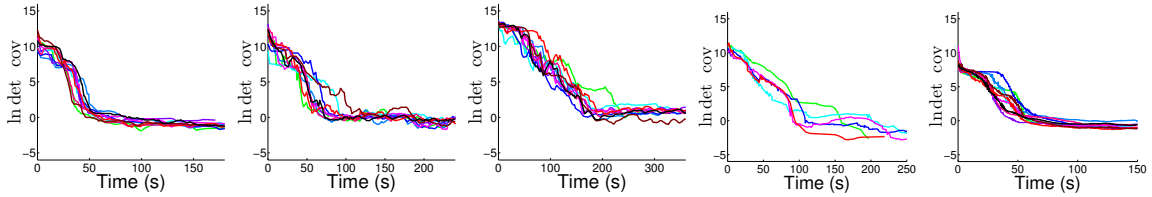
This chapter described an active control strategy which leverages the current estimate of the target's state and knowledge of the sensor model to direct a team of robots towards locations where they will make informative measurements. We showed how approximations for the entropy of Gaussian mixture models can be used to calculate the necessary control inputs in real time. Most importantly, we presented extensive experimental results demonstrating that 1) our measurement model results in accurate estimates of the target's location, 2) LOS and NLOS measurements differ for typical RF range sensors, and that classifying measurements with a Hidden Markov Model outperforms a simple geometric approach, 3) our approximation of mutual information results in the team following intelligent trajectories and 4) our results are repeatable across different environments with various starting conditions.

Our methods for approximating mutual information work well, but still present significant computational challenges. In Ch. 4 we show how approximating the belief can significantly speed up the calculation of mutual information, while introducing limited error. In Ch. 5 we address the exponential complexity inherent in planning actions for multiple robots.



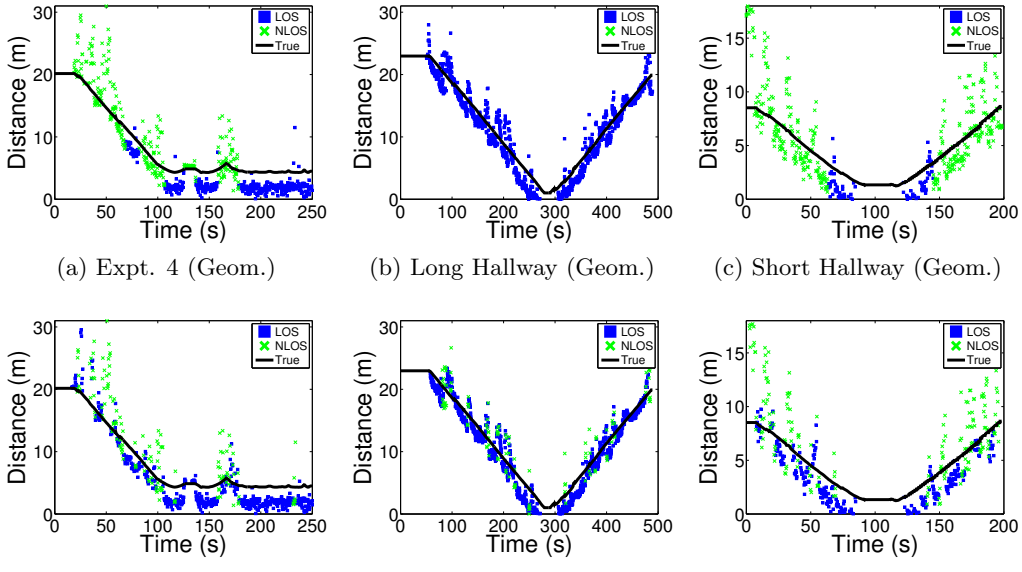
(a) Expt. 1 (b) Expt. 2 (c) Expt. 3 (d) Expt. 4 (e) Expt. 5

Figure 3.9: Distance from weighted average of particles to target location using an HMM. Experiments 1-3 and 5 have 10 trials, while Experiment 4 has 5. Overall, the filter converges to an accurate estimate.



(a) Expt. 1 (b) Expt. 2 (c) Expt. 3 (d) Expt. 4 (e) Expt. 5

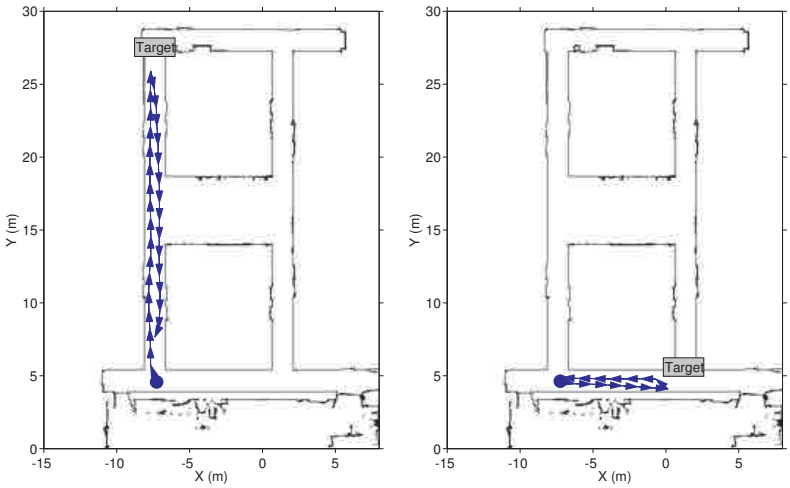
Figure 3.10: Logarithm of the determinant of covariance when using an HMM. The covariance decreases, indicating the filter converges to a single hypothesis.



(a) Expt. 4 (Geom.) (b) Long Hallway (Geom.) (c) Short Hallway (Geom.)

(d) Expt. 4 (HMM) (e) Long Hallway (HMM) (f) Short Hallway (HMM)

Figure 3.11: HMM vs. Geometric. (a)-(c) labels from geometric LOS. (d)-(e) labels from an HMM. The HMM more accurately classifies measurements, enabling the filter to accurately correct for the different biases.



(a) Long Hallway (b) Short Hallway

Figure 3.12: Long Hallway and Short Hallway paths. (a) The robot and target always have geometric LOS. (b) The robot and target only have geometric LOS at one end of the path.

Chapter 4

Approximate Representations for Maximizing Mutual Information

In this chapter, we address the problem of controlling a small team of robots to estimate the location of a mobile target using range-only sensors. While the mutual information based control law in Ch. 3 is a good starting point, it has two problems that preclude it from solving this problem. The first problem is that the control law is computationally expensive due to the complexity of the particle filter representation. If the team cannot compute control actions quickly in this scenario, the mobile target will move substantially, making it difficult to track. The second issue is that the control law only considers mutual information at a single point for each team member. This means that the team does not consider the impact of multiple measurements over time, making it more difficult to select appropriate control actions. Calculating mutual information over a finite time horizon is difficult, as it requires integration over the joint measurement space of the team, which grows with the time horizon of the plan and size of the team.

To address these difficulties we develop a new approximation and theoretical design criterion which enables us to build a real-time mutual information controller for our mobile target tracking problem. A primary contribution in this chapter is a method to approximate mutual information by approximating the distribution over the target's predicted location.

While this approximation necessarily introduces error into the control law, we show that this error is bounded for Gaussian measurement models. Importantly, this bound yields insight into the degree to which we can speed up the calculation of mutual information via approximation without significantly affecting the team's performance. The other primary contribution is a bound on the difference in mutual information between two control inputs as a function of the noise of the range sensors. This analysis further aids in the design of a real-time system by enabling us to appropriately design the set of control inputs.

We evaluate the proposed methodology and design trade-offs through simulations and real world experiments. Our results show that the approximation significantly decreases the time to compute mutual information and enables the team to quickly and successfully estimate a mobile target's position.

This chapter continues to focus on range-only RF-based sensors. However, as the proposed mutual information based techniques are broadly applicable, the approximations in Sect. 4.2 remain general.

This chapter originally appeared in [15, 16].

4.1 Mutual Information Based Control

In this section we describe a mutual information based control law for the mobile target tracking problem. The approach and discussion are similar to Sect. 3.1 except that here 1) the filter estimates a moving target, 2) the control law considers the impact of multiple measurements made by the team, and 3) the discrete action set is generated using path planners as opposed to a Delaunay triangle decomposition of the environment. As before, the team shares observations to jointly estimate the target's location (Sect. 4.1.1). By predicting where the target will go next and what measurements they are likely to receive (Sect. 4.1.2), the team chooses control directions that maximize the predicted reduction in entropy of the target's uncertainty over a finite time horizon (Sect. 4.1.3).

4.1.1 Target Estimation

To estimate the target's position, we use a particle filter, and the same basic representation described previously (Sect. 3.1.2). However, we now also need to account for the target's mobility. Similar to other work on predicting the motion of an un-cooperative target [110, 138] we use a Gaussian random walk for the process model: $p(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \mathbf{x}_{t-1}, \sigma_p^2 I)$ where I is the identity matrix and $\mathcal{N}(\mathbf{x}; \mu, \Sigma)$ is a Gaussian with mean μ and covariance Σ.

Throughout this chapter, we also make the simplifying assumption that the measurement model, $p(\mathbf{z}_t \mid \mathbf{x}_t)$, can be modeled as the true distance between the target and measuring robot perturbed by Gaussian noise. As discussed in Sect. 3.1.1 this assumption is not always valid for range sensors, but this simpler model enables us to analyze the effect of various approximations.

4.1.2 Target and Measurement Prediction

To determine how they should move, the team must predict where the target will go and what measurements they will receive. Specifically, if the team plans an action over the time interval $\boldsymbol{\tau} \stackrel{\triangle}{=} t+1 \,:\, t+T$, they need to estimate the target's future state: $\mathbf{x}_{\boldsymbol{\tau}} = [\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+T}]$. This can be done using the current belief and process model:

$$p(\mathbf{x}_{\boldsymbol{\tau}} \mid \mathbf{z}_{1:t}) = \int p(\mathbf{x}_t \mid \mathbf{z}_{1:t}) \prod_{i=t+1}^{t+T} p(\mathbf{x}_i \mid \mathbf{x}_{i-1})\, d\mathbf{x}_t \approx \sum_{j=1}^{M} w_j \delta(\mathbf{x}_{\boldsymbol{\tau}} - \tilde{\mathbf{x}}_{\boldsymbol{\tau}}^j) \tag{4.1}$$

Where $\delta(\cdot)$ is the Dirac delta function, $\tilde{\mathbf{x}}_{\boldsymbol{\tau}}^j$ is the trajectory of the j^{th} particle, and w_j is its weight.

Over this time interval, the team will execute an action, which we define as a discrete sequence of poses for each team member. An action can be expressed as $\mathbf{c}_{\boldsymbol{\tau}} = [\mathbf{c}_{t+1}, \ldots, \mathbf{c}_{t+T}]$ where $\mathbf{c}_i^r$ is the 2D position of robot r at time i (we ignore orientation as it does not affect range measurements). As the team executes it action it will receive range measurements $\mathbf{z}_{\boldsymbol{\tau}} = [\mathbf{z}_{t+1}, \ldots, \mathbf{z}_{t+T}]$. For a team of R robots, $\mathbf{z}_i = [z_i^1, \ldots, z_i^R]$ is the vector of 1D measurements the team receives at time i. The measurement distribution can be calculated by

marginalizing over the state and applying the conditional independence assumption of the measurements:

$$p(\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{z}_{1:t}, \mathbf{c}_{\tau}) = \int p(\mathbf{x}_{\boldsymbol{\tau}} \mid \mathbf{z}_{1:t}) \prod_{i=t+1}^{t+T} p(\mathbf{z}_i \mid \mathbf{x}_i, \mathbf{c}_i)\, d\mathbf{x}_{\boldsymbol{\tau}} \tag{4.2}$$

Because the measurement model is Gaussian, substituting (4.1) into (4.2) results in a Gaussian mixture model representation of the measurement density. After applying the conditional independence assumption, the j^{th} mixture component has a weight of w_j with a distribution equal to $\prod_{i=t+1}^{t+T} \prod_{r=1}^{R} p(z_i^r \mid \mathbf{x}_i = \tilde{\mathbf{x}}_i^j, \mathbf{c}_i^r)$. Individual range measurements are 1-dimensional, which means the measurement space is RT-dimensional.

4.1.3 Information Based Control

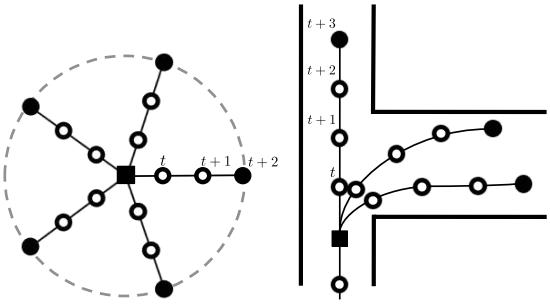
Control Law

As in Ch. 3, our control law maximizes the mutual information between the future state of the target and future measurements made by the team. The primary difference is that it now considers the impact of multiple measurements over time, selecting a better action than a greedy maximization would. Formally, at time t the team plans how it will move over the time interval $\boldsymbol{\tau}$ by considering a set of actions, $\mathcal{C}$, that they can follow. After generating $\mathcal{C}$, the team selects the action that maximizes mutual information:

$$\mathbf{c}_{\boldsymbol{\tau}}^* = \arg\max_{\mathbf{c}_{\boldsymbol{\tau}} \in \mathcal{C}} \mathrm{I}_{\mathrm{MI}}[\mathbf{z}_{\boldsymbol{\tau}}; \mathbf{x}_{\boldsymbol{\tau}} \mid \mathbf{c}_{\boldsymbol{\tau}}] = \arg\max_{\mathbf{c}_{\boldsymbol{\tau}} \in \mathcal{C}} \mathrm{H}[\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{c}_{\boldsymbol{\tau}}] - \mathrm{H}[\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}, \mathbf{c}_{\boldsymbol{\tau}}] \tag{4.3}$$

Although it is not explicit, both entropies are conditioned on measurements made up to time t.

To generate $\mathcal{C}$, we use motion primitives generated by path planners. For example, in an open environment the set of actions could be composed of straight line paths whose final destinations are uniformly spaced on a circle (Fig. 4.1a). When non-convex obstacles are present (e.g., in an office building), we generate $\mathcal{C}$ by generating paths to all points that are a user specified distance away using Dijkstra's algorithm. Each endpoint has a single



(a) Points on a Circle (b) Indoor Planner

Figure 4.1: Representative actions for a single robot using a finite set of motion primitives with a finite horizon. The black boxes are the starting locations, the black circles are the final destinations, and the hollow circles show intermediate points.

Figure 4.2: Representative actions generated by path planning. Blue lines show shortest paths from the red circle to nearby locations using an occupancy grid.

shortest path which defines an action. To determine the points at which a robot makes measurements, the path is discretized into a fixed number of waypoints (Fig. 4.1b).

We generate motions by planning paths instead of using the connectivity graph approach described in Sect. 3.1.3 for two reasons. First, paths can be planned using an occupancy grid, while the previous approach required creating a polygonal representation of the environment. Second, paths planned in a occupancy grid with a reasonable resolution are smoother than those between incenters of Delaunay triangles (e.g., compare Fig. 4.2 to the zig-zag paths that would be planned in Fig. 3.5).

Calculating the Objective

To evaluate the objective for an action $\mathbf{c}_{\boldsymbol{\tau}}$, we separately evaluate the conditional measurement entropy and measurement entropy.

Calculating the conditional entropy is straightforward. Using its definition and the assumed conditional independence of the measurements given the state:

$$\mathrm{H}[\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}, \mathbf{c}_{\boldsymbol{\tau}}] \approx \sum_{i=1}^{M} w_i \sum_{j=t+1}^{t+T} \sum_{r=1}^{R} \mathrm{H}[z_j^r \mid \mathbf{x}_j = \tilde{\mathbf{x}}_j^i, \mathbf{c}_j^r] \tag{4.4}$$

M is the number of particles used to predict the target's future state (4.1) and the approximate equality is due to the particle representation. Because each measurement comes from a Gaussian distribution, the entropies in the summand can be calculated analytically, making the running time $\Theta(MRT)$.

Unfortunately, as shown in Sect. 4.1.2 the measurement density, $p(\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{z}_{1:t}, \mathbf{c}_{\boldsymbol{\tau}})$, is a Gaussian mixture model (GMM) and there is no analytic method for calculating its entropy. Numerical integration methods can be used, but the domain of integration is RT-dimensional, making it large even when there are a few robots over short time scales. This necessitates the use of algorithms like Monte Carlo integration which scale exponentially with the dimension, making them unsuitable for real-time performance.

However, there is a deterministic approximation for the entropy of a GMM developed by Huber et al. [59]. This algorithm uses a truncated Taylor series expansion of the logarithmic term in the entropy which allows the integration to be performed analytically. We use the 0^{th} order approximation:

$$\mathrm{H}[g] \approx -\sum_{k=1}^{M} w_k \log \sum_{j=1}^{M} w_j \mathcal{N}(\mu_k; \mu_j, \Sigma_j) \tag{4.5}$$

where $g(\mathbf{x}) = \sum_{i=1}^{M} w_i \mathcal{N}(\mathbf{x}; \mu_i, \Sigma_i)$. While this approximation is not tight, we show experimentally that it suffices for information based control.

The time complexity of (4.5) is $\Theta(M^2RT)$. The dependence on the number of particles which represent the target's future state, M, is problematic as it grows exponentially with the time horizon, T. While there are algorithms for reducing the number of components in a GMM, their time complexity is also quadratic in the number of components [108, 58]. In

Sect. 4.2.2 we describe a different approach which reduces the number of components by approximating the distribution over the target's future state and prove that such approximations result in similar distributions over measurements.

4.2 Approximate Representations

In Sect. 4.1 we outlined a general control law for a multi-robot team. However, it is computationally infeasible as the team's size and time horizon grow. It is also unclear which actions to consider. We address these problems in Sect. 4.2.2 where we analyze how an approximation of the target's future state affects the objective function, and in Sect. 4.2.3 where we develop a theoretical design criterion for generating actions. These analyses use an information theoretic relationship that we prove in Sect. 4.2.1.

4.2.1 Bounding Entropy Difference

Intuitively, if two Gaussian mixture models (GMMs) are similar, then their entropies should be similar as well. In particular, one would expect that if the means of individual components of a GMM are slightly perturbed, then the entropy of the GMM would not change substantially. Our primary theoretical contribution is proving that this intuition is correct. We do this by building on a previously known result that relates the L_1 distance between discrete distributions and their difference in entropies. For clarity of presentation, all proofs are deferred to the Appendix.

The L_1 distance is one way of measuring the difference between distributions. For two densities f and g, it is defined as $\|f - g\|_1 = \int |f(x) - g(x)|\, dx$. To concisely state the following results we do not use boldface to denote vectors, define $\phi(x) = \mathcal{N}(x; 0, 1)$ as the standard 1-dimensional Gaussian, $\Phi(x)$ as its cumulative distribution function, $\|x\|_\Sigma = \sqrt{x^T \Sigma^{-1} x}$ as the Mahalanobis distance, and $|\cdot|$ as the determinant of a matrix.

Theorem 1 (L_1 bound on entropy)**.** *Let f and g be probability density functions such that $\|f - g\|_1 \leq 1/2$, $0 \leq f(x) \leq 1$ and $0 \leq g(x) \leq 1$. Then*

$$|\,\mathrm{H}[f] - \mathrm{H}[g]\,| \leq -\|f - g\|_1 \log \|f - g\|_1 + \|f - g\|_1\, \mathrm{H}\left[\frac{|f(x) - g(x)|}{\|f - g\|_1}\right] \tag{4.6}$$

Cover and Thomas [27, Theorem 17.3.3] prove this result for discrete distributions, but the proof extends to continuous distributions whose likelihood is between 0 and 1. Using the following lemmas, we use Theorem 1 to prove that the entropy of a GMM does not vary substantially if the means of its components are perturbed.

Lemma 1 (L_1 between Gaussians with same covariance)**.** *If $f(x) = \mathcal{N}(x; \mu_1, \Sigma)$ and $g(x) = \mathcal{N}(x; \mu_2, \Sigma)$ are k-dimensional Gaussians then*

$$\|f - g\|_1 = 2\left(\Phi\left(\delta\right) - \Phi\left(-\delta\right)\right) = 2\left(2\Phi\left(\delta\right) - 1\right) \tag{4.7}$$

where $\delta = \|\mu_1 - \mu_2\|_\Sigma / 2$.

Lemma 2 (Entropy bound of normalized difference of Gaussians)**.** *If $f(x) = \mathcal{N}(x; \mu_1, \Sigma)$ and $g(x) = \mathcal{N}(x; \mu_2, \Sigma)$ are k-dimensional Gaussians with $\mu_1 \neq \mu_2$ then*

$$\mathrm{H}\left[\frac{|f - g|}{\|f - g\|_1}\right] \leq \log \sqrt{(2\pi e)^k |\Sigma| \left(1 + \delta^2 + \frac{2\delta\phi(\delta)}{2\Phi(\delta) - 1}\right)} \tag{4.8}$$

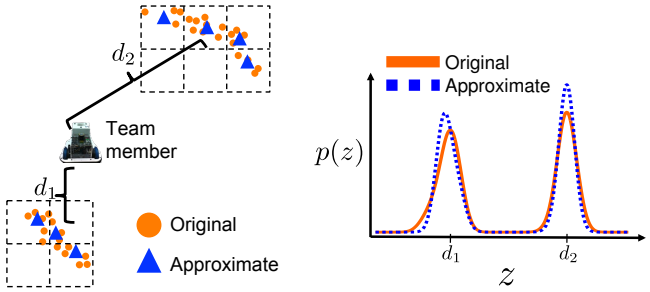
where $\delta = \|\mu_1 - \mu_2\|_\Sigma / 2$

Theorem 2 (Bound on Entropy Difference of GMMs)**.** *Let $f(x) = \sum_{i=1}^N w_i f_i(x)$ be a k-dimensional GMM where $f_i(x) = \mathcal{N}(x; \mu_i, \Sigma_i)$ and $|\Sigma_i| \geq (2\pi)^{-k}$ so $0 \leq f_i(x) \leq 1$. If $g(x) = \sum_{i=1}^N w_i g_i(x)$ where $g_i(x) = \mathcal{N}(x; \mu_i + \Delta_i, \Sigma_i)$ is the GMM created by perturbing the means of components of f by Δ_i such that $\|f_i - g_i\|_1 \leq 1/2$ for all i then:*

$$|\,\mathrm{H}\left[f\right] - \mathrm{H}\left[g\right]\,| \leq \sum_{i=1}^N -w_i \|f_i - g_i\|_1 \log \frac{w_i \|f_i - g_i\|_1}{\sqrt{(2\pi e)^k |\Sigma_i| \left(1 + \delta_i^2 + \frac{2\delta_i \phi(\delta_i)}{2\Phi(\delta_i) - 1}\right)}} \tag{4.9}$$

where $\delta_i = \|\Delta_i\|_{\Sigma_i} / 2$ and $\|f_i - g_i\|_1 = 2(2\Phi\left(\delta_i\right) - 1)$.

There are two primary limitations to Thm. 2, both of which come from Thm. 1. The first is that the bound only applies to GMMs where the determinant of the covariance of individual components is not too small; this is necessary to ensure that the maximum likelihood of each component is at most 1. This issue is not significant for our application as the covariance is determined by the noise of range measurements, which is typically quite



(a) Distributions over target locations

(b) Distributions over range measurements

Figure 4.3: Approximate measurement distribution.. (a) The original particle set (circles) can be approximated by a smaller set of particles (triangles) using a grid. (b) The different particle sets result in similar distributions over range measurements (e.g., peaks at d_1 and d_2)

large [17]. Another limitation is the L_1 constraint between components, which requires that each perturbation not be large relative to the covariance. This requirement can be relaxed by using a variant of Thm. 1. The L_1 constraint arises in the proof of Thm. 1 from an inequality of the entropy function, $h(t) = -t \log t$. Fannes [38] proved a similar inequality for $h(t)$ that is weaker, but does not have the L_1 constraint; this inequality can be substituted into the proof for Thm. 1 to eliminate the requirement that components be near each other.

There are other results that connect similarity metrics between distributions to the difference of their entropies. For example, Silva and Parada [118, Theorem 2] recently proved that in the limit as the KL divergence between bounded continuous distributions goes to 0, their entropies become equal. Research on rate distortion theory often looks at similar problems (e.g., how to find approximations of distributions without exceeding a certain distortion). However, we are not aware of prior work which bounds the entropy difference of continuous densities like GMMs whose support is unbounded.

4.2.2 Approximating Belief

As discussed in Sect. 4.1.3, the number of particles needed to predict the location of the target grows exponentially in time. This complicates calculating the entropy of the measurement distribution (4.2), as it depends on the predictive distribution of the target (4.1).

In this section, we show that a simple approximation of the predictive distribution can speed up calculations of the objective (4.3), without significantly affecting its value.

Our approach for approximating a particle set is to replace subsets of similar particles with their average. Specifically, we partition the state space with a regular square grid and create an approximate particle set by replacing particles in the same cell with their weighted average. Figure 4.3 shows an example of this process. In Fig. 4.3a the original particle set representing a multi-modal distribution over the target's location is reduced via a grid. The new set is smaller, but it slightly changes the measurement distribution as shown in Fig. 4.3b.

To analyze the error introduced by this approximation, we assume that the range based measurement model has a constant variance σ_m^2 and is unbiased. With these assumptions, the conditional entropy (4.4) is $\mathrm{H}[\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}] \approx (TR/2)\log(2\pi e\sigma_m^2)$, which doesn't depend on the number of particles or their location. Thus, the approximation does not introduce any error for this term.

The approximation does introduce error for the entropy of the measurement density, $\mathrm{H}[\mathbf{z}_{\boldsymbol{\tau}}]$. We use Thm. 2 to bound this error, by showing that the approximate measurement distribution is a perturbed version of the original measurement distribution. Let $\mathcal{X}$ be the original set of particles and $f(\mathbf{z}) = \sum_{j=1}^{M} w_j \mathcal{N}(\mathbf{z}; \mu^{f_j}, \sigma_m^2 I)$ be the Gaussian mixture model (GMM) it creates with the action $\mathbf{c}_{\boldsymbol{\tau}}$ and measurement model, using superscripts to index components. Let $\mathcal{G}(j)$ be a map from the index of a particle in $\mathcal{X}$ to the index of the corresponding particle in the approximate particle set $\mathcal{Y}$. The density $\mathcal{Y}$ creates is $g(\mathbf{z}) = \sum_{i=1}^{N} \pi_i \mathcal{N}(\mathbf{z}; \mu^{g_i}, \sigma_m^2 I)$, where the i^{th} component's mean and weight is determined by the particles in $\mathcal{X}$ that map to the i^{th} particle in $\mathcal{Y}$. While $f(\mathbf{z})$ and $g(\mathbf{z})$ may have a different number of components, $M \geq N$, the expressions are related in that each component in $f(\mathbf{z})$ corresponds to a single component in $g(\mathbf{z})$. This means the approximated measurement distribution can be re-expressed as $g(\mathbf{z}) = \sum_{j=1}^{M} w_j \mathcal{N}(\mathbf{z}; \mu^{f_j} + \Delta^j, \sigma_m^2 I)$ where Δ^j is the difference between μ^{f_j} and $\mu^{g_{\mathcal{G}(j)}}$. Each Δ^j can be bounded. Note that measurements are unbiased. If $\mathbf{c}_k^r$ is robot r's position at time k and $\tilde{\mathbf{x}}_k^{\mathcal{X}_j}$ is the position of the j^{th} particle in

$\mathcal{X}$ at time k, the corresponding element of μ^{f_j} is $\mu^{f_j}_{k,r} = \|\mathbf{c}^r_k - \tilde{\mathbf{x}}^{\mathcal{X}_j}_k\|_2$. Similarly, if $\tilde{\mathbf{x}}^{\mathcal{Y}_i}_k$ is the particle in $\mathcal{Y}$ that lies in the same cell as $\tilde{\mathbf{x}}^{\mathcal{X}_j}_k$, then $\mu^{g_i}_{k,r} = \|\mathbf{c}^r_k - \tilde{\mathbf{x}}^{\mathcal{Y}_i}_k\|_2$. Because $\tilde{\mathbf{x}}^{\mathcal{X}_j}_k$ and $\tilde{\mathbf{x}}^{\mathcal{Y}_i}_k$ lie in the same cell the farthest they can be apart is $\|\tilde{\mathbf{x}}^{\mathcal{X}_j}_k - \tilde{\mathbf{x}}^{\mathcal{Y}_i}_k\|_2 \leq \sqrt{2}L$ where L is the cell's length. Thus, each component of the perturbation is at most $\sqrt{2}L$. As the perturbation vector is RT dimensional for all j:

$$\|\Delta^j\|_{\sigma^2_m I} \leq \frac{\sqrt{2}LRT}{\sigma_m} \tag{4.10}$$

This bounds the Mahalanobis distance between the mixtures corresponding means. When the difference is small, Thm. 2 bounds the difference in entropies. Importantly, the bound shows that if the length of a grid cell is below the standard deviation of the measurement model, then the error from the approximation will be negligible.

4.2.3 Selecting Motion Primitives

A central aspect of our approach is the use of motion primitives to generate actions. To select an informative action quickly, we wish to generate as few as possible without ignoring important directions. In this section we show that mutual information cannot change significantly when the team's movements change on the order of the standard deviation of the measurement model. This suggests that generating multiple actions that are close to one another is computationally wasteful as their mutual information will be similar. It also suggests that the intermediate waypoints from motion primitives (e.g., Fig. 4.1) should be spaced as a function of the standard deviation of the measurement model. We exploit both of these observations when designing the system.

Let $\mathbf{c}$ be an action and $f(\mathbf{z}) = \sum_{i=1}^{M} w_i \mathcal{N}(\mathbf{z}; \mu^{f_i}, \sigma^2_m I)$ be the measurement density it determines when combined with the measurement model and $\mathcal{X}$, the particle set representing the predicted motion of the target. If α is a perturbation to the team's position at each point in time, $\mathbf{c} + \alpha$ is a new action with measurement density $g(\mathbf{z}) = \sum_{i=1}^{M} w_i \mathcal{N}(\mathbf{z}; \mu^{g_i}, \sigma^2_m I)$. $\mathcal{X}$ is the same for $\mathbf{c}$ and $\mathbf{c} + \alpha$, so $f(\mathbf{z})$ and $g(\mathbf{z})$ have the same number of components with identical weights. Similar to the last section, the conditional entropy of the measurements is

the same for the original and perturbed actions. Consequently, to determine the difference in mutual information, we need to determine the difference in entropies between $f(\mathbf{z})$ and $g(\mathbf{z})$.

The entropy of $f(\mathbf{z})$ and $g(\mathbf{z})$ are not necessarily equal as the means of individual components may change. To bound this difference, consider the distance between the i^{th} particle and the r^{th} robot at time k for both actions: $\mu_{k,r}^{f_i} = \|\tilde{\mathbf{x}}_k^i - \mathbf{c}_k^r\|_2$ and $\mu_{k,r}^{g_i} = \|\tilde{\mathbf{x}}_k^i - (\mathbf{c}_k^r + \alpha_k^r)\|_2$. The triangle inequality tells us $|\mu_{k,r}^{f_i} - \mu_{k,r}^{g_i}| \leq \|\alpha_k^r\|_2$. Defining Δ^i as a perturbation to μ^{f_i} that transforms it to μ^{g_i}, and summing over all time steps and robots:

$$\|\Delta^i\|_{\sigma_m^2 I} = \|\mu^{g_i} - \mu^{f_i}\|_{\sigma_m^2 I} \leq \sum_{k=t+1}^{t+T} \sum_{r=1}^{R} \frac{\|\alpha_k^r\|_2}{\sigma_m} \tag{4.11}$$

When the change in a robot's position is small relative to the standard deviation of the measurement model, the mixture components will not be significantly perturbed. By Thm. 2, this means the entropy of the mixture will not change significantly.

4.2.4 Bounding Entropy Difference using Kullback-Leibler Divergence

In previous work, we argued that the KL divergence can be used to bound the difference in entropies of GMMs. However, this reasoning was flawed and we incorrectly concluded that the bounds provided useful insight into how the entropy of a GMM can change. To correct the record, we detail this prior result and clarify the source of the issue.

Lemma 3 (KLD Bound on Entropy for Gaussians)**.** *If $f(x)$ and $g(x)$ are two k-dimensional Gaussian densities:*

$$|\,\mathrm{H}[f] - \mathrm{H}[g]| < \min\{\mathrm{D}_{\mathrm{KL}}[f \mid\mid g], \mathrm{D}_{\mathrm{KL}}[g \mid\mid f]\} + \frac{k}{2}$$

Lemma 4 (Entropy Upper Bound for Mixture Models)**.** *If $f(x) = \sum_i w_i f_i(x)$ is a mixture model density then $\mathrm{H}[f] \leq \mathrm{H}[w] + \sum_i w_i \mathrm{H}[f_i]$ where $\mathrm{H}[w]$ is the discrete entropy of the mixture weights (Proved by Huber et al. [59]).*

Lemma 5 (Entropy Lower Bound for Mixture Models)**.** *If $f(x) = \sum_i w_i f_i(x)$ is a mixture*

model, $\mathrm{H}[f] \geq \sum_i w_i \mathrm{H}[f_i]$.

Theorem 3 (Entropy Difference for Mixture Models)**.** *If* $f(x) = \sum_{i=1}^{N} w_i f_i(x)$ *and* $g(x) = \sum_{i=1}^{N} w_i g_i(x)$ *are mixture models whose components have equal weights then*

$$|\mathrm{H}[f] - \mathrm{H}[g]| \leq \mathrm{H}[w] + \sum_{i=1}^{N} w_i |\mathrm{H}[f_i] - \mathrm{H}[g_i]|$$

where $\mathrm{H}[w]$ *is the discrete entropy of the mixture weights.*

Theorem 3 does not connect the difference in entropies between f and g to their similarity as measured by KL divergence or a similar quantity. In previous work, we listed a slight variant of it, [15, Thm. 1], which used Lem. 3 to bound the summand in Thm. 3. While that is valid, it is not relevant for GMMs where each component has the same covariance because the summand is always 0: the entropy of a Gaussian only depends on its covariance, so when corresponding components have the same covariance, they have the same entropy, making their difference 0. Because the fixed covariance noise model results in GMMs whose components have identical covariance, it is incorrect to conclude, as we previously did, that a bounded KL divergence between components gives insight into the entropy difference between the GMMs.

A challenge to the pragmatic contribution of Thm. 3 is that the upper and lower bounds used to prove it are strong at different times. The upper bound on the entropy (Lem. 4) is tightest when each of the components is separated and their shared support is minimal [59]. In contrast, the lower bound (Lem. 5) is tightest when each component is essentially identical. It is also worth noting that the bound on entropy difference is looser as the mixture weights come closer to each other. In the worst case, w is the uniform distribution meaning $\mathrm{H}[w] = \log N$ [27]. Note that Thm. 2 has no such offset term, making it a more insightful bound. Consequently, Thm. 2 in this paper enables us to reach the same conclusions we previously made, which we believe resolves the issue.

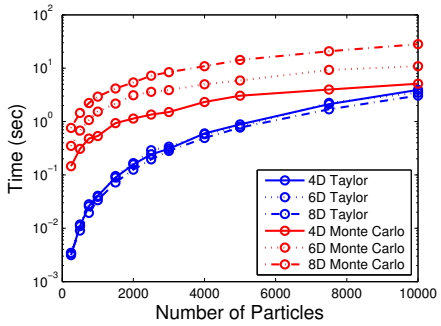


Figure 4.4: Monte Carlo integration vs. 0^{th} order Taylor approximation for evaluating mutual information. The Taylor approximation is more efficient.

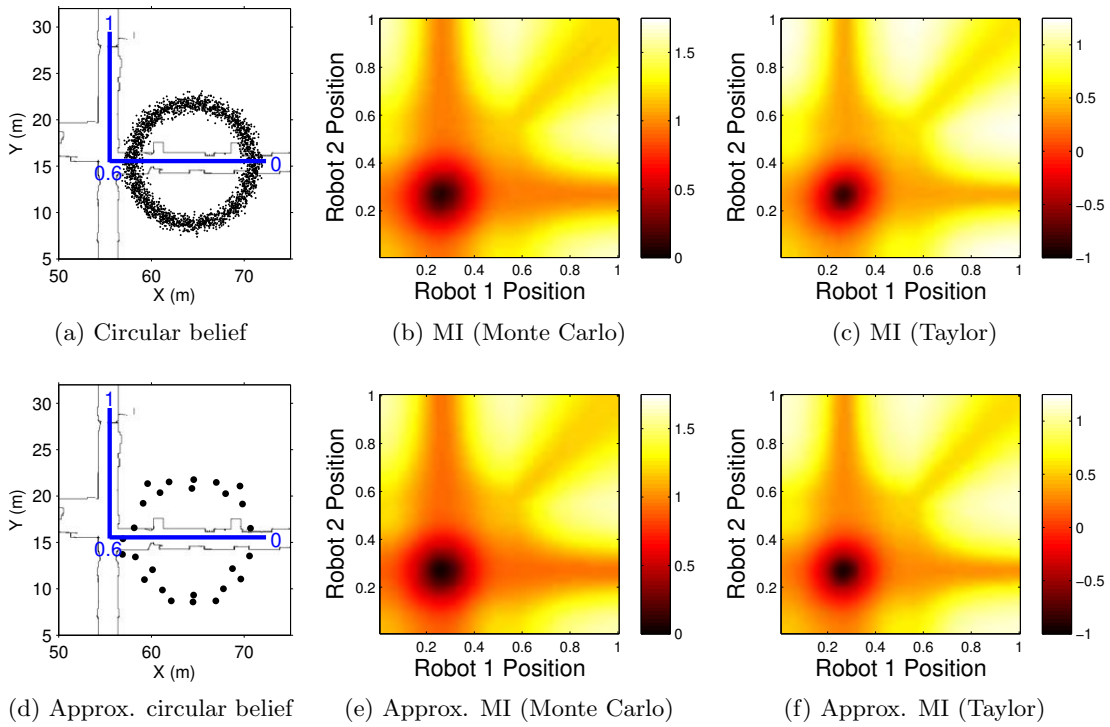
4.3 Experiments

This section describes a series of simulations and real world experiments which evaluate the proposed methodology and design trade-offs. All of our software is written in C++, run on an Intel Core i7 processor, and interfaced using the Robot Operating System [106].

4.3.1 Computational Performance

In Sect. 4.1.3 we discussed how the primary computational challenge of the objective is the calculation of the measurement density's entropy. We now study the performance of Huber's 0^{th} order Taylor approximation and show that the approximation algorithm in Sect. 4.2.2 makes it tractable for longer time horizons. We also discuss the impact of using an alternative filter for estimation and calculating the objective.

To assess the performance of the Taylor approximation, we compare it to the Vegas algorithm – a Monte Carlo integration algorithm provided by the Cuba library [50] – on a variety of mixture models. To generate a mixture we sample particles and a team action uniformly at random and then apply the measurement model. Figure 4.4 shows the time to calculate the entropy of a single mixture using different numbers of particles and measurement dimensions. Teams often have to evaluate dozens to hundreds of actions, so calculations longer than a fraction of a second make the control law infeasible. The data show that unlike Monte Carlo integration, the Taylor approximation is feasible in higher dimensional spaces when the number of particles is not large. Its lack of growth with dimension is not



(a) Circular belief (b) MI (Monte Carlo) (c) MI (Taylor)

(d) Approx. circular belief (e) Approx. MI (Monte Carlo) (f) Approx. MI (Taylor)

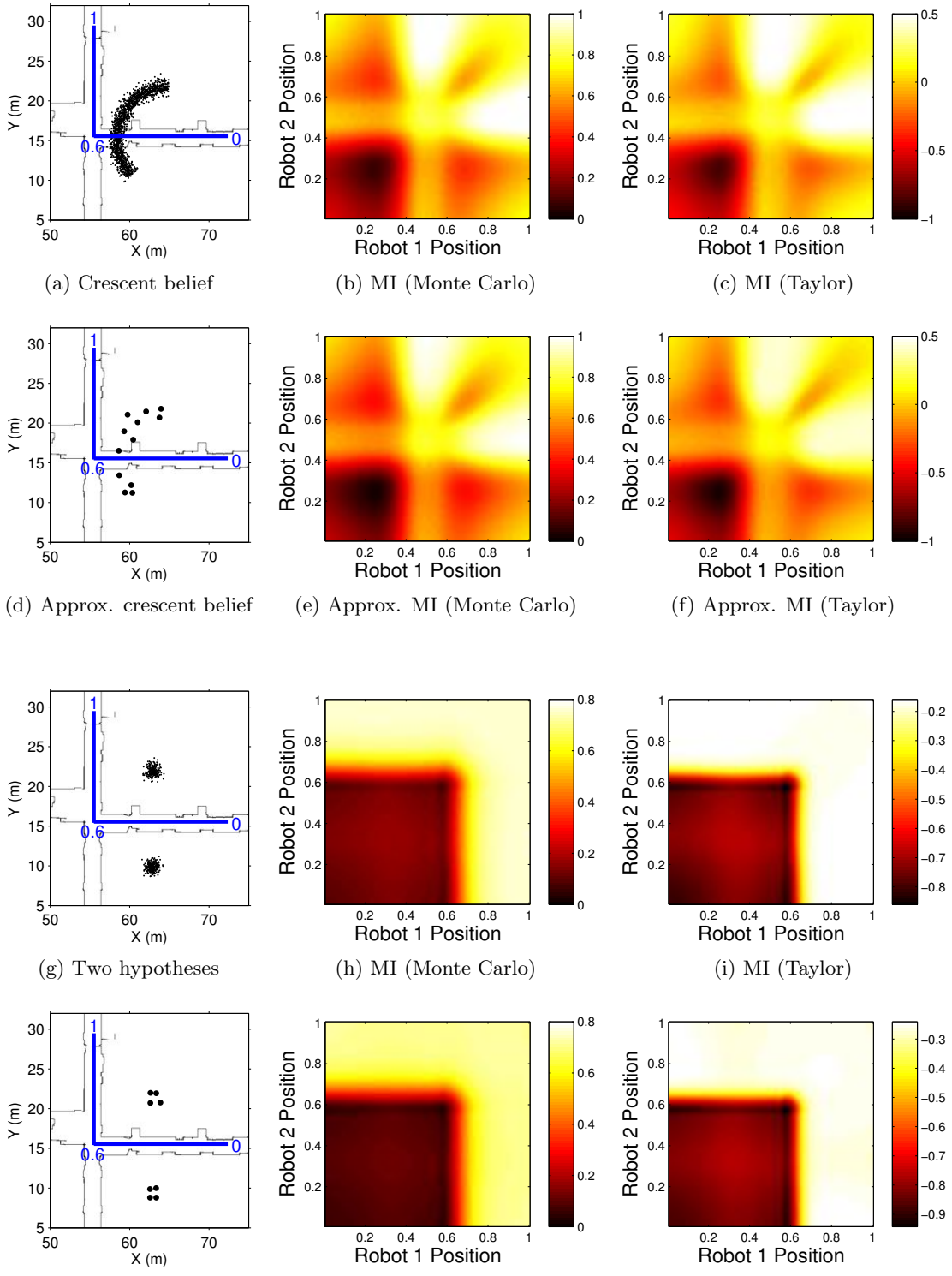
Figure 4.5: Effect of approximating belief on mutual information (MI). Two robots move along a parameterized line from [0,1] (blue, (a) and (d)). Approximating belief introduces limited error into mutual information when calculating it via Monte Carlo integration ((b) vs. (e)) or the Taylor approximation ((c) vs. (f)).

surprising as its dependence on dimension is from simple matrix operations.

4.3.2 Effects of Approximations

To understand the effect of the approximations on the team's decisions, we visualize their impact on the mutual information cost-surface in a few typical scenarios.

Figure 4.5 depicts the effect of approximating the belief on the calculation of mutual information as two robots are constrained to move along a parameterized line from [0, 1]. Given a circular belief distribution represented by a particle filter (M = 2500, Fig. 4.5a), mutual information is computed using Monte Carlo integration for all possible locations of the team with range-only observations (σ_p^2 = 3 m, Fig. 4.5b). The value of mutual information matches intuition; as the robots move away from the center of the belief (0.3 on the line), measurements become increasingly informative with corresponding increases



(a) Crescent belief (b) MI (Monte Carlo) (c) MI (Taylor)

(d) Approx. crescent belief (e) Approx. MI (Monte Carlo) (f) Approx. MI (Taylor)

(g) Two hypotheses (h) MI (Monte Carlo) (i) MI (Taylor)

(j) Approx. two hypotheses (k) Approx. MI (Monte Carlo) (l) Approx. MI (Taylor)

Figure 4.6: Effect of approximating different beliefs on mutual information (MI). Two robots move along a parameterized line from [0,1] (blue, (a), (d), (g) and (j). The approximation introduces limited error for Monte Carlo integration and the Taylor approximation.

in mutual information. An approximate belief representation with fewer particles (N = 26, Fig. 4.5d) corresponding to a grid length of 3.0 m yields mutual information values that are similar to the original distribution (Fig. 4.5e). The mean absolute error between the original and approximate mutual information is 0.03.

It is also important to understand the effect of approximating the belief on the Taylor approximation, as that is how we calculate mutual information in simulations and experiments. Comparing Fig. 4.5c to Fig. 4.5f, it is clear that the approximation introduces limited error for the Taylor approximation as well; the mean absolute error is 0.06. While this is larger than it was for numerical integration, it still represents a small change.

To further verify that the approximation introduces limited error, Fig. 4.6 shows its effect when the belief is distributed as a crescent or two separate hypotheses. These distributions are useful examples as they often arise when using range-only measurements [32, 17]. As in Fig. 4.5, it is clear that the belief approximation introduces limited error for both Monte Carlo integration (mean absolute error below 0.04) and the Taylor approximation (mean absolute error below 0.09). It also significantly reduces the number of particles resulting in a substantial computational speedup ($M = 1500$ to $N = 12$ for the crescent and $M = 500$ to $N = 8$ for two hypotheses).

Looking at the Taylor approximation in isolation might suggest that it performs poorly. For example, it sometimes results in a negative value for mutual information which is a non-negative quantity [27]. However, for the purposes of selecting a control action, the location of the maxima matter more than their value. Given this, by comparing Fig. 4.5b and Fig. 4.5c it is clear that using the Taylor approximation will cause similar actions to be selected: the maxima occur at the same place in both plots, demonstrating that the Taylor approximation performs well.

The previous plots demonstrate that the approximation introduces relatively small amounts of error in typical scenarios. However, they do not illustrate the theoretical bound on entropy difference (Thm. 2). A simple example comparing the entropy of a single Gaussian to a two-component mixture model makes the bound clearer. Let $f(x) = \mathcal{N}(x; 0, 1)$

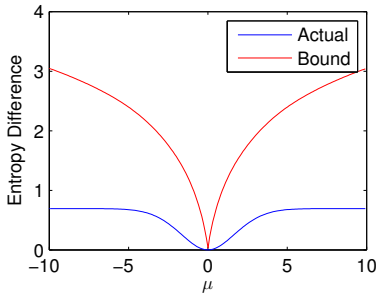


Figure 4.7: Comparison of actual entropy difference to bound from Thm. 2 for mixture models that differ by one component. The theoretical bound is not sharp, but becomes tighter as the components become more similar.

be the standard 1D Gaussian and $g(x;\mu) = 0.5\mathcal{N}(x;0,1) + 0.5\mathcal{N}(x;\mu,1)$ be a 2-component mixture model where the mean of the second component is μ. Figure 4.7 shows the actual entropy difference, $|\,\mathrm{H}[f] - \mathrm{H}[g]|$, between these mixtures along with the upper bound from Thm. 2 for various values of μ. When μ is close to 0 the mixtures are similar and have similar entropies, which the bound captures. As μ increases, the bound is weaker, becoming much larger than the true entropy difference. Despite this fact, Thm. 2 still provides useful theoretical insight into the effect of the approximation.

4.3.3 Flexibility of Motion and Number of Robots

The team can presumably gather better measurements by evaluating more actions, but this adds computational expense, and additional actions may not be helpful. To examine these trade-offs in a controlled setting, we run simulations in an open environment and vary the number of robots in the team and destination points they use to generate actions (see Fig. 4.1a). We evaluate 1) how quickly the team obtains a stable and accurate estimate and 2 how accurate the estimate is over longer periods of time.

For each of these experiments the team moves at 0.2 m/s while the target moves at 0.15 m/s. All members of the team start in the same location with the target 10 m to their right. The target repeatedly drives straight for 5 m and then selects a new direction uniformly at random. To make meaningful comparisons across experiments, the target always follows the same random path. The simulator generates range measurements between the target

and each team member at 10 Hz using a Gaussian whose mean and variance are equal to the true distance. The simulator is asynchronous; the target moves even if the team is planning.

At each point in time, the team plans a 15 second action – simulating 3 target steps – and evaluates mutual information at 3 points along each action. Although each robot will make significantly more than 3 measurements while executing its action, it is not necessary to evaluate mutual information at each position that a measurement will be received. As described in Sect. 4.2.3, mutual information at nearby locations must be similar, so densely evaluating it will not further distinguish which actions are more informative than others.

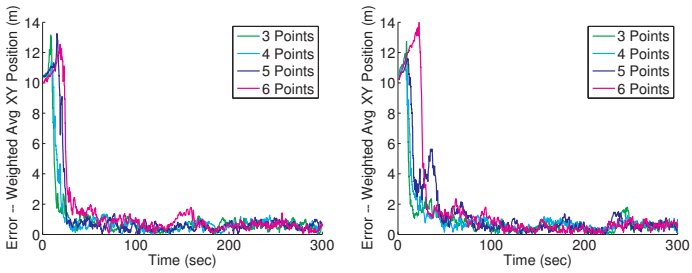
To ensure that the motion model of the filter generates samples in all areas where the target could be, we set the standard deviation to be $\sigma_p = 0.3$, a small multiple of the maximum speed of the target multiplied by the sampling interval. To predict the target's future location (4.1), the team simulates a step every 5 seconds with $\sigma_p = 0.25$.

The team "acquires" the target once they sustain an estimate for 10 seconds with a mean error below 1.5 mand a *spread* (i.e., the square root of the determinant of the covariance) below 1.5. We chose these values as they are close to the best the estimator can do. Table 4.1 shows acquisition times for 5 trials with a variable number of robots and destination points. Surprisingly, the 3 robot team does worse with more destination points. This is a direct consequence of the exponential growth in the number of actions the control law evaluates, which makes the robots spend a long time planning their initial actions. In contrast, the 2 robot team's acquisition time improves by considering more than 3 destination points, after which point it does not change. Planning time is not an issue for the 2 robot team – they have at most 36 actions – which suggests that they are primarily limited by their team size. This is emphasized by the 10 second difference in performance between the best 2 robot time and the best 3 robot time.

To assess the long term performance of the team, we ran the same set of experiments for 5 minutes. Figure 4.8a shows the estimate's error over time. While the target is acquired at different points in time, the final error is the same for all parameters. The computational

Table 4.1: Effect of motion primitives and team size on the time to acquire the target. Mean and std. dev. are shown for 5 trials.

	2 Robots	3 Robots
3 Points	80.55s (22.68)	42.57s (3.55)
4 Points	56.00s (6.07)	49.00s (15.24)
5 Points	56.17s (7.31)	51.83s (15.94)
6 Points	55.10s (12.24)	69.71s (9.94)



(a) Random walk (b) Back and forth

Figure 4.8: Mean error of estimate for various motion primitives with 3 robots. The number of primitives do not affect the long term error regardless of how the target moves.

difficulties that teams with more actions have are not present once the filter has converged, as the belief can be represented with fewer particles. Figure 4.8b shows the long term error from a separate experiment where the target moves 15 m to the right of its starting location and then back. This motion makes it harder to obtain and maintain an accurate estimate because the target moves away from the team for a long time without periodically stopping and turning. The similar results across experiments indicate that the estimation and control approach work well.

4.3.4 Indoor Experiment

To study the real world performance of a team we run two real world experiments where two mobile robots track a third mobile robot as it moves in a loop through an office building as shown in Fig. 4.9. The target's path is 55 m long and it moves at 0.12 m/s, traversing the loop in about 8 minutes. The team starts across the map, travels at 0.2 m/s, and plans using destination points that are 6.0m away. Each robot is equipped with a Hokuyo-URG04LX laser scanner for localization, a 802.11s wireless mesh card for communication, and a range

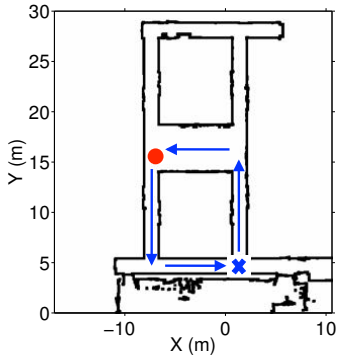
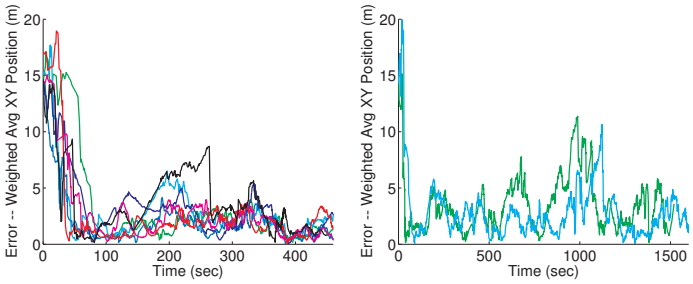


Figure 4.9: Indoor experimental setup. The target starts at the blue "X" and moves in a loop. The team starts at the red circle.

sensor that is commercially available as part of the nanoPAN 5375 Development Kit [86]. The nanoPAN 5375's measurements often have an error larger than 2.0 m [17].

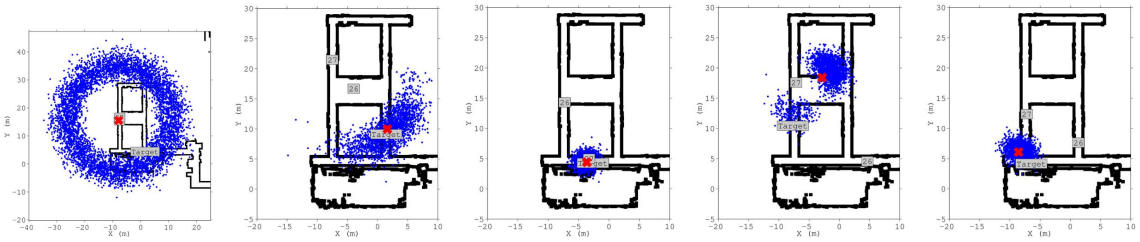
In our first experiment, the target traverses its loop once. Figure 4.10a shows the error of the estimate over 7 trials. Overall, the team is able to obtain a good estimate quickly and maintain it. The mean root mean square error (RMSE) from $t = 60$ s to the end of the experiment was 2.43 m with a standard deviation of 0.68. The authors previously achieved errors of $1.0 - 2.0$ m when localizing *stationary* nodes with the same sensor [17]. Given that the target is moving in this experiment, we believe these results indicate that the team performs well.

In our second experiment, we assess the longer term performance of our approach by having the target traverse the same loop 4 times. Figure 4.10b shows the error of 2 different trials and Fig. 4.11 shows the filter's evolution in trial 1. Again, the team does a good job tracking the target and obtains RMSEs of 3.88 m and 2.83 m. These slightly higher error rates are due to multiple hypotheses that sometimes arise as a result of the limited information range measurements provides. Given the limited travel directions in this environment, it is also difficult for the team to obtain measurements that remove ambiguities. Figure 4.11d and Fig. 4.10b at $t \approx 1000$ s show this happening. However, whenever the error increases, it eventually decreases. It is also important to note that when the estimate's error increases, its covariance also increases; the rise in error is not indicative of the team being overly-confident of a bad estimate.



(a) 1 loop (7 trials) (b) 4 loops (2 trials)

Figure 4.10: Distance from the estimate's mean to the target's true location as the target drives around.



(a) $t = 2.29$ s (b) $t = 47.08$ s (c) $t = 349.64$ s (d) $t = 1028.41$ s (e) $t = 1091.81$ s

Figure 4.11: Evolution of particle filter for trial 1 of the 4 loop experiment. Blue dots represent particles and the red "X" shows the filter's mean. Gray boxes with numbers show the team's position. a) Initially, the target's location is uncertain. b-c) By moving intelligently, the team accurately estimates the target's location. d-e) Multiple hypotheses can arise due to the target's motion and the limited information of range measurements, but they are eventually eliminated.

4.4 Conclusion

In this chapter we presented a control policy for maximizing mutual information over a finite time horizon to enable a team of robots to estimate the location of a mobile target using range-only sensors. To enable calculation of our policy in real-time, we developed an approximation of the belief. By proving that the entropy of a Gaussian mixture model cannot change substantially when its components are perturbed, we argued that these approximations do not significantly affect the teams behavior. We also developed a theoretical design criterion for generating motion primitives by connecting the incremental motions the team makes to the variance of the measurement model. We validated our approach using simulations and real world experiments in which a team of robots successfully track a mobile

target.

While the techniques in this chapter substantially improve the online information gathering strategy, there are two fundamental issues that still need to be addressed. The first is that the complexity of the control law is still exponential in the number of team members. This means that this approach will not scale to teams with more than 2 or 3 members. The second issue is that the control law only works when the team knows that the target is present and can always obtain measurements to it. We address all of these issues in Ch. 5.

4.5 Proofs

4.5.1 Integrating Gaussians Over a Half-Space

Proving Lems. 1 and 2 requires integrating Gaussian functions over a half-space. The following identities will be used several times. Owen [96] gives these identities for 1-dimensional Gaussians, but we have been unable to find a reference for multivariate case. For completeness, we prove them here.

Lemma 6. *If $f(x) = \mathcal{N}(x; \mu, \Sigma)$ is a k-dimensional Gaussian and $A = \{x : a^T x + b > 0\}$ is a half-space then*

$$\int_A f(x)\,dx = \Phi(p) \tag{4.12}$$

$$\int_A x f(x)\,dx = \Phi(p)\,\mu + \phi(p)\,\frac{\Sigma a}{\sqrt{a^T \Sigma a}} \tag{4.13}$$

$$\int_A x x^T f(x)\,dx = \Phi(p)\,(\Sigma + \mu\mu^T) + \phi(p)\left(\frac{\Sigma a \mu^T + \mu a^T \Sigma}{\sqrt{a^T \Sigma a}} - p\frac{\Sigma a a^T \Sigma}{a^T \Sigma a}\right) \tag{4.14}$$

where $p = (a^T\mu + b)/\sqrt{a^T\Sigma a}$ is a scalar, $\phi(x) = \mathcal{N}(x; 0, 1)$ is the PDF of the standard 1-dimensional Gaussian and $\Phi(x)$ is its CDF.

Proof of (4.12). All of these integrals are evaluated by making the substitution $x = Ry$, where R is a rotation matrix that makes the half-space A axis aligned. Specifically, define R such that $a^T R = \|a\| e_1^T$ where e_1 is a k-dimensional vector whose first entry is 1 and all others are 0. This substitution is advantageous, because it makes the limits of integration

for all components of y except y_1 $[-\infty, \infty]$.

Because $R^T x = y$, y is a k-dimensional Gaussian with density $q(y) = \mathcal{N}(y; R^T\mu, R^T\Sigma R)$. The determinant of the Jacobian of y is the determinant of a rotation matrix: $|\partial y/\partial x| = |R^T| = 1$. Substituting:

$$\int_{a^T x+b>0} f(x)\,dx = \int_{(a^T R)y+b>0} |R^T| q(y)\,dy = \int_{\|a\|y_1+b>0} q(y)\,dy = \int_{-b/\|a\|}^{\infty} q_1(y_1)\,dy_1 \tag{4.15}$$

Where $q_1(y_1) = \mathcal{N}(y_1; e_1^T(R^T\mu), e_1^T(R^T\Sigma R)e_1)$ is the density for the first component of y. The final step follows as the limits of integration marginalize $q(y)$. To simplify the remaining integral, apply the definition of R, $q_1(y_1) = \mathcal{N}(y_1; \mu^T a/\|a\|, a^T\Sigma a/\|a\|^2)$, and use $1 - \Phi(-x) = \Phi(x)$:

$$\int_{-b/\|a\|}^{\infty} q_1(y_1)\,dy_1 = 1 - \Phi\left(\frac{-b - \mu^T a}{\sqrt{a^T\Sigma a}}\right) = \Phi(p)$$

□

Proof of (4.13). First, we perform a change of variables so that the integral is evaluated over the standard multivariate Gaussian $g(x) = \mathcal{N}(x; 0, I)$. This substitution is $x = \Sigma^{1/2}y + \mu$ which can be seen by noting that 1) $|\partial y/\partial x| = |\Sigma|^{-1/2}$ and 2) $f(x) = |\Sigma|^{-1/2} g(\Sigma^{-1/2}(x - \mu))$:

$$\int_A x f(x)\,dx = \int_B (\Sigma^{1/2}y + \mu) g(y)\,dy = \Sigma^{1/2}\int_B y g(y)\,dy + \mu\int_B g(y)\,dy \tag{4.16}$$

where $B = \{y : (a^T\Sigma^{1/2})y + (a^T\mu + b) > 0\}$ is the transformed half-space.

To evaluate the first term in (4.16), we calculate $\int_C yg(y)\,dy$, where $C = \{y : c^T y + d > 0\}$ is a new generic half-space. This integral is easier than the original problem as $g(y)$ is the density of a zero-mean Gaussian with identity covariance. To proceed, substitute $y = Rz$ where R is a rotation matrix that makes C axis-aligned (i.e., $c^T R = \|c\|e_1^T$) and observe

that $g(Rz) = g(z)$:

$$\int_C yg(y)\,dy = \int_{c^T Rz+d>0} Rzg(Rz)|R^T|\,dz = Re_1 \int_{-d/\|c\|}^{\infty} z_1\phi(z_1)\,dz_1 = \frac{c}{\|c\|}\phi\left(\frac{d}{\|c\|}\right) \quad (4.17)$$

e_1 appears because $g(x)$ is 0-mean; the only non-zero component of the integral is from z_1. The 1-dimensional integral follows as $d\phi(x)/dx = -x\phi(x)$.

To finish, (4.16) can be evaluated using the formula in (4.12) and (4.17):

$$\int_A xf(x)\,dx = \Sigma^{1/2}\left(\frac{\Sigma^{1/2}a}{\sqrt{a^T\Sigma a}}\right)\phi(p) + \mu\Phi(p)$$

□

Proof of (4.14). Similar to the last proof, make the substitution $x = \Sigma^{1/2}y + \mu$ with $g(y)$ as the standard multivariate Gaussian and expand terms.

$$\begin{aligned}\int_A xx^T f(x)\,dx =& \Sigma^{1/2}\int_B yy^T g(y)\,dy\Sigma^{1/2} + \Sigma^{1/2}\int_B yg(y)\,dy\mu^T \\ &+ \mu\int_B y^T g(y)\,dy\Sigma^{1/2} + \mu\mu^T\int_B g(y)\,dy \end{aligned} \quad (4.18)$$

where $B = \{y : (a^T\Sigma^{1/2})y + (a^T\mu + b) > 0\}$ is the transformed half-space.

To evaluate (4.18), we only need a formula for the first integral; the previous proofs have expressions for the other 3 integrals. To evaluate the first integral, let $C = \{y : c^T y + d > 0\}$ be a new half-space and use the same rotation substitution $y = Rz$ as in the last proof.

$$\begin{aligned}\int_C yy^T g(y)\,dy &= R\left(\int_{z_1\|c\|+d>0} zz^T g(z)\,dz\right)R^T \\ &= R\left(\Phi\left(\frac{d}{\|c\|}\right)I - \frac{d}{\|c\|}\phi\left(\frac{d}{\|c\|}\right)e_1e_1^T\right)R^T \\ &= \Phi\left(\frac{d}{\|c\|}\right)I - \frac{d}{\|c\|}\phi\left(\frac{d}{\|c\|}\right)\frac{cc^T}{\|c\|^2}\end{aligned} \quad (4.19)$$

The previous integral can be evaluated by analyzing each scalar component. g is the standard multivariate Gaussian, so $g(z) = \prod_{i=1}^k \phi(z_i)$. There are three types of terms:

- z_1^2: The limits of integration marginalize g and the resulting integral can be solved using integration by parts:

$$\int_{z_1\|c\|+d>0} z_1^2 g(z)\,dz = \int_{-d/\|c\|}^{\infty} z_1^2 \phi(z_1)\,dz_1 = \Phi\left(\frac{d}{\|c\|}\right) - \frac{d}{\|c\|}\phi\left(\frac{d}{\|c\|}\right)$$

- $z_i^2 (i > 1)$:

$$\int_{z_1\|c\|+d>0} z_i^2 g(z)\,dz = \int_{-d/\|c\|}^{\infty} \phi(z_1)\,dz_1 \int_{-\infty}^{\infty} z_i^2 \phi(z_i)\,dz_i = \Phi\left(\frac{d}{\|c\|}\right)$$

  The integral over z_i is 1 because it's the variance of the standard normal.

- $z_i z_j$ $(i \neq j, i \neq 1)$: These indices cover all non-diagonal elements.

$$\int_{z_1\|c\|+d>0} z_i z_j g(z)\,dz = \int_{\alpha}^{\beta} z_j \phi(z_j)\,dz_j \int_{-\infty}^{\infty} z_i \phi(z_i)\,dz_i = 0$$

  $z_i \neq 1$, so its limits are the real line. Because g is 0 mean the integral is 0.

We now have an expression for all of the terms in (4.18) and we just need to plug them in. Recall that $B = \{x : (\Sigma^{1/2}a)^T x + (a^T\mu + b) > 0\}$. Using $p = (a^T\mu + b)/\sqrt{a^T\Sigma a}$, (4.12), (4.13), and (4.19):

$$\begin{aligned}
\int_{a^T x+b} xx^T f(x)\,dx =& \Sigma^{1/2}\left(\Phi(p)\,I - p\phi(p)\,\frac{\Sigma^{1/2}aa^T\Sigma^{1/2}}{\|a\|^2}\right)\Sigma^{1/2} + \\
& \Sigma^{1/2}\left(\phi(p)\,\frac{\Sigma^{1/2}a}{\sqrt{a^T\Sigma a}}\right)\mu^T + \mu\left(\phi(p)\,\frac{\Sigma^{1/2}a}{\sqrt{a^T\Sigma a}}\right)^T\Sigma^{1/2} + \\
& \Phi(p)\,\mu\mu^T \\
=& \Phi(p)\left(\Sigma + \mu\mu^T\right) + \phi(p)\left(\frac{\Sigma a\mu^T + \mu a^T\Sigma}{\sqrt{a^T\Sigma a}} - p\frac{\Sigma aa^T\Sigma}{\|a\|^2}\right)
\end{aligned}$$

which is the formula we sought.

□

4.5.2 Proof of Lem. 1

The norm can be evaluated by splitting the integral up into regions where the absolute value disappears. Let A be the set where $f(x) > g(x)$ and A^c be the complement of A where $f(x) \leq g(x)$. Noting that $\int_A g(x) = 1 - \int_{A^c} g(x)$:

$$\begin{aligned}\int |f(x) - g(x)|\, dx &= \int_A f(x) - g(x)\, dx + \int_{A^c} g(x) - f(x)\, dx \\ &= 2\left(\int_A f(x) - g(x)\, dx\right) \end{aligned} \tag{4.20}$$

f and g have the same covariance, so f is bigger than g on a half-space $A = \{x : a^T x + b > 0\}$ where $a = \Sigma^{-1}(\mu_1 - \mu_2)$ and $b = (\mu_1 + \mu_2)\Sigma^{-1}(\mu_2 - \mu_1)/2$. This means (4.20) can be evaluated using Lem. 6. To do this, we need to evaluate p for $\int_A f(x)\, dx$ and $\int_A g(x)\, dx$. There are three relevant terms:

$$\mu_1^T a + b = \frac{\|\mu_1 - \mu_2\|_\Sigma^2}{2} \tag{4.21}$$

$$\mu_2^T a + b = -\frac{\|\mu_1 - \mu_2\|_\Sigma^2}{2} \tag{4.22}$$

$$\sqrt{a^T \Sigma a} = \|\mu_1 - \mu_2\|_\Sigma \tag{4.23}$$

Using $\delta = \|\mu_1 - \mu_2\|_\Sigma/2$ we get $(a^T\mu_1 + b)/\sqrt{a^T\Sigma a} = \delta$ and $(a^T\mu_2 + b)/\sqrt{a^T\Sigma a} = -\delta$. Plugging these values into Lem. 6 completes the proof.

4.5.3 Proof of Lem. 2

Let X be a random variable whose density is $|f(x)-g(x)|/\|f-g\|_1$. Calculating the entropy of X is difficult as the expression involves the log of the absolute value of the difference of exponentials. Fortunately, the covariance of X can be calculated. This is useful because the maximum entropy of any distribution with covariance Σ is $(1/2)\log\left((2\pi e)^k|\Sigma|\right)$, the entropy of the multivariate Gaussian [27, Theorem 8.6.5]. By explicitly calculating the determinant of the covariance of X, we prove the desired bound.

To calculate X's covariance, use the formula $\operatorname{cov} X = \mathbb{E}\left[XX^T\right] - \mathbb{E}\left[X\right]\mathbb{E}\left[X\right]^T$. Similar

to the proof of Lem. 1, we evaluate the mean by breaking the integral up into a region A where $f(x) > g(x)$ and A^C where $g(x) \geq f(x)$ vice-versa:

$$\mathbb{E}[X] = \frac{\left(\int_A x(f(x) - g(x))\,dx + \int_{A^c} x(g(x) - f(x))\,dx\right)}{\|f - g\|_1} \tag{4.24}$$

As before, f and g have the some covariance, so $A = \{x : a^T x + b > 0\}$ and $A^c = \{x : (-a)^T x + (-b) \geq 0\}$ are half-spaces with $a = \Sigma^{-1}(\mu_1 - \mu_2)$ and $b = (\mu_1 + \mu_2)\Sigma^{-1}(\mu_2 - \mu_1)/2$.

Each of the terms in (4.24) are Gaussian functions integrated over a half-space, which can be evaluated using Lem. 6. To simplify the algebra, we evaluate the integrals involving f and g separately. This involves calculating a few terms, three of which are repeats: (4.21), (4.22), and (4.23). The other term is $\Sigma a = \mu_1 - \mu_2$. As the difference of the means will arise repeatedly, define $\Delta = \mu_1 - \mu_2$. Noting $2\delta = \|\Delta\|_\Sigma$ and $\phi(x) = \phi(-x)$, the integrals involving f are:

$$\begin{aligned}
&\int_A x f(x)\,dx - \int_{A^c} x f(x)\,dx \\
&\quad = \Phi(\delta)\,\mu_1 + \frac{\phi(\delta)}{2\delta}\Delta - \left(\Phi(-\delta)\,\mu_1 + \frac{\phi(-\delta)}{2\delta}(-\Delta)\right) \\
&\quad = (\Phi(\delta) - \Phi(-\delta))\mu_1 + \frac{\phi(\delta)}{\delta}\Delta
\end{aligned} \tag{4.25}$$

Next, evaluate the integrals in (4.24) involving g. This can be done by pattern matching from (4.25). The main change is that μ_1 becomes μ_2 and the sign of p in Lem. 6 flips, meaning the sign of the arguments to $\phi(\cdot)$ and $\Phi(\cdot)$ flip (see (4.21) and (4.22)).

$$\begin{aligned}
&\int_A x g(x)\,dx - \int_{A^c} x g(x)\,dx \\
&\quad = (\Phi(-\delta) - \Phi(\delta))\mu_2 + \frac{\phi(-\delta)}{\delta}\Delta
\end{aligned} \tag{4.26}$$

Subtracting (4.26) from (4.25), recognizing $\phi(x) = \phi(-x)$, and dividing by $\|f - g\|_1 =$

$2(\Phi(\delta) - \Phi(-\delta))$ simplifies (4.24):

$$\mathbb{E}[X] = \frac{\mu_1 + \mu_2}{2} \tag{4.27}$$

We now evaluate X's second moment. Split the integral up over A and A^c:

$$\mathbb{E}\left[XX^T\right] = \frac{\left(\int_A xx^T(f(x) - g(x))\,dx + \int_{A^c} xx^T(g(x) - f(x))\,dx\right)}{\|f - g\|_1} \tag{4.28}$$

Once again, we evaluate this expression by separately evaluating the integrals involving f and g.

Starting with f and using Lem. 6 with Δ and δ:

$$\int_A xx^T f(x)\,dx = \Phi(\delta)(\Sigma + \mu_1\mu_1^T) + \phi(\delta)\left(\frac{\Delta\mu_1^T + \mu_1\Delta^T}{2\delta} - \delta\frac{\Delta\Delta^T}{4\delta^2}\right) \tag{4.29}$$

$$\int_{A^c} xx^T f(x)\,dx = \Phi(-\delta)(\Sigma + \mu_1\mu_1^T) - \phi(\delta)\left(\frac{\Delta\mu_1^T + \mu_1\Delta^T}{2\delta} - \delta\frac{\Delta\Delta^T}{4\delta^2}\right) \tag{4.30}$$

Taking the difference of (4.29) and (4.30):

$$\begin{aligned}&\int_A xx^T f(x)\,dx - \int_{A^c} xx^T f(x)\,dx \\ &\quad = (\Phi(\delta) - \Phi(-\delta))(\Sigma + \mu_1\mu_1^T) + \frac{\phi(\delta)}{\delta}\left(\Delta\mu_1^T + \mu_1\Delta^T - \frac{\Delta\Delta^T}{2}\right)\end{aligned} \tag{4.31}$$

To evaluate the integrals in (4.28) involving g, we can pattern match using (4.29) and (4.30). As in the derivation of (4.26), this involves negating the p terms and replacing μ_1 with μ_2.

$$\begin{aligned}&\int_A xx^T g(x)\,dx - \int_{A^c} xx^T g(x)\,dx \\ &\quad = (\Phi(-\delta) - \Phi(\delta))(\Sigma + \mu_2\mu_2^T) + \frac{\phi(\delta)}{\delta}\left(\Delta\mu_2^T + \mu_2\Delta^T + \frac{\Delta\Delta^T}{2}\right)\end{aligned} \tag{4.32}$$

Note the sign difference of $\Delta\Delta^T$ compared to (4.31).

To finish calculating the second moment, subtract (4.32) from (4.31) and divide by

$\|f-g\|_1$, simplifying (4.28)

$$\mathbb{E}\left[XX^T\right] = \frac{1}{2}(2\Sigma + \mu_1\mu_1^T + \mu_2\mu_2^T) + \frac{\phi\left(\delta\right)}{\delta\|f-g\|_1}\Delta\Delta^T \tag{4.33}$$

We can now express the covariance of X.

$$\mathbb{E}\left[XX^T\right] - \mathbb{E}\left[X\right]\mathbb{E}\left[X\right]^T = \Sigma + \frac{\Delta\Delta^T}{4} + \frac{\phi\left(\delta\right)}{\delta\|f-g\|_1}\Delta\Delta^T \tag{4.34}$$

$$= \Sigma + \left(\delta^2 + \frac{2\phi\left(\delta\right)\delta}{2\Phi\left(\delta\right)-1}\right)\frac{\Delta\Delta^T}{4\delta^2} \tag{4.35}$$

Which follows as $(\mu_1\mu_1^T + \mu_2\mu_2^T)/2 - (\mu_1+\mu_2)(\mu_1+\mu_2)^T/4 = \Delta\Delta^T/4$.

To calculate the determinant of the covariance, factor Σ out of (4.35) and define $\alpha = \delta^2 + \frac{2\phi(\delta)\delta}{2\Phi(\delta)-1}$:

$$|\operatorname{cov} X| = \left|\Sigma\left(I + \alpha\frac{\Sigma^{-1}\Delta\Delta^T}{4\delta^2}\right)\right| = |\Sigma|\left(1+\alpha\right) \tag{4.36}$$

The last step follows from the eigenvalues. $\Sigma^{-1}\Delta\Delta^T$ only has one non-zero eigenvalue; it is a rank one matrix as Σ^{-1} is full rank and $\Delta\Delta^T$ is rank one. The trace of a matrix is the sum of its eigenvalues, so $\operatorname{tr}(\Sigma^{-1}\Delta\Delta^T) = \operatorname{tr}(\Delta^T\Sigma^{-1}\Delta) = \|\Delta\|_\Sigma^2 = 4\delta^2$ is the non-zero eigenvalue. Consequently, the only non-zero eigenvalue of $\alpha\Sigma^{-1}\Delta\Delta^T/4\delta^2$ is α. Adding I to a matrix increases all its eigenvalues by 1 so the only eigenvalue of $I + \Sigma^{-1}\Delta\Delta^T/4\delta^2$ that is not 1 has a value of $1+\alpha$. The determinant of a matrix is the product of its eigenvalues, so $\det(I + \Sigma^{-1}\Delta\Delta^T/4\delta^2) = 1+\alpha$. As discussed at the beginning of the proof, substituting (4.36) into the expression for the entropy of a multivariate Gaussian achieves the desired upper bound.

4.5.4 Proof of Theorem 2

To prove the theorem, we build on Thm. 1. However, it cannot be directly applied because it is difficult to evaluate 1) the L_1 norm between GMMs and 2) the entropy of the normalized difference of mixture models. However, from Lem. 1 and Lem. 2 provide a way to evaluate

these quantities when the densities are individual Gaussians. To exploit this fact, we split the problem up and analyze the difference in entropies between GMMs that only differ by a single component.

To start, define $d_j(x) = \sum_{i=1}^{j} w_i g_i(x) + \sum_{i=j+1}^{n} w_i f_i(x)$. d_j is a GMM whose first j components are the first j components in g and last $n-j$ components are the last $n-j$ components in f. Note that $d_0(x) = f(x)$ and $d_n(x) = g(x)$. Using the triangle inequality:

$$\begin{aligned} |\,\mathrm{H}\,[f] - \mathrm{H}\,[g]\,| &= |\,\mathrm{H}\,[d_0] - \mathrm{H}\,[d_1] + \mathrm{H}\,[d_1] - \mathrm{H}\,[d_n]\,| \\ &\leq |\,\mathrm{H}\,[d_0] - \mathrm{H}\,[d_1]\,| + |\,\mathrm{H}\,[d_1] - \mathrm{H}\,[d_n]\,| \\ &\leq \sum_{j=1}^{N} |\,\mathrm{H}\,[d_{j-1}] - \mathrm{H}\,[d_j]\,| \end{aligned} \tag{4.37}$$

where the last step applied the same trick $n-2$ more times. Because $d_{j-1}(x) - d_j(x) = w_j(f_j(x) - g_j(x))$ each term in the summand can be bounded using Thm. 1:

$$|\,\mathrm{H}\,[d_{j-1}] - \mathrm{H}\,[d_j]\,| \leq -\,\|w_j(f_j - g_j)\|_1 \log \|w_j(f_j - g_j)\|_1 \tag{4.38}$$

$$+\,\|w_j(f_j - g_j)\|_1\,\mathrm{H}\left[\frac{|f_j(x) - g_j(x)|}{\|f_j - g_j\|_1}\right] \tag{4.39}$$

Because $f_j(x)$ and $g_j(x)$ are Gaussians with the same covariance, we can apply Lemmas 1 and 2 to complete the proof.

Chapter 5

Discovering and Localizing Multiple Devices with Range-only Sensors

This chapter addresses the problem of actively controlling robotic teams with range-only sensors to a) discover and b) localize an unknown number of devices. We develop separate information based objectives to achieve both goals, and examine ways of combining them into a unified approach. Despite the computational complexity of calculating these policies for multiple robots over long time horizons, a series of approximations enable all calculations to be performed in polynomial time. We demonstrate the tangible benefits of our approach through a series of simulations in complex indoor environments.

There are three major components to this problem: 1) estimating the location of discovered devices 2) estimating where undiscovered devices might be, and 3) controlling the team to reduce the uncertainty of both estimates until the team is confident that every device has been discovered and localized. Figure 5.1 illustrates the complexity of these goals, and shows why the team must account for the limited information their sensors provide, as well as their limited mobility in office environments.

The primary contribution of this chapter is an approach for controlling a multi-robot

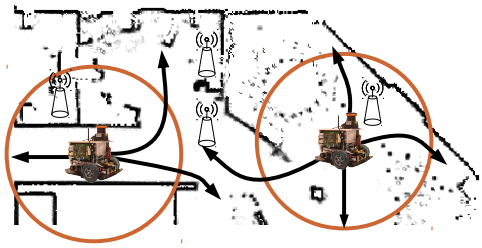


Figure 5.1: Problem overview. A robotic team must discover and localize an unknown number of devices (shown as beacons) using range-only sensors. The team must consider multiple trajectories (black arrows) and the finite range of their sensors (orange circles).

team to concurrently discover and localize an unknown number of devices using noisy range-only sensors. To start, in Sect. 5.1 we present a generic algorithm for reducing the uncertainty of an estimate by maximizing mutual information. We then separately address the problems of actively localizing discovered devices in Sect. 5.2 and actively discovering devices in Sect. 5.3. In Sect. 5.4 we discuss how the team can simultaneously localize and discover devices by creating a unified objective function that can be used in the algorithm in Sect. 5.1. We evaluate the performance of our approach in Sect. 5.5 through a series of simulations and demonstrate that it outperforms two baseline strategies with teams of up to 8 robots.

This chapter originally appeared in [18].

5.1 Preliminaries

5.1.1 Assumptions

In the problem we are considering a robotic team must localize all devices within one floor of a modern building. We assume the building has a wireless network that allows high bandwidth communication throughout the team at all times. This enables us to adopt centralized estimation and control strategies where all measurements are aggregated at a single robot who sends commands to the rest of the team. In situations where the building's network is only available in certain places, we could use the approach proposed by Dames and Kumar [29]. We also assume that robot's are equipped with a map of the

Algorithm 1: Adaptive Sequential Information Planning (ASIP)

Iteration = 0 // Iteration counter
repeat
  Iteration = Iteration + 1
  $\boldsymbol{\tau} = t + 1 : t + (T \cdot \text{Iteration})$ // Adapt end time of plan
  $\mathcal{C}^* = \{\}$ // Best team trajectory, $\mathcal{C}[r]$ is trajectory of robot r
  $\text{Info}^* = 0.0$ // Information resulting from current $\mathcal{C}^*$
  for each robot $r = 1 : R$ **do**
    $\mathcal{P} =$ Plan trajectories robot i can follow over $\boldsymbol{\tau}$
    $\mathcal{C} = \mathcal{C}^* \times \mathcal{P}$ // Trajectories for robots $1 : r$; trajectories of $1 : r - 1$ are fixed
    Info, $\mathbf{c}_{\boldsymbol{\tau}}^{r*} = \max_{\mathbf{c}_{\boldsymbol{\tau}} \in \mathcal{C}} \mathrm{I}_{\mathrm{MI}}[\mathbf{x}_{\boldsymbol{\tau}}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{c}_{\boldsymbol{\tau}}]$ // r's best action given previous actions
    if Info $-$ Info* $>$ MinRobotInfo **then**
      $\mathcal{C}^*[r] = \mathbf{c}_{\boldsymbol{\tau}}^{r*}$, $\text{Info}^* = \text{Info}$
    else
      $\mathcal{C}^*[r] = \emptyset$ // r should do nothing; it can't further reduce uncertainty of $\mathbf{x}_{\boldsymbol{\tau}}$
until Info* $>$ MinTotalInfo **or** Iteration $>$ MaxIteration

environment and are capable of localizing themselves which is reasonable for an indoor office environments. While planning, we do not account for uncertainty in the team's position. We expect that incorporating this uncertainty would not significantly affect the selected control actions as range-only sensors typically have errors of a few meters [17] whereas robot localization solutions are substantially more accurate. We further assume that each measurement has a unique identifier (e.g., MAC address), which is typically the case for range-only sensors that are designed for localization [74, 86, 99, 43].

Because we use a probabilistic approach, we assume that devices' individual positions are independent of each other, that measurements arrive at discrete time steps, and that multiple measurements to a device are conditionally independent given the device's location [134].

5.1.2 Adaptive Sequential Information Planning

We are interested in a general control policy for a team of robots which maximizes mutual information over a finite time horizon. This is difficult for two reasons. First, the number of potential trajectories the team can follow grows exponentially in the team's size. Second, it is unclear how to select an appropriate time horizon over which to plan. To

address these issues we develop "Adaptive Sequential Information Planning" (ASIP, Alg. 1) an algorithm that efficiently selects actions and adapts the time horizon over which it plans.

Before describing ASIP in detail, we formalize a basic approach on which it builds. This is the same approach developed in previous chapters, but we restate it here for ease of reference. Assume the team is trying to estimate some unknown quantity $\mathbf{x}$. At time t, the team plans how it will move over the time interval $\boldsymbol{\tau} \triangleq t+1 : t+T$ by considering a set of trajectories $\mathcal{C}$ that they can follow. A trajectory for an individual robot r is a discrete sequence of poses $\mathbf{c}^r_{\boldsymbol{\tau}} = [\mathbf{c}^r_{t+1}, \ldots, \mathbf{c}^r_{t+T}]$ where $\mathbf{c}^r_k$ is the 2D pose of robot r at time k. The team's trajectories, $\mathcal{C}$, is the Cartesian product of each robot's individual trajectories. As the team moves they will receive a random vector of measurements $\mathbf{z}_{\boldsymbol{\tau}} = [\mathbf{z}_{t+1}, \ldots, \mathbf{z}_{t+T}]$, that each depend on the team's location at that point in time. Our objective is to select the trajectory $c_{\boldsymbol{\tau}} \in \mathcal{C}$ that maximizes $\mathrm{I}_{\mathrm{MI}}[\mathbf{x}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{c}_{\boldsymbol{\tau}}]$, the mutual information between the device's estimate and future measurements the team makes.

A major computational challenge of the basic approach is that the number of trajectories for the team, $|\mathcal{C}|$, grows exponentially in the team's size. To address this issue, ASIP sequentially optimizes mutual information over individual robot's trajectories given the trajectories of preceding robots. This approach is similar to the "Sequential Information Planning" algorithm of Singh et al. [119]. For each robot r, ASIP generates the trajectories it can follow (Alg. 1, Line 8) and combines them with trajectories the preceding robots have already selected (Line 9). This gives a set of trajectories for robots 1 to r, but *only* robot r's trajectory changes: all others are fixed. ASIP then optimizes mutual information over this set of trajectories (Line 10), and repeats until all robots have been considered. The advantage of this approach is that robots still account for each others' movements but mutual information is only calculated $O(RC)$ times, where R is the number of robots and C is the number of trajectories per robot.

Another shortcoming of the basic approach is that it is difficult to determine the time horizon, $\boldsymbol{\tau}$, that the team plans over. If it is too short, the team may get trapped in low information regions, but making it too long will significantly affect computation time.

To balance these issues, ASIP adapts the horizon over which it plans. It does this by requiring the team to decrease the uncertainty of their estimates by a sufficient amount (MinTotalInfo on Line 15). If the team is unable to achieve this gain, it increases the time horizon of the plan (Line 4). It is possible that there are no further actions the team can take to reduce the uncertainty of their estimate. By including a maximum time horizon over which the team can plan (MaxIteration on Line 15), ASIP ensures the team will eventually terminate.

Finally, because the team may be spread out over the environment, only some members of the team may be able to decrease the uncertainty of the estimates over the given horizon. ASIP identifies these robots by examining the change in mutual information given a robot's action (MinRobotInfo on Line 11). Robots that do not reduce the uncertainty are not commanded anywhere (Line 14) freeing them for other tasks. We discuss strategies for what to do with these robots when we discuss how to simultaneously localize and discover devices in Sect. 5.4.

5.2 Actively Localizing Discovered Devices

This section describes a strategy for actively localizing a known number of devices with prior estimates using range-only sensors. In previous work, we presented a similar approach for localizing individual devices [15, 16]. In comparison, here we model the finite range of the sensors.

5.2.1 Estimating Devices' Locations

Because devices are independent of each other and measurements have a known data association, we estimate the position of D different devices using separate particle filters. Each filter uses a measurement model that accounts for noisy range measurements, as well as the probability of a measurement being received.

Formally, the distribution over device d's 2D position at time t is approximated as $p(\mathbf{x}^d \mid \mathbf{z}_{1:t}) = \sum_{i=1}^{N} w_i \delta(\mathbf{x}^d - \tilde{\mathbf{x}}^d)$ where $\delta(\cdot)$ is the Dirac delta function, $\tilde{\mathbf{x}}_i^d$ is the 2D position of the i^{th} particle, w_i is its weight, and $\mathbf{z}_{1:t}$ are measurements the team has made

from time 1 to t [134]. Devices are static, so we omit a time subscript for them, but we do use a zero mean 2D Gaussian with a small fixed covariance for the process model of the filter to avoid particle degeneracy problems.

At time t the team receives a random vector of measurements $\mathbf{z}_t$, where $z_t^{d,r}$ is the 1-dimensional range measurement robot r makes to device d. If the distance from a robot to a device is within the maximum range of the sensor, $z_{\max}$, we assume that with probability γ the robot receives a measurement of the true distance perturbed by Gaussian noise. With probability $1-\gamma$, the robot does not detect the device, and gets a measurement of $z_{\max}$. Defining the true distance as $s = \|\mathbf{x}^d - \mathbf{c}_t^r\|$, the measurement model can be expressed as:

$$p(z_t^{d,r} = z \mid \mathbf{x}^d) = \begin{cases} \gamma\mathcal{N}(z-s,\sigma^2) + (1-\gamma)\mathcal{N}(z-z_{\max},\sigma_{\max}^2) & s < z_{\max} \\ \mathcal{N}(z-z_{\max},\sigma_{\max}^2) & \text{otherwise} \end{cases} \tag{5.1}$$

where σ^2 is the variance of the sensor and $\mathcal{N}(z-\mu,\sigma^2)$ is the likelihood of z with a Gaussian whose mean is μ with covariance σ^2.

Real world sensors may return an error when they fail to measure the distance to a device instead of $z_{\max}$. To compensate for this behavior, a robot can incorporate a "virtual" measurement of $z_{\max}$ for each measurement error. Additionally, when a sensor fails to make a measurement, it will not be perturbed by noise; $\sigma_{\max}^2$ should be 0, resulting in Dirac deltas in (5.1) that are centered on $z_{\max}$. However, this measurement model would be difficult to work with analytically (e.g., the entropy would become $-\infty$). We have also encountered difficulties using small values for $\sigma_{\max}^2$, as this results in a rapid change in variance of particles that are close to $z_{\max}$. Consequently, we set $\sigma_{\max}^2 = \sigma^2$, which is a reasonable model: when a measurement of $z_{\max}$ is incorporated into the filter, particles that are less than $z_{\max}$ away from the robot will become less likely, while those that are farther away will become more likely.

5.2.2 Calculating Mutual Information

To evaluate mutual information between the expected future location of discovered devices and measurements the team will make, we use the particle representation of the device's position, $p(\mathbf{x} \mid \mathbf{z}_{1:t})$ and the range-only measurement model (5.1), to calculate the distribution over expected future measurements, $p(\mathbf{z}_{\boldsymbol{\tau}})$. This approach, which we covered in detail in Ch. 4, results in a computationally intractable problem that we address through a series of approximations.

The expression for mutual information between all devices and measurements for a given team trajectory is:

$$\mathrm{I}_{\mathrm{MI}}[\mathbf{x}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{c}_{\boldsymbol{\tau}}] = \sum_{d=1}^{D} \mathrm{I}_{\mathrm{MI}}[\mathbf{x}^d; \mathbf{z}^d_{\boldsymbol{\tau}}] = \sum_{d=1}^{D} \mathrm{H}[\mathbf{z}^d_{\boldsymbol{\tau}}] - \mathrm{H}[\mathbf{z}^d_{\boldsymbol{\tau}} \mid \mathbf{x}^d] \tag{5.2}$$

$\mathrm{H}[\mathbf{z}^d_{\boldsymbol{\tau}}]$ is the differential entropy of the measurements to device d, while $\mathrm{H}[\mathbf{z}^d_{\boldsymbol{\tau}} \mid \mathbf{x}^d]$ is the conditional differential entropy of measurements to device d. We drop the measurements dependence on the team's trajectory, $\mathbf{c}_{\boldsymbol{\tau}}$, for brevity. The expression is a sum over the information from devices because devices and their associated measurements are pairwise independent, $p(\mathbf{x}, \mathbf{z}_{\boldsymbol{\tau}}) = \prod_d p(\mathbf{x}^d, \mathbf{z}^d_{\boldsymbol{\tau}})$ [27].

Calculating the entropy, $\mathrm{H}[\mathbf{z}^d_{\boldsymbol{\tau}}]$, is difficult because the distribution over future measurements to each device d is a Gaussian mixture model (GMM): $p(\mathbf{z}^d_{\boldsymbol{\tau}}) = \sum_{i=1}^{N} w_i \prod_{j=t+1}^{t+T} \prod_{r=1}^{R} p(z_j^{d,r} \mid \mathbf{x}^d = \tilde{\mathbf{x}}^d_i)$ where w_i is the weight of the i^{th} particle in $p(\mathbf{x}^d)$, $\tilde{\mathbf{x}}^d_i$ is its location, N is the number of particles, R is the number of robots, and T is the time horizon of the plan (Sect. 5.1.2, Sect. 5.2.1). Unfortunately, $p(z_j^{d,r} \mid \mathbf{x}^d)$ is also a GMM when the robot is in range of the device (5.1). Consequently, $p(\mathbf{z}^d_{\boldsymbol{\tau}})$ can be the sum of products of GMMs, resulting in a GMM with a number of components that is exponential in RT. We avoid this computational issue by approximating (5.1) with the most likely component (i.e., the one with maximum weight) when calculating entropy, making the number of components equal to RT. This is reasonable when the probability of detection is high, which is typically the case for range-only sensors. Despite this simplification, $p(\mathbf{z}^d_{\boldsymbol{\tau}})$ is

still a GMM, whose entropy cannot be evaluated analytically. We approximate it using the 2^{nd} order Taylor-series approximation developed by Huber et al. [59]. This approach has a time complexity of $O(N^2RT)$.

Using the conditional independence assumption, the expression for the conditional entropy simplifies $\mathrm{H}[\mathbf{z}^d_{\boldsymbol{\tau}} \mid \mathbf{x}^d] = \sum_{i=1}^{N} w_i \sum_{j=t+1}^{t+T} \sum_{r=1}^{R} \mathrm{H}[z_j^{d,r} \mid \mathbf{x}^d = \tilde{\mathbf{x}}_i^d]$. Re-applying the maximum likelihood estimate for detection, each term is the entropy of a 1-dimensional Gaussian, which can be evaluated in constant time [27].

5.3 Discovering All Devices

In this section we present a method for actively controlling the team to discover all devices. We achieve this by estimating the probability of an undiscovered device being present at any point in the environment, and formulate another information based control law to reduce the uncertainty of this estimate.

5.3.1 Estimating Locations of Undiscovered Devices

We form a probabilistic estimate of any undiscovered device existing at different locations in the environment using a 2D occupancy grid. The grid, $\mathbf{g}$, is made up of a set of G different cells $\{g^1, \ldots, g^G\}$, which are created by uniformly discretizing the environment at a fixed resolution. Each cell is associated with a Bernoulli random variable that represents the probability of any undiscovered device existing at that point in the environment (i.e., $g^i = 1$ means an undiscovered device is present at cell g^i). Like occupancy grids that are used in mapping [134], because device's locations are independent of each other, we also assume the probability of undiscovered devices being in different cells are independent of each other: $p(\mathbf{g}) = \prod_i p(g^i)$. When the team starts, we initialize each cell in the environment with a uniform prior.

We update $p(\mathbf{g})$ using the detection model for the sensors that the team carries. We assume that each robot receives a binary measurement to each cell within the maximum range of its sensor. A reading of 1 corresponds to an undiscovered device being present, while a reading of 0 corresponds to no device being present. Letting $q_t^{i,r}$ be the measurement

robot r gets to cell g^i at time t the measurement model is:

$$\begin{aligned} p(q_t^{i,r} = 1 \mid g^i = 1) &= \gamma & p(q_t^{i,r} = 0 \mid g^i = 1) &= 1 - \gamma \\ p(q_t^{i,r} = 1 \mid g^i = 0) &= 0 & p(q_t^{i,r} = 0 \mid g^i = 0) &= 1 \end{aligned} \tag{5.3}$$

We model the probability of a false positive (i.e., the robot detects a new device is present in a cell when there is no new device at the cell) as 0 because range-only sensors with known association will not return a measurement to a device that does not exist.

Real world range-only sensors will return a set of range measurements to devices that are actually detected. Consequently, if at time t robot r only receives measurements to devices that have been previously observed, we treat that as a measurement of $q_t^{i,r} = 0$ for all cells within $z_{\max}$. Alternatively, if robot r detects a new device, we treat that as a measurement of $q_t^{i,r} = 1$, and immediately initialize a new particle filter to estimate its location.

Using this model with the standard occupancy grid filtering equations it is straightforward to determine the posterior probability of the occupancy grid given all detection measurements the team has made: $p(\mathbf{g} \mid \mathbf{q}_{1:t})$.

5.3.2 Active Device Discovery

To discover all devices, we maximize mutual information between the estimate of undiscovered devices, g, and the expected future binary measurements the team will make, $q_{\boldsymbol{\tau}}$. Similar to Sect. 5.2.2, we use a series of approximations to achieve computational tractability.

For these quantities, mutual information can be expressed as:

$$\mathrm{I}_{\mathrm{MI}}[\mathbf{g}; \mathbf{q}_{\boldsymbol{\tau}}] = \sum_{i=1}^{G} \mathrm{I}_{\mathrm{MI}}[g^i; \mathbf{q}_{\boldsymbol{\tau}}^i] = \sum_{i=1}^{G} \mathrm{H}[\mathbf{q}_{\boldsymbol{\tau}}^i] - \mathrm{H}[\mathbf{q}_{\boldsymbol{\tau}}^i \mid g^i] \tag{5.4}$$

where $\mathbf{q}_{\boldsymbol{\tau}}^i$ is the set of measurements that the team makes of grid cell i at any point in time along the trajectory. Note that $\mathbf{g}$ and $\mathbf{q}_{\boldsymbol{\tau}}$ are both discrete random variables, and so we

use the discrete versions of entropy and conditional entropy, as opposed to the differential entropies in Sect. 5.2.2. (5.4) is a sum because cells and their associated measurements are independent of other cells and measurements $p(\mathbf{g}, \mathbf{q}_{\boldsymbol{\tau}}) = \prod_i p(g^i, \mathbf{q}^i_{\boldsymbol{\tau}})$.

As before, calculating the entropy, $\mathrm{H}[\mathbf{q}^i_{\boldsymbol{\tau}}] = -\sum_{\tilde{\mathbf{q}}} p(\mathbf{q}^i_{\boldsymbol{\tau}} = \tilde{\mathbf{q}}) \log p(\mathbf{q}^i_{\boldsymbol{\tau}} = \tilde{\mathbf{q}})$ is computationally difficult. While $p(\mathbf{q}^i_{\boldsymbol{\tau}} = \tilde{\mathbf{q}})$ can be evaluated by marginalizing over the state:

$$p(\mathbf{q}^i_{\boldsymbol{\tau}} = \tilde{\mathbf{q}}) = p(g^i = 0)p(\mathbf{q}^i_{\boldsymbol{\tau}} = \tilde{\mathbf{q}} \mid g^i = 0) + p(g^i = 1)p(\mathbf{q}^i_{\boldsymbol{\tau}} = \tilde{\mathbf{q}} \mid g^i = 1) \tag{5.5}$$

the sum in the entropy calculation is over *all* possible instantiations $\tilde{\mathbf{q}}$. An individual measurement is binary, so the number of terms grows exponentially in the number of measurements at a cell.

To address this issue, we again approximate entropy. Fortunately, for binary detection measurements, there is relatively little gain in planning to make multiple observations of the same cell. This is because when the probability of detection is reasonably high, even a single observation will substantially reduce the cell's uncertainty. Consequently, we calculate the information gain between all measurements and a cell as the gain from the most informative measurement:

$$\mathrm{I}_{\mathrm{MI}}[g^i; \mathbf{q}^i_{\boldsymbol{\tau}}(c_{\boldsymbol{\tau}})] \geq \max_{q \in \mathbf{q}^i_{\boldsymbol{\tau}}} \mathrm{I}_{\mathrm{MI}}[g^i; q] = \max_{q \in \mathbf{q}^i_{\boldsymbol{\tau}}} \mathrm{H}[q] - \mathrm{H}[q \mid g^i] \tag{5.6}$$

where q is an individual measurement made by one robot at one point in time. The inequality holds because mutual information increases monotonically with additional measurements [27]. (5.6) can be evaluated in $O(QRT)$ where Q is the number of cells one robot observes at a single time step.

5.4 Actively Localizing and Discovering All Devices

We propose several different active control strategies for localizing and discovering all devices by using ASIP (Sect. 5.1) with the objectives in Sect. 5.2 and Sect. 5.3. We also describe two baseline approaches that serve as a useful benchmark for our strategies. For each approach,

Table 5.1: Computational complexity of selecting a trajectory for the team to follow. R is number of robots, D is number of devices, N is particles per device, $|\mathcal{P}|$ is number of trajectories per robot, W is number of waypoints, T is length of horizon, and Q is number of grid cells within the maximum range of the sensor.

Approach	Complexity
Switching	$O(\|\mathcal{P}\|(DN^2RT + QRT))$
Combined	$O(\|\mathcal{P}\|(DN^2RT + QRT))$
Coverage	$O(\|\mathcal{P}\|QRT)$
Exhaustive	$O(\max\{R, W\}^4)$

we are interested in 1) whether all devices will be discovered, 2) whether all devices will be localized, 3) how long it takes to compute plans, and 4) how long it takes to discover and localize all devices. We describe some theoretical properties of points 1-3 for each algorithm. However, analyzing the completion time is difficult given that the number of devices is unknown and the beliefs of the devices' positions evolve in complex ways as a function of many different parameters of our model. Consequently, we evaluate this aspect through a series of simulations described in Sect. 5.5.

5.4.1 Proposed Approaches

Switching Between Localization and Discovery

One approach to localizing and discovering all devices is to adopt a policy where robots make forward progress on either task: each robot either tries to localize known devices, or discover new ones. At each planning step, the team uses ASIP to maximize the information gained about localized devices using (5.2). As described in Sect. 5.1.2, it is possible that only some members of the team will be able to reduce the devices' uncertainty over a given horizon. For robots that cannot help, the team again uses ASIP, but this time maximizes the information gained about undiscovered devices using (5.4).

The team stops when information is 0 (MinTotalInfo in Alg. 1), so this approach will eventually discover all of the devices and localize them to the best of their ability. This is because mutual information is 0 if and only if the two random variables are independent. For the estimate of a device's position, this would mean $p(\mathbf{x} \mid \mathbf{z}_{\boldsymbol{\tau}}, \mathbf{z}_{1:t}) = p(\mathbf{x} \mid \mathbf{z}_{1:t})$ and for a grid cell this would mean $p(g \mid \mathbf{q}_{\boldsymbol{\tau}}, \mathbf{q}_{1:t}) = p(g \mid \mathbf{q}_{1:t})$; in these cases expected future

measurements will not change the estimate. In practice, mutual information will not reach 0 due to the team's noisy estimates. However, we have found that small positive cutoff values (e.g., MinTotalInfo ≈ 0.1) work well.

Combining Localization and Discovery

A general approach for devising a single objective from multiple information-theoretic functions is to normalize each one and introduce weights to trade-off between them [10]. We propose using ASIP with the objectives for localizing and discovering all devices:

$$I[\mathbf{x}; \mathbf{z}_{\boldsymbol{\tau}}] = \alpha \frac{\mathrm{I_{MI}}[\mathbf{x}; \mathbf{z}_{\boldsymbol{\tau}}]}{\max \mathrm{I_{MI}}[\mathbf{x}; \mathbf{z}_{\boldsymbol{\tau}}]} + (1 - \alpha) \frac{\mathrm{I_{MI}}[\mathbf{g}; \mathbf{q}_{\boldsymbol{\tau}}]}{\max \mathrm{I_{MI}}[\mathbf{g}; \mathbf{q}_{\boldsymbol{\tau}}]} \tag{5.7}$$

Where the maximums are taken with respect to all actions the team considers over the current planning horizon and $0 \leq \alpha \leq 1$ is the parameter that weighs the relative importance of the two objectives. The normalization is necessary because the information gains are not directly comparable: the reduction in uncertainty of the devices location (a set of continuous random variables) can be substantially different the reduction in uncertainty of undiscovered devices (a much larger set of Bernoulli variables).

We have encountered two problems combining objectives this way. One is that when one of the terms is small, its impact is substantially elevated by the normalization. To address this issue, we drop a term if the absolute information drops below a small threshold (e.g., 0.1). The other issue is that robots that do not contribute to the change in objective are not given a separate task. Consequently, we modify ASIP so that whenever *any* member of the team does not improve the objective, the planning horizon is extended. This means the team may plan over longer horizons, but ensures the whole team is used more efficiently.

We consider three different values of α. The first is $\alpha = 0.1$, which we refer to as "Discovery" because the team primarily seeks to discover unknown devices. The opposite extreme is to heavily favor actions that reduce the uncertainty of discovered device's positions by setting $\alpha = 0.9$, which we refer to as "Localization." In between these two extremes is $\alpha = 0.5$, which we refer to as "Balanced." Similar to task switching, these approaches

will continue making actions that reduce the uncertainty of both estimates. Consequently, they will eventually discover all of the devices that they can and localize them to the best of their ability.

5.4.2 Baseline Approaches

Coverage

A coverage based strategy is one that seeks to obtain at least one measurement everywhere in the environment. We formulate this policy by setting $\alpha = 0$ in (5.7). While this approach will discover all devices, it will not necessarily localize all of them given the limited information of range-only measurements.

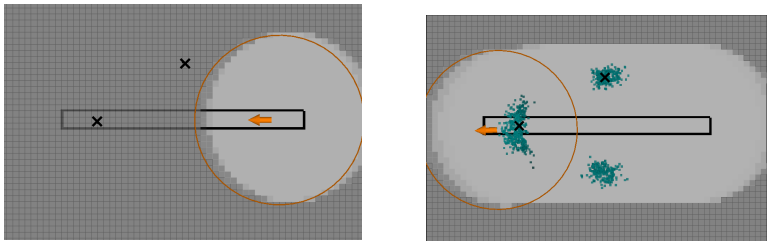
Exhaustive

All of the previous approaches incorporated the uncertainty of discovered or undiscovered devices in some way. A useful comparison is to ignore this uncertainty, and have robots exhaustively gather measurements by visiting every location in the environment. This approach may take longer, but should discover and localize all devices.

There are many different ways to formulate this approach. Here, we manually define a set of W waypoints that any member of the team must visit at least once. Planning paths that minimize the total distance traveled by the entire team is a variant of the multiple traveling salesman problem, and it is unlikely that an exact polynomial time algorithm exists [6]. Instead, we use the nearest neighbor heuristic, and at each planning step assign robots to unvisited waypoints such that the maximum distance any robot travels is minimized. Calculating each assignment can be done in $O(\max\{R, W\}^4)$ time by repeatedly solving linear assignment problems using the Hungarian algorithm [124].

5.5 Evaluation

In this section we evaluate the strategies in Sect. 5.4 and examine their ability to discover and localize devices. To evaluate an approach, we measure the wall clock time – including planning time – that it takes to be confident that all devices are discovered and localized.



(a) Initial estimates (b) Final estimates

Figure 5.2: A single robot (orange arrow) is present in a corridor environment with two devices (black x's). Using the information based strategies, the robot discovers both devices and obtains the best estimate it can for both of them.

We define discovery and localization as the bounding of the uncertainty of each estimate. Formally, we define the discovery of all devices as an indicator function that is 1 when the probability of an undiscovered device at any point in the environment is below 0.05. Similarly, a device is localized when the variance of its x and y position estimate both drop below 0.4 (i.e., $\sigma_x^2 \leq 0.4$ and $\sigma_y^2 \leq 0.4$). These metrics enable reasonable comparisons between different strategies. If the team fails to meet either of these requirements, we define the completion time as the point at which the algorithm stops commanding the team.

We use a real time asynchronous simulator based on ROS. All code is written in C++ and runs on an Intel Core i7 processor. To simulate the range sensors, we set a maximum range of $z_{\max} = 7.0$ m with a variance of $\sigma^2 = 5.0$ m^2 and a measurement rate of 5 hz. We use a constant probability of detection, $\gamma = 0.9$, though in general it could change as a function of the team's distance to a device or their line of sight conditions. Each member of the team is considered to be a differential drive robot with a maximum linear speed of 0.4 m/s. We use a resolution of 0.25 m for the occupancy grid used to discover devices. To plan trajectories, we generate paths to destinations that are within 10.0 m of each robot and discard endpoints that are within 1.0 m of each other. We also stop commanding the team when no trajectory under 60 m is above the minimum information threshold.

5.5.1 Corridor Environment

In our first simulation, we examine the ability of the information based strategies to discover devices and determine that they cannot fully localize all of them. To do this, we consider

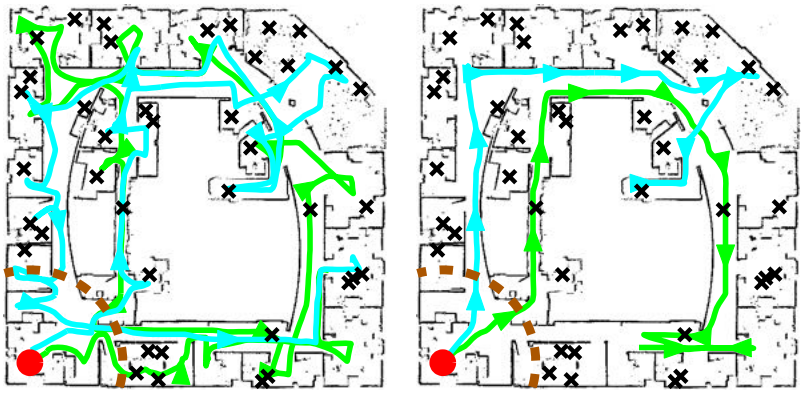
an environment where a single robot is in a narrow corridor with two devices. Because the corridor is narrow and the variance of measurements is high, there are two valid hypotheses for one of the devices. Figure 5.2a shows the initial setup; the orange circle indicates the robot's maximum range and the black x's show the true locations of the devices. Each of the information based strategies moves along the environment, and discovers both devices, with a final result similar to Fig. 5.2. Particles are shown as teal dots, gray cells represent areas where the probability of an undiscovered device is 0.5, while clear cells indicate the probability is close to 0. While the information gain never drops to exactly 0, in all our simulations it eventually decreased below 0.1, resulting in every information based strategy successfully terminating. This demonstrates the ability of these strategies to correctly reason about the sensors they carry, and what effect their actions can have on the position estimate of the devices.

5.5.2 Large Office Environment

To more completely evaluate the utility of information based strategies, we conduct a larger scale simulation study in which teams of up to 8 robots simultaneously localize 40 different devices spread throughout a complex indoor environment. Due to its interesting structure and widespread use in the robotics community, we conduct this simulation in the Intel Research Lab using an occupancy grid generated by Stachniss [92]. For each strategy, we ran 5 trials with teams of 2, 4, and 8 robots.

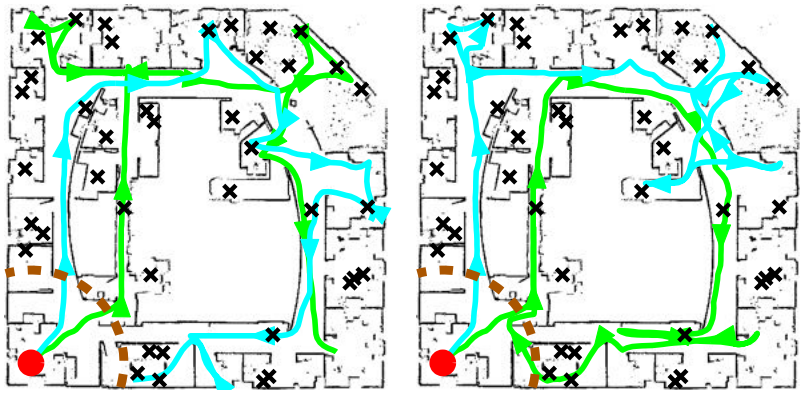
Figure 5.3 shows representative trajectories for each strategy with 2 robots and the location of all 40 devices. The exhaustive strategy visits 71 distinct waypoints in every room of the environment. In contrast, the coverage based strategy has the robots stay in the corridors which is sufficient to observe all of the grid cells. The other strategies follow trajectories in between these two extremes, and generally only enter rooms when devices are present. Note that for the information based approaches, robots naturally spread out, and did not repeatedly get stuck in an area, demonstrating they did not get trapped in local minima and effectively extended their planning horizon when necessary.

In every trial, each approach discovered all devices. The team ensured that the proba-



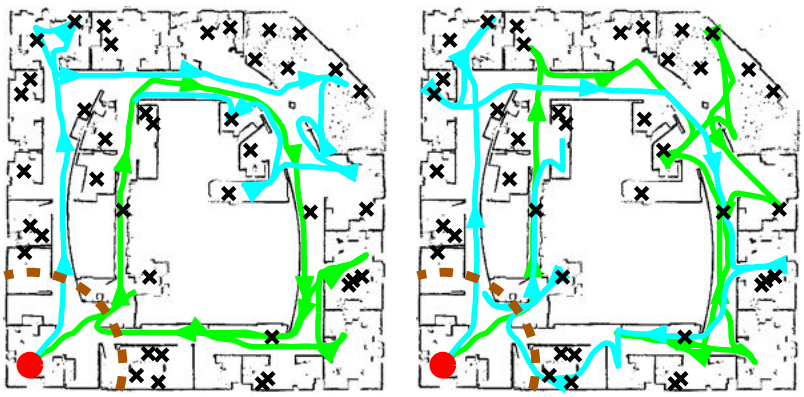
(a) Exhaustive
578 seconds, all devices localized

(b) Coverage
184 seconds, 15 devices *not* localized



(c) Switching
419 seconds, all devices localized

(d) Discovery
393 seconds, all devices localized



(e) Balanced
373 seconds, all devices localized

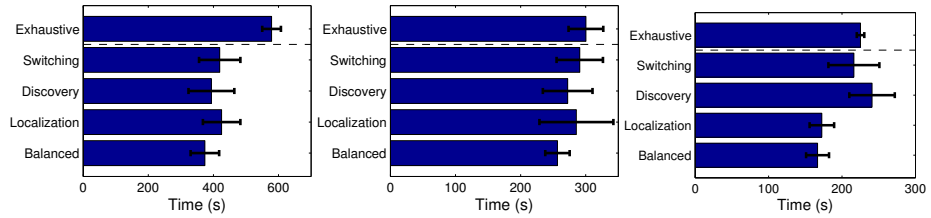
(f) Localization
425 seconds, all devices localized

Figure 5.3: Time to localize targets. The proposed approaches (c)-(f) outperform the baselines (a)-(b) by localizing all devices faster. Solid lines show the team's path and black x's show device locations. The dashed orange line shows the maximum range of the sensors at the team's starting location (red dot).

bility of an undiscovered target being present anywhere in the environment was below 0.05. Not surprisingly, the coverage approach was the fastest to discover all devices. However, it often failed to localize all of them, in many cases obtaining poor estimates for more than 8 devices. This failure is caused by the coverage strategy not rewarding the team for reducing the uncertainty of the devices' position estimates. Localizing a device with range-only sensors requires measurements that are either close to it or at multiple angles relative to it. If these types of measurements are not rewarded, the team will not necessarily obtain them, resulting in devices not being localized. All of the other approaches localized all devices in every trial as they reward measurements that reduce the uncertainty of devices' positions.

Figure 5.4 shows the completion times of all approaches except coverage. We omit the coverage approach's completion time as it routinely failed to localize all devices. Across all team sizes, the information based approaches generally performed better, and never performed substantially worse, than the exhaustive approach. Overall, the balanced strategy discovered and localized all devices the fastest, with an average improvement of 25% over exhaustive. We attribute the balanced strategy's performance to its tendency to gather every potentially useful measurement when it is in an area, meaning it tends to not revisit areas. In contrast, the task switching and localization approaches prioritize reducing the uncertainty of discovered devices. Consequently, they tend to localize all devices quickly, but not fully reduce the uncertainty of grid cells, meaning they must revisit parts of the environment (e.g., rooms without devices). The discovery approach has the opposite issue: it quickly discovers all devices, but then has to retraverse parts of the environment in order to localize them. For completeness, the coverage approach took 184 s, 170 s, and 128 s to complete for teams of 2, 4, and 8 robots respectively.

The performance gains in Fig. 5.4 are substantial given the density of devices throughout the environment. Robots must move to many different areas, resulting in actions closer to that of exhaustive sampling. In environments where the density of devices was lower, we'd expect the information based strategies to outperform exhaustive sampling even further, given that they reduce to the coverage strategy when all discovered devices are localized.



(a) 2 robots (b) 4 robots (c) 8 robots

Figure 5.4: Time to localize all devices. In general, the balanced approach performed the best, outperforming the baseline exhaustive approach by $\approx 25\%$

Table 5.2: Average percent of time spent planning per trial. Due to the series of approximations we use, the team's performance is primarily dominated by travel time.

Robots	Exhaustive	Switching	Localization	Balanced	Discovery	Coverage
2	3.5%	3.2%	3.8%	4.7%	4.4%	5.2%
4	6.7%	4.7%	6.5%	5.8%	7.2%	14.2%
8	17.8%	12.4%	18.5%	22.6%	39.0%	14.0%

Table 5.2 shows the percentage of time that the team spent planning. Overall, the trial time was dominated by traveling places, demonstrating the computational effectiveness of the series of approximations we use.

Finally, the completion times show that adding robots increase performance for all strategies. The information based strategies absolute performance increase over the exhaustive strategy also decreases. This highlights the fact that given unlimited resources, active control strategies offer fewer gains. However, the data in this section shows that with even with moderately sized teams, there is a clear benefit to using informed strategies such as mutual information to select control actions.

5.6 Conclusion

We presented a variety of information based approaches for actively discovering and localizing an unknown number of devices using range-only sensors. Through a series of approximations and a sequential optimization technique, our approaches can be calculated in time that is polynomial in all relevant variables. We compared our approaches to two baseline strategies in a complex indoor environment, and found that their gain in performance was substantial.

Chapter 6

Mapping Using Cauchy-Schwarz Quadratic Mutual Information

The preceding chapters developed a mutual information based control law and applied it to several different range-only estimation problems. A central issue was the difficulty of quickly calculating mutual information. In this chapter, we examine an alternative information-theoretic objective based on Cauchy-Schwarz quadratic mutual information (CSQMI). Like mutual information, CSQMI is theoretically well motivated. However, for certain types of noise models the requisite integrals can be calculated analytically, enabling substantial gains in performance.

To examine the performance of CSQMI, this chapter focuses on the problem of active 3D mapping. This is a challenging active perception problem. Noise, limited angle of view, and limited sensing range are common considerations for many robotics sensors. These systemic limitations yield degraded algorithm performance as the robot also contends with limited mobility, precluding it from positioning its sensors arbitrarily throughout the environment. In this chapter, we seek a method that accounts for these limitations, enabling a robot to build and refine a map through continued control actions and sensor observations. Using active mapping also enables us to examine how well information based approaches generalize to robots with different mobility capabilities; where a robot can move has a substantial

impact on the type of map it can construct.

This chapter makes three primary contributions. First, we show that by using CSQMI, we get significant gains in efficiency, enabling mobile robots to build and refine a 3D map. Second, in contrast to many previous methods, we explicitly model the dependence of separate measurements over multiple time steps to develop online algorithms that search over multiple, multi-step actions enabling us to go beyond one step maximizations. By selecting informative paths, rather than greedily selecting informative destinations, robots choose better control actions that improve the quality of the map with greater efficacy. Third, because we consider models of sensing and mobility, our method naturally applies to both ground and aerial vehicles. We evaluate the approach through a series of simulations and experiments in which mobile robots must build a 3D map using an RGB-D sensor. Our results show that maximizing CSQMI results in a robot exploring the entire environment, while simultaneously ensuring that the map is sufficiently certain.

This chapter appeared in [21] and an accompanying technical report [20].

6.1 Occupancy Grid Mapping

We represent the environment as a 3D occupancy grid. Occupancy grids are advantageous in this setting because they are a volumetric and probabilistic representation of 3D space. The volumetric representation enables the robot to determine where it can move next, and knowing the probabilities that various regions are occupied enables it to reason about how its noisy sensors will change the representation.

The typical occupancy grid mapping algorithm discretizes 3D space into a set of regular cubes that we refer to as cells or voxels (volumetric pixels). The map, $\mathbf{m}$, is made of cells $\{c_1, \ldots, c_{|m|}\}$ each of which have a probability of being occupied. Occupancy grid mapping makes three standard assumptions: 1) cells are independent of one another, so the probability of a particular map is $p(\mathbf{m}) = \prod_i p(c_i)$, 2) the robot's position is known and 3) unobserved cells have a uniform prior of being occupied [134]. The probability that an

individual cell c_i is occupied given sensor measurements from time 1 to t (i.e., $\mathbf{z}_{1:t}$) is:

$$p(c_i \mid \mathbf{z}_{1:t}) = \left(1 + \frac{1 - p(c_i \mid \mathbf{z}_t)}{p(c_i \mid \mathbf{z}_t)} \frac{1 - p(c_i \mid \mathbf{z}_{1:t-1})}{p(c_i \mid \mathbf{z}_{1:t-1})}\right)^{-1} \tag{6.1}$$

This expresses the probability that a cell is occupied in terms of the probability of the cell being occupied given just the latest measurement, $p(c_i \mid \mathbf{z}_t)$, and all other measurements, $p(c_i \mid \mathbf{z}_{1:t-1})$. $p(c_i \mid \mathbf{z}_{1:t})$ can be efficiently calculated by expressing (6.1) in log-odds notation:

$$L(c_i \mid \mathbf{z}_{1:t}) = L(c_i \mid \mathbf{z}_{1:t-1}) + L(c_i \mid \mathbf{z}_t) \tag{6.2}$$

Where $L(c_i \mid \mathbf{z}_t)$ is an inverse measurement model [134].

As described, occupancy grid mapping does not estimate the pose of the robot. Consequently, we assume that the robot is capable of estimating its own pose as well as the pose of its sensor in its local frame. Although they are not the focus of this work, modern SLAM systems [93], which account for uncertainty in the map and the robot's state, can be used to construct an occupancy grid using the maximum likelihood estimate of the robot's path. The grid can typically be updated incrementally, but when estimates of previous poses change substantially (e.g., due to loop closures) a completely new occupancy grid can be quickly regenerated [131].

6.2 Cauchy-Schwarz Quadratic Mutual Information

We are interested in a control policy for the robot that maximizes the Cauchy-Schwarz quadratic mutual information (CSQMI) between the map and measurements the robot will receive over the next several time steps. This formulation gives a unified objective that drives map refinement and exploration, while accounting for limitations of the sensor (e.g., noise, field of view, maximum range) and the robot's own limited mobility.

6.2.1 The Control Policy

We define an action over a time interval $\boldsymbol{\tau} \triangleq t+1 : t+T$ as a discrete sequence of poses, $\mathbf{x}_{\boldsymbol{\tau}} = [\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+T}]$. For a given action, the robot will receive a random vector of measurements $\mathbf{z}_{\boldsymbol{\tau}} = [\mathbf{z}_{t+1}, \ldots, \mathbf{z}_{t+T}]$ that each depend on the pose of the robot at that time. Using the distribution over the current state of the environment, $p(\mathbf{m})$, and its expected future locations, the robot can estimate the distribution over measurements, $p(\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}})$. At each planning step, the robot generates a set of actions $\mathcal{X}$ (Sect. 6.4), and selects the one that maximizes the rate of information gain:

$$\mathbf{x}^*_{\boldsymbol{\tau}} = \arg\max_{\mathbf{x}_{\boldsymbol{\tau}} \in \mathcal{X}} \frac{\mathrm{I}_{\mathrm{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]}{D(\mathbf{x}_{\boldsymbol{\tau}})} \tag{6.3}$$

Where $\mathrm{I}_{\mathrm{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]$ is the CSQMI between the map and measurements the robot will receive if it follows action $\mathbf{x}_{\boldsymbol{\tau}}$, and $D(\mathbf{x}_{\boldsymbol{\tau}})$ is the time to execute the action. In this problem, CSQMI can be expressed as:

$$\mathrm{I}_{\mathrm{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}] = -\log \frac{\left(\sum \int p(\mathbf{m}, \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}) p(\mathbf{m}) p(\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}})\, d\mathbf{z}_{\boldsymbol{\tau}}\right)^2}{\sum \int p^2(\mathbf{m}, \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}})\, d\mathbf{z}_{\boldsymbol{\tau}} \sum \int p^2(\mathbf{m}) p^2(\mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}})\, d\mathbf{z}_{\boldsymbol{\tau}}} \tag{6.4}$$

where the sums are over all possible maps, and the integrals are over all possible measurements the robot can receive. In Sect. 6.3 we give an algorithm that can approximately evaluate this quantity with a time complexity that is *linear* in the number of cells that potential measurements could hit. For a general review of CSQMI and its definition, see Ch. 2.

Unlike the objectives previously developed in this thesis, (6.3) incorporates the time to execute an action, $D(\mathbf{x}_{\boldsymbol{\tau}})$. This is useful as it enables meaningful comparisons between actions that take substantially different amounts of time to execute. In the range-only problems in Chapts. 3-5, robots primarily compared actions of similar length, and so it was not necessary to incorporate a cost. However, in an active mapping scenario a robot often needs to consider its ability to reduce the map's uncertainty where it currently is (short actions) compared to areas much farther away (long actions). Researchers have examined

a variety of utility functions for combining information (reward) and time (cost) [129, 139]. We choose this objective as it can be intuitively understood as the rate of information gain.

6.2.2 Relationship to Mutual Information

A natural question is how CSQMI compares to mutual information (MI), a well understood and widely used objective function for active perception [110, 65]. We explain this relationship using a simple 2D environment when a robot is equipped with an omnidirectional 2D laser. The MI between the map and the robot's future measurements can be expressed as:

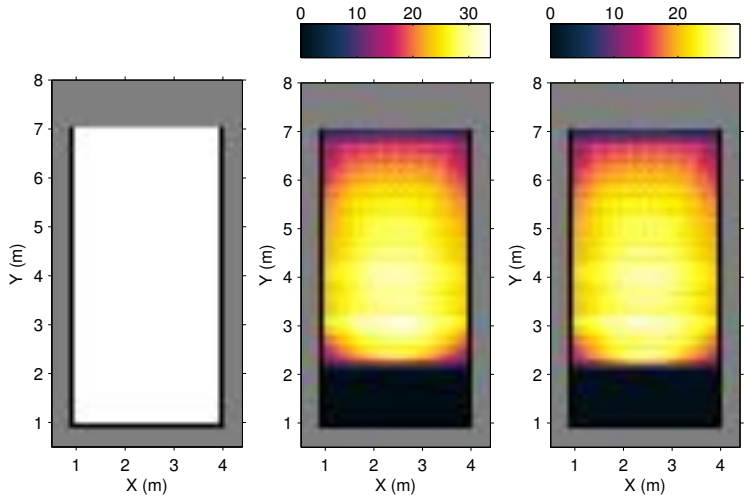
$$\mathrm{I}_{\mathrm{MI}}[\mathbf{m}; \mathbf{z}] = \mathrm{H}[\mathbf{z}] - \mathrm{H}[\mathbf{z} \mid \mathbf{m}] \tag{6.5}$$

where both quantities are implicitly conditioned on the robot's pose. Figure 6.1 shows the reward surfaces for MI and CSQMI. In Fig. 6.1a, white cells have a low probability of occupancy, black cells have a high probability of occupancy, and gray cells are uncertain. Color in Fig. 6.1b and Fig. 6.1c show the value of each objective if the robot places a laser with a minimum range of 0.5 m and a maximum range of 5 m at that position. Julian et al. [65, Fig. 1] use this example to illustrate why choosing MI is beneficial: its maximization would clearly result in the robot moving to regions where it could observe high uncertainty regions of the map. In Fig. 6.1c we see that maximizing CSQMI leads to similar behavior. While frontier-based methods would also perform well in this scenario, directly extending them to 3D environments is difficult as noted by Shen et al. [117] and demonstrated in our experiments (Sect. 6.5).

Importantly, all of the integrals to calculate CSQMI can be evaluated analytically in this setting, making it fast to evaluate, whereas calculating MI requires numerical integration [65]. Therefore, we choose CSQMI to derive our control policies.

6.3 Calculating CSQMI

This section describes how to efficiently calculate the Cauchy-Schwarz quadratic mutual information (CSQMI) between the map and the robot's future measurements (i.e.,



(a) Occupancy grid (b) MI (c) CSQMI

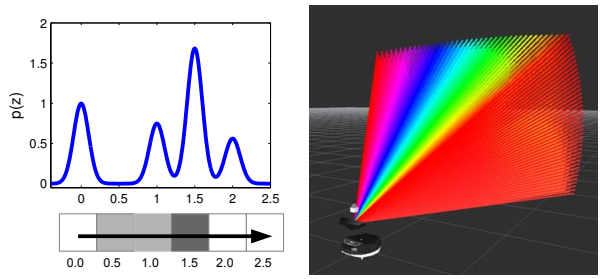
Figure 6.1: Mutual information vs. CSQMI. For an omnidirectional 2D laser in (a), mutual information (MI) and CSQMI are similar and result in the robot moving to make observations that reduce the map's uncertainty.

$\mathrm{I}_{\mathrm{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]$ in (6.3)). We first describe the measurement model, and then use that to calculate the CSQMI for a single beam at a single time step. We then extend this to CSQMI with multiple beams over several time steps.

6.3.1 Measurement Model

As discussed in Sect. 6.2, at time k, the robot receives a random vector of measurements $\mathbf{z}_k$. We model this vector as a collection of B beams so that $\mathbf{z}_k = [z_k^1, \ldots, z_k^B]$, where z_k^b is the 1-dimensional random variable of the distance that beam b travels. We model multiple beams using spherical coordinates by uniformly discretizing the horizontal and vertical field of view of the sensors and generating a beam for each combination of angles (e.g., Fig. 6.2b).

We assume the distribution over distances for a single beam is determined by the true



(a) (b)

Figure 6.2: Beam based measurement model. (a) one beam, the cells it intersects at various distances, and the resulting distribution over measurements when $z_{\min} = 0.5$ and $z_{\max} = 2$. (b) Raycasts for multiple beams.

distance, d, to the first obstacle that the beam intersects:

$$p(z_k^b = z \mid d) = \begin{cases} \mathcal{N}(z - 0, \sigma^2) & d \leq z_{\min} \\ \mathcal{N}(z - z_{\max}, \sigma^2) & d \geq z_{\max} \\ \mathcal{N}(z - d, \sigma^2) & \text{otherwise} \end{cases} \tag{6.6}$$

Where $z_{\min}$ and $z_{\max}$ are the minimum and maximum range of the sensor, and $\mathcal{N}(z - \mu, \sigma^2)$ is a Gaussian with mean μ and variance σ^2. This noise model is somewhat simplistic, as measurements from real sensors have non-axial distance dependent noise as well as radial distortion [121]. While (6.6) only accounts for axial noise and assumes a constant variance, Sect. 6.5 shows that this is sufficient for good experimental results.

Given a map, m, we can calculate the distribution over measurements, $p(z_k^b)$, via marginalization. First, let $\mathbf{c} \subseteq \mathbf{m}$ be the list of C cells within the maximum range of the sensor that a beam intersects (Fig. 6.2a). Given the measurement model, (6.6), cells not in $\mathbf{c}$ have no effect on z_k^b, and can be ignored. The distance to the first occupied cell completely determines the distribution for each beam, so cells in $\mathbf{c}$ behind an occupied cell can be ignored. This means:

$$p(z_k^b) = \sum_{i=0}^{C} p(\mathbf{c} = \mathbf{e}_i) p(z_k^b \mid \mathbf{c} = \mathbf{e}_i) \tag{6.7}$$

Where $\mathbf{e}_i$, $i > 0$, means that the i^{th} voxel in $\mathbf{c}$ is the first occupied cell the beam encounters and $\mathbf{e}_0$ means that all cells in c are unoccupied, so the maximum range will be returned. See Julian et al. [66] for a similar derivation. Fig. 6.2a shows a typical distribution over measurements using (6.7).

6.3.2 CSQMI for a Single Beam

In this section we give the expression for the CSQMI between beam b at time k and the map. This is equivalent to calculating the CSQMI between the beam and cells the beam intersects because the beam cannot reduce the uncertainty of cells that it does not intersect (i.e., $\mathrm{I_{CS}}[\mathbf{m}; z_k^b \mid \mathbf{x}_k] = \mathrm{I_{CS}}[\mathbf{c}; z_k^b \mid \mathbf{x}_k]$). As we show in Appendix 6.7.2, $\mathrm{I_{CS}}[\mathbf{c}; z_k^b \mid \mathbf{x}_k]$ can be calculated by plugging the measurement distribution (6.7) into (6.4):

$$\begin{aligned}\mathrm{I_{CS}}[\mathbf{c}; z_k^b \mid \mathbf{x}_k] = \log \sum_{\ell=0}^{C} w_\ell \mathcal{N}(0, 2\sigma^2) \\ + \log \prod_{i=1}^{C} (o_i^2 + (1-o_i)^2) \sum_{j=0}^{C} \sum_{\ell=0}^{C} p(\mathbf{e}_j) p(\mathbf{e}_\ell) \mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \\ - 2 \log \sum_{j=0}^{C} \sum_{\ell=0}^{C} p(\mathbf{e}_j) w_\ell \mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \end{aligned} \tag{6.8}$$

where $C = |\mathbf{c}|$, $o_i = p(c_i = 1)$ (i.e., the i^{th} cell in $\mathbf{c}$ is occupied), and $p(\mathbf{e}_j)$ is the probability that c_j is the first occupied cell. μ_j is determined by (6.6) and $\mathbf{e}_j$. The w's are:

$$w_\ell = p^2(\mathbf{e}_\ell) \prod_{j=\ell+1}^{C} (o_j^2 + (1-o_j)^2) \tag{6.9}$$

for $0 < \ell < C$ while $w_0 = p^2(\mathbf{e}_0)$ and $w_C = p^2(\mathbf{e}_C)$.

Because they can be calculated iteratively, computing $p(\mathbf{e}_\ell)$ and w_ℓ for all ℓ takes $O(C)$ time. The double sums each have $C+1$ Gaussian components, so calculating them exactly takes $O(C^2)$ time.

The primary advantage of (6.8) is that all of the integration is performed analytically. Another nice property is that the double sums can be approximated in $O(C)$ time with

practically no error. While general techniques like the Fast Gauss Transform [48] could also be used, this problem lends itself to a simpler approximation.

The approximation relies on the fact that Gaussians fall off quickly, and $\mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \approx 0$ when $|\mu_\ell - \mu_j| > \beta\sqrt{2}\sigma$ (i.e., the means are more than a few standard deviations apart) and β specifies the degree of approximation. Determining which means are close to each other is normally difficult, but in this case we know the means monotonically increase with the index, because they are determined by their distance along the beam. Occupancy grid mapping typically uses cell lengths that are slightly larger than the variance of an individual beam, because finer resolutions will not result in significantly different maps. Thus, means separated by more than a few grid cells must be substantially different, so the double sums can be approximated as:

$$\sum_{j=0}^{C} \sum_{\ell=j-\Delta}^{j+\Delta} \alpha_{j,\ell} \mathcal{N}(\mu_\ell - \mu_j, 2\sigma^2) \tag{6.10}$$

where Δ is the number of cells beyond which the contribution is effectively 0 and $\alpha_{j,\ell}$ is a weight that is 0 when $\ell < 0$ or $\ell > C$. As the $\alpha_{j,\ell}$ in (6.8) are small, Δ is typically 3 or 4 for $\beta = 3$, making the complexity of (6.10) $O(C)$, meaning CSQMI's complexity is linear in the number of cells the beam intersects.

6.3.3 CSQMI for Multiple Beams and Time Steps

We now discuss evaluating CSQMI between all beams across all time steps: $\mathrm{I}_{\mathrm{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]$. Exact calculation is computationally intractable because different beams can observe the same cells. As we show, this problem arises from the measurement model and affects all information-theoretic policies and not just those based on CSQMI. Researchers typically address this issue by assuming individual measurements are independent [66, 73]. We adopt a different approach by calculating CSQMI with a subset of the measurements that *are* nearly independent. This enables efficient computation, often yields better performance than the independence assumption (Sect. 6.5), and can be applied to other information based metrics like mutual information.

To simplify the discussion, consider two beams z^i and z^j which originate from different robot poses. For brevity, in the remainder of this subsection, we will often not explicitly condition measurements on the robot's position. Marginalizing over all cells in the map, their joint density is:

$$p(z^i, z^j) = \sum_{\mathbf{o}} p(\mathbf{o}) \sum_{\mathbf{c}^i} p(\mathbf{c}^i) p(z^i \mid \mathbf{c}^i, \mathbf{o}) \sum_{\mathbf{c}^j} p(\mathbf{c}^j) p(z^j \mid \mathbf{c}^j, \mathbf{o}) \tag{6.11}$$

where $\mathbf{c}^i$ is the set of cells that only beam i intersects, $\mathbf{c}^j$ is the set of cells that only beam j intersects, and $\mathbf{o}$ is the set of overlapping cells both beams intersect. When z^i and z^j have no cells in common they are independent because $\mathbf{o}$ is the empty set. CSQMI simplifies in this case: $\mathrm{I_{CS}}[\mathbf{m}; z^i, z^j] = \mathrm{I_{CS}}[\mathbf{c}^i; z^i] + \mathrm{I_{CS}}[\mathbf{c}^j; z^j]$ (see Appendix 6.7.1). Unfortunately, calculating CSQMI – or any other information metric – when the beams overlap requires summing over each instantiation of $\mathbf{o}$ in (6.11). There are $2^{|\mathbf{o}|}$ such instantiations, and while the integrals can be done analytically, exact calculation is computationally infeasible.

However, even when the beams overlap, the probability that they both hit the same cell may be small. For example, at each time step all beams intersect cells close to the robot, but those cells usually have a low probability of occupancy, so none of the beams has a high probability of hitting them. Fig. 6.3 illustrates this point. In these cases, the joint density can be approximated as:

$$p(z^i, z^j) \approx \sum_{\mathbf{c}^i, \mathbf{o}^i} p(\mathbf{c}^i, \mathbf{o}^i) p(z^i \mid \mathbf{c}^i, \mathbf{o}^i) \sum_{\mathbf{c}^j, \mathbf{o}^j} p(\mathbf{c}^j, \mathbf{o}^j) p(z^j \mid \mathbf{c}^j, \mathbf{o}^j) = \hat{p}(z^i) \hat{p}(z^j) \tag{6.12}$$

which means z^i and z^j are approximately independent. Here $\mathbf{o}^i \subseteq \mathbf{o}$ is the subset of overlapping cells that only z^i is likely to hit and $\mathbf{o}^j \subseteq \mathbf{o}$ is the subset of overlapping cells that only z^j is likely to hit.

To establish (6.12), consider the case when the two beams are unlikely to collide (i.e., don't hit the same cell). Let $h_{z^i}^{o_\ell}$ be the event that z^i hits o_ℓ, the ℓ^{th} cell in o. If z^i is likely to hit o_ℓ (i.e., $p(h_{z^i}^{o_\ell}) \neq 0$), then beam z_j is not (i.e., $p(h_{z^j}^{o_\ell}) \approx 0$). For beam z^j, this means $p(z^j \mid c^j, o_\ell) \approx p(z^j \mid c^j)$. Consequently, $p(o_\ell)$ can be factored out of the outermost

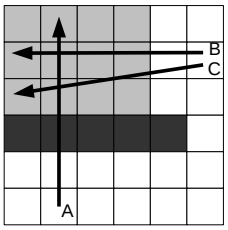


Figure 6.3: Nearly independent beams. Beam A intersects beams B and C, but it is nearly independent of them because the probability it reaches the same cells is close to 0. Beams B and C are not nearly independent, because they will likely hit the same cell.

sum in (6.11) and included in the sum over c^i. This same argument works when $p(h^{o_\ell}_{z^j})$ is non-negligible. If neither beam is likely to reach o_ℓ, it is effectively marginalized out by the outermost sum. Repeating this process for each cell in o results in (6.12), which means $\mathrm{I_{CS}}[\mathbf{m}; z^i, z^j] \approx \mathrm{I_{CS}}[\mathbf{c}^i, \mathbf{o}^i; z^j] + \mathrm{I_{CS}}[\mathbf{c}^j, \mathbf{o}^j; z^j]$

This insight provides a way of approximating CSQMI using a subset of measurements, $\mathcal{Z}$, that are nearly independent:

$$\mathrm{I_{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}] \approx \sum_{z^b_k \in \mathcal{Z}} \mathrm{I_{CS}}[\mathbf{m}; z^b_k \mid \mathbf{x}_k] \tag{6.13}$$

While the sum is an approximation, it tends to lower bound the true CSQMI as $\mathcal{Z}$ does not contain all measurements.

Finding $\mathcal{Z} \subseteq \mathbf{z}_{\boldsymbol{\tau}}$ that maximizes (6.13) is an instance of the maximum-weight independent set problem, meaning it is NP-hard and no polynomial time constant factor approximation algorithm exists [26]. Despite this, we have obtained good results by iterating over all measurements and greedily building $\mathcal{Z}$ (Alg. 2). To test if a new beam z^b_k is independent of those in $\mathcal{Z}$, we check if beams in $\mathcal{Z}$ do not hit any cell that z^b_k hits (Lines 8-11). To check if a collision is unlikely at cell c_i we use the following test:

$$p\left(h^{c_i}_{z^b_k}, \bigcup_{z \in \mathcal{Z}} h^{c_i}_z\right) \leq \min\left\{p(h^{c_i}_{z^b_k}), \sum_{z \in \mathcal{Z}} p(h^{c_i}_z)\right\} \leq \gamma \tag{6.14}$$

where γ is a user specified cutoff. If this is true for each cell that z^b_k, intersects, we treat it as independent and add it to $\mathcal{Z}$ (Line 13). The inequality in (6.14) uses the monotonicity of

Algorithm 2: Calculate $\mathrm{I}_{\mathrm{CS}}[m; z_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]$

```
1: CSQMI = 0 // CSQMI from measurements in 𝒵
2: AnyHit = 0 // Mapping from each cell to probability that any measurement in 𝒵 hits
   it; initially all 0
3: for k = t + 1 to t + T do
4:   for b = 1 to B do
5:     c = Raycast to get C cells that z_k^b intersects
6:     Calculate [p(e_0), ..., p(e_C)] // i^th element is the probability that z_k^b hits cell c_i
7:     // See if z_k^b is independent of measurements in 𝒵
8:     independent = true
9:     for i = 1 to C do
10:      if min{p(e_i), AnyHit[c_i]} > γ then
11:        independent = false
12:    if independent then
13:      CSQMI += I_CS[m; z_k^b | x_k] // Add z_k^b's info
14:      for i = 1 to C do
15:        AnyHit[c_i] += p(e_i)
```

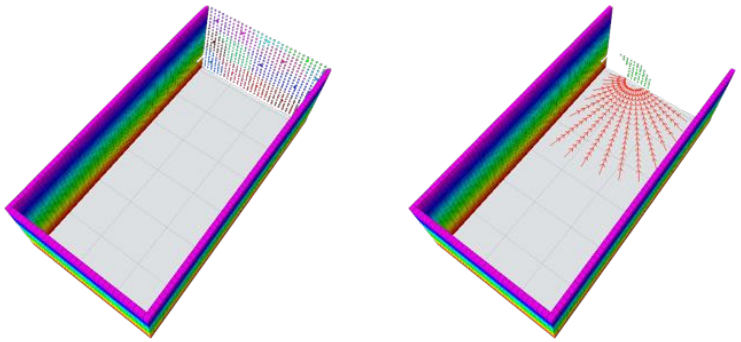
probability and the union bound to upper bound the probability of collision in a way that does not require determining interactions between beams (Lines 14-15).

The computational complexity of Alg. 2 is $O(TBC)$, where T is the number of poses in the action and B is the number of beams the sensor has. The outer loops iterate over all TB beams. For each beam, a raycast is performed and the weights are built, which as described in Sect. 6.3.2 is linear, $O(C)$, in the number of cells the beam intersects. Both the independence check and updating AnyHit iterate over each cell, performing constant time operations at each one.

6.4 Action Generation

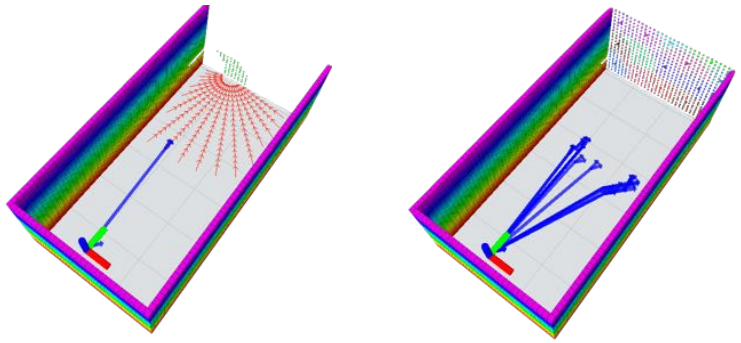
The primary goal of action generation is to create paths whose rate of information gain is large. To do this, we plan shortest paths to places where frontiers can be observed as shown in Fig. 6.4.

First, we identify frontier voxels (i.e., voxels that are unobserved but neighbor observed voxels) [142]. In 3D environments there are too many voxels to generate a path for each one, so we group them into "clusters" with a greedy algorithm (Fig. 6.4a). Each cluster is



(a) Detect and cluster frontiers

(b) Sample poses that view a cluster



(c) Plan a path to the closest sampled pose

(d) Use Dijkstra's algorithm to plan paths that view all clusters

Figure 6.4: Planning paths to view frontier clusters. Occupied voxels in the map are colored by their z-height. (a) Detected and clustered frontier voxels at the end of the hallway (top of the figure). (b) Poses where a robot can view one cluster are shown as red arrows. (c) The shortest path from the robot's location (bottom of the figure) to one of the sampled poses is shown as a blue line. (d) Paths that view all clusters. To plan all paths simultaneously, first build a lookup table of where each cluster can be viewed (Alg. 3), and use it while running Dijkstra's algorithm to plan single source shortest paths.

created by sampling a frontier voxel and grouping it with other voxels within a specified distance (0.4 m works well in practice). This process repeats until no frontier voxels remain.

The robot generates one action per cluster by planning a path to where the cluster is visible. A cluster is "visible" from a pose if it is in the robot's sensor footprint and the probability that a ray from the sensor reaches the cluster is high (i.e., the ray is unobstructed). Planning these paths is difficult for two reasons. First, in 3D environments clusters may exist in places that a robot can observe, but cannot go to (e.g., space above obstacles). Second, a robot must consider its orientation when its sensor has a limited angle of view.

Algorithm 3: Calculate VisibilityLookupTable, a mapping from robot poses to the clusters it can view at that pose

```
1: for each cluster in clusters do
2:     poses = samplePosesWhereVisible(cluster)
3:     for each pose in poses do
4:         VisibilityLookupTable.at(pose).update(cluster)
```

A straightforward approach to generate a path that can view each cluster is to use Dijkstra's algorithm to plan shortest paths from the robot's current pose to every destination in the environment and check to see which clusters are visible from each destination. However, this is computationally expensive because it requires performing a visibility check to all clusters at every destination. To speed the process up, we precompute which clusters are visible from different destinations using Alg. 3. samplePosesWhereVisible(cluster) (Line 2) returns a set of poses where a robot can view a cluster; it can be implemented by sampling poses near the cluster and rejecting those that cannot view it [36]. For example, the poses in Fig. 6.4b can all view a single cluster. Using a lookup table of these poses during the Dijkstra search is substantially faster than performing the visibility checks at each destination. Fig. 6.4c shows a path that can view a single cluster, while Fig. 6.4d shows paths that view all clusters.

6.5 Results

6.5.1 Approximating CSQMI With and Without Independence

We examine the result of our approximate evaluation of the objective (6.13), and compare it to the values we get when we assume measurements are independent. Consider a simple example where a robot with a planar 2D laser must evaluate the CSQMI at two different positions as shown in Fig. 6.5. The laser has 30 beams uniformly spaced over its 90° field of view. In this case, calculating CSQMI by assuming measurements are independent results in an overestimate of the true CSQMI, and chooses pose A. Our approach lower bounds the true value and picks pose B, which maximizes the true CSQMI.

Because we represent the map as an occupancy grid, the independence assumption may

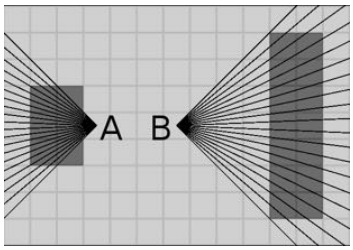


Figure 6.5: Independence leads to overconfidence. Positioning a sensor at B results in a larger CSQMI than at A. The approximation from Sect. 6.3.3 picks pose B, but assuming measurements are independent picks A.

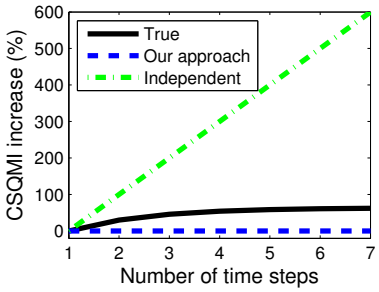
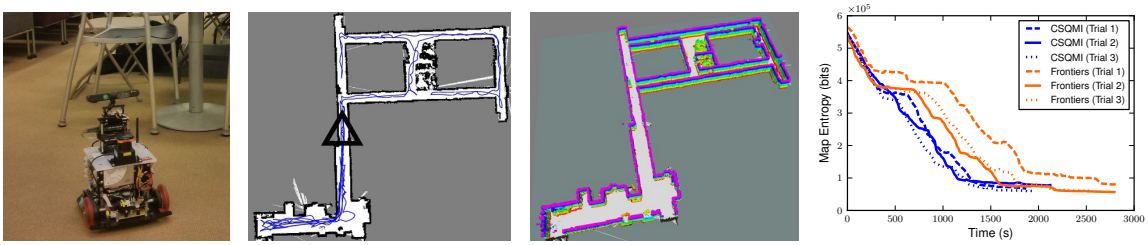


Figure 6.6: Percentage increase in CSQMI for a single beam observing the same cells over multiple time steps. The approximation from Sect. 6.3.3 is more conservative, but also more accurate.

appear more reasonable as a sufficiently fine discretization of space will result in beams that are independent at a particular point in time. However, extending this to multiple time steps is problematic. In particular, with the independence assumption the optimal policy will be to position the sensor at the place that maximizes CSQMI at a single time step, and stays there for all remaining steps. Fig. 6.6 shows this effect in more detail by calculating the CSQMI of a single beam that intersects the same 10 cells multiple times. Each cell is occupied with probability 0.5. Assuming independence results in a linear growth in CSQMI, while the true CSQMI asymptotically reaches a 63% increase. Calculating CSQMI using Alg. 2 only uses the beam at the first time step, as all subsequent beams are dependent on it, which lower bounds the true value. In general, this approach is preferable as maximizing a function by maximizing a lower bound results in better performance than maximizing an upper bound.



(a) Ground robot (b) 2D map with followed path (c) Final 3D map (d) Rate of entropy decrease

Figure 6.7: Ground robot results using CSQMI. (b) The black triangle shows the robot's starting location and the blue line shows the path it followed for one trial. (c) The final 3D map from one trial (ceiling omitted) with occupied cells colored by height. (d) Shannon's entropy of 3D map over time for multiple trials and approaches. Using CSQMI to evaluate the utility of actions helps the robot reduce the map's uncertainty faster.

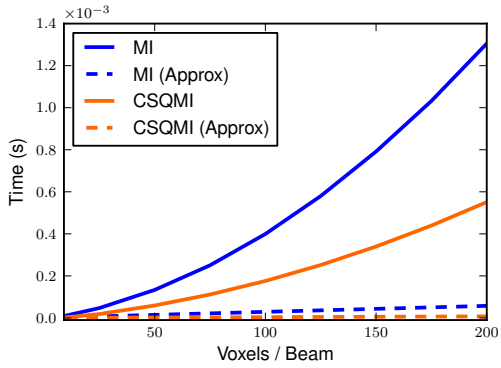


Figure 6.8: Time to evaluate the information of a single beam. CSQMI is faster to compute than MI and the approximation from (6.10) makes computing information linear in the number of voxels.

6.5.2 Computational Performance

In this section, we benchmark the computational performance of CSQMI by comparing it to mutual information (MI). To do this, we examine the time it takes to evaluate the information of a single beam and a group of voxels over a large number of random trials. Voxels are spaced 0.1 m apart, their probability of occupancy is randomly generated, the beam's noise is $\sigma = 0.03$ m and the distance to the last voxel is treated as the maximum range of the sensor. MI must be computed numerically using (6.5). We also compare calculations that exploit the rapid decay of a Gaussian, as we did in (6.10), by ignoring

cross-terms whose means are separated by more than 4 standard deviations. Fig. 6.8 shows the time to compute information with varying numbers of voxels (mean time using 10^5 samples). Using the approximation makes the asymptotic complexity of calculating MI and CSQMI linear instead of quadratic. However, CSQMI is faster in both cases and approximately 7 times faster when using the approximation. This difference is substantial, as a robot often needs to evaluate tens to hundreds of actions during exploration, each of which consists of multiple poses which each have hundreds of beams. Finally, note that this approximation introduced little error; the relative error was below 10^{-3} for all trials.

6.5.3 Experimental Results With a Ground Robot

To test our approach experimentally, we use a ground robot to map a 40 m $\times$ 35 m $\times$ 2 m portion of an office environment in Levine Hall at the University of Pennsylvania. A video of experimental results is available at http://youtu.be/R3CW7VmkjFk.

The ground robot (Fig. 6.7a) is equipped with a UTM-30LX laser range finder and an Asus Xtion Pro as its RGB-D sensor. The UTM-30LX is mounted parallel to the ground, which the robot uses to estimate odometry and to construct a 2D map using a pose graph based SLAM system [134]. The 3D occupancy grid is constructed with a resolution of 0.1 m using data from the Xtion and pose information from the 2D SLAM system. The grid is updated incrementally as the robot moves, and regenerated whenever previous pose estimates of the robot change significantly (e.g., due to loop closures). The robot's maximum linear velocity is 0.5 m/ s.

Evaluating CSQMI requires specifying several parameters. To predict future measurements we discretize the 58° horizontal field of view and 48° vertical field of view of the RGB-D sensor into 20 separate values each, resulting in 400 separate beams. With the 0.1 m resolution of the map, finer angular discretizations typically result in dependent beams that do not substantially change the objective (Sect. 6.3.3). The sensor's minimum range is $z_{\min} = 0.5$ m, its maximum range is $z_{\max} = 4.5$ m, and its noise is $\sigma = 0.03$ m. To classify measurements as independent, we use a cutoff of $\gamma = 0.1$. When evaluating the CSQMI of an action, we include a pose every 3 m along the path or when the robot's

heading changes by more than 50°. All software is written in C++ and executes on the robot which has an Intel Core i5 processor with 8 GB RAM.

Fig. 6.7 shows typical results from maximizing CSQMI. Fig. 6.7b shows a top down view of the path the robot followed overlaid on its 2D laser based map and Fig. 6.7c shows the maximum likelihood estimate of the 3D map at the end of the trial. The final map had low uncertainty for every area that the robot could observe. Overall, the robot's followed path was efficient, with relatively little time spent revisiting areas that were already well observed. While the robot did revisit sections of the environment, particularly in the lower left corner, these actions occur near the end of the trial, after the robot has already observed most of the environment. The accompanying video submission illustrates the robot's overall behavior more clearly.

To determine whether maximizing CSQMI yields improvements in performance over other methods, we compare it to the common approach of driving to the nearest frontier. To implement frontier-based exploration, we use the same action generation approach, but select the action that takes the smallest amount of time to execute without evaluating CSQMI. The CSQMI approach stops executing actions once the highest information gain is below 30 (a low value for the sensor setup). The frontier-based approach terminates once no frontiers are detected. When comparing these strategies, we are interested in how quickly the robot obtains a low uncertainty estimate of the entire map.

To measure how the map's uncertainty changes, we look at the Shannon entropy of cells the robot could possibly observe (i.e., $\mathrm{H}[\mathbf{m}] = -\sum_i o_i \log_2 o_i + (1 - o_i) \log_2 (1 - o_i)$ where o_i is the probability that the i^{th} cell is occupied). We sum over cells in the 3D map whose x and y coordinate are within 0.1 m of an occupied or unoccupied cell in the final 2D map. This excludes most cells that are impossible to observe (e.g., in between hallways) and – because the laser based 2D map is complete – includes cells that the robot may have failed to observe with its RGB-D sensor. Fig. 6.7d shows the entropy over time for 3 trials for both approaches. For reference, there were approximately 6×10^5 observable cells and the maximum entropy of a single cell is 1 bit (i.e., an occupancy probability of 0.5). Overall,

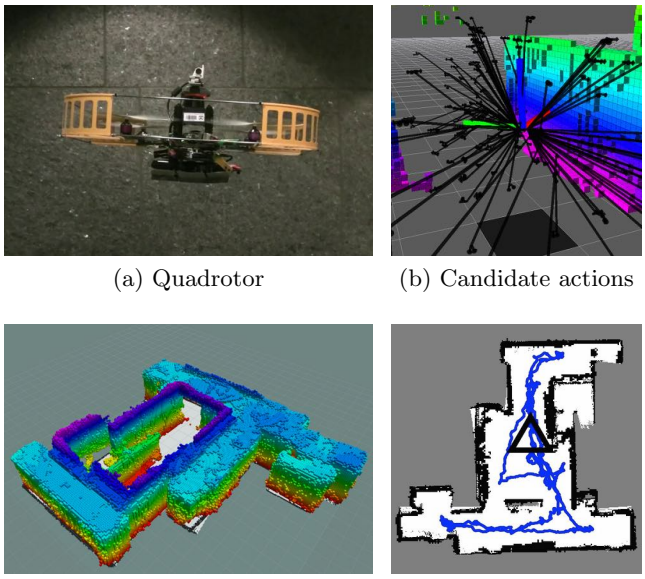
CSQMI significantly outperformed the nearest frontier approach. While the final maps had similar entropies, the rate of decrease using CSQMI was faster. The mean time to completion for CSQMI was 1945 s while for nearest frontier it was 2720 s. The primary reason for this difference is that in 3D environments there are often a large number of frontiers. However, these can be spurious in that the surrounding environment may already be well observed. Evaluating the CSQMI of actions enables the robot to determine whether traveling to a frontier will improve the map, while the frontier-based approach spends a lot of time traveling to areas that have already been well explored. The long flat tail of the entropy curve for the CSQMI trials indicates that the information termination criterion was appropriately conservative.

6.5.4 Experimental Results With an Aerial Robot

To examine how our approach works with robots that have greater mobility, we conducted experiments with a quadrotor in a 18 m $\times$ 17 m $\times$ 3 m section of Skirkanich Hall at the University of Pennsylvania. Unlike a ground robot, a quadrotor can change its height, enabling it to travel to more sections of the environment.

We use an AscTec Pelican (Fig. 6.9a) equipped with a UTM-30LX and Asus Xtion Pro. It performs laser-based localization onboard using the system by Shen et al. [116]. The state estimates are fed into the same SLAM and CSQMI framework used by the ground robot, which run on a wirelessly connected laptop.

Fig. 6.9 shows results from a single trial which took 11 min to complete. Due to its limited battery life, approximately 5 min, we manually landed the quadrotor twice to replace its battery. Obstacles (e.g., tables) were present throughout the environment, which the quadrotor accounted for when generating actions (Fig. 6.9b). Despite these challenges, Figs. 6.9c and 6.9d shows that it built a complete map, demonstrating that CSQMI generalizes to robots with more flexible motions.



(a) Quadrotor (b) Candidate actions

(c) Final map (d) Followed path

Figure 6.9: Quadrotor results. By evaluating CSQMI, the quadrotor is able to build a complete map with low uncertainty.

6.6 Conclusion

We presented a general approach and algorithms to plan multi-step actions for active perception that lend themselves to robots equipped with high data rate sensors like RGB-D cameras. The use of the CSQMI and reasoning about the independence and dependence of observations over successive time steps allowed us to develop algorithms that are both accurate and efficient. The paper included several experimental and simulation results that illustrate the ability of robots to explore and map three-dimensional environments, even with mobility constraints that may constrain them to a plane. Our current work addresses cooperative control of aerial and ground robots.

6.7 Proofs

6.7.1 CSQMI for Independent Subsets

If z^1 and $\mathbf{c}^1$ are independent of z^2 and $\mathbf{c}^2$: $p(z^1, z^2, \mathbf{c}^1, \mathbf{c}^2) = p(z^1, \mathbf{c}^1)p(z^2, \mathbf{c}^2)$. Applying this to $\mathrm{I_{CS}}[z^1, z^2; \mathbf{c}^1, \mathbf{c}^2]$ and defining $dz = dz^1\, dz^2$

$$
\begin{aligned}
&- \log \frac{\left(\sum \int p(z^1, \mathbf{c}^1)p(z^2, \mathbf{c}^2)p(z^1)p(z^2)p(\mathbf{c}^1)p(\mathbf{c}^2)\, dz\right)^2}{\sum \int (p(z^1, \mathbf{c}^1)p(z^2, \mathbf{c}^2))^2\, dz \sum \int (p(z^1)p(\mathbf{c}^1)p(z^2)p(\mathbf{c}^2))^2\, dz} \\
&= - \log \frac{\left(\sum \int p(z^1, \mathbf{c}^1)\, dz^1\right)^2}{\sum \int (p(z^1, \mathbf{c}^1)^2\, dz^1 \sum \int (p(z^1)p(\mathbf{c}^1))^2\, dz^1} - \\
&\quad \log \frac{\left(\sum \int p(z^2, \mathbf{c}^2)\, dz^2\right)^2}{\sum \int (p(z^2, \mathbf{c}^2)^2\, dz^2 \sum \int (p(z^2)p(\mathbf{c}^2))^2\, dz^2} \\
&= \mathrm{I_{CS}}[z^1; \mathbf{c}^1] + \mathrm{I_{CS}}[z^2; \mathbf{c}^2]
\end{aligned}
$$

Induction on this equality establishes it for more general subsets.

6.7.2 Derivation of CSQMI for Single Beam

To derive (6.8), we note that because cells are independent of each other, it is possible to show $\mathrm{I_{CS}}[z; \mathbf{m}] = \mathrm{I_{CS}}[z; \mathbf{c}]$ using the technique in Appendix 6.7.1.

To evaluate $\mathrm{I_{CS}}[z; c]$, we individually calculate each of the three separate terms in (6.4). Starting with the square of the joint distribution we get:

$$\int \sum_{\mathbf{c}} p^2(z, \mathbf{c})\, dz = \int \sum_{k=0}^{C} p^2(z \mid \mathbf{e}_k) \sum_{\mathbf{c} \in E_k} p^2(\mathbf{c})\, dz \tag{6.15}$$

Where E_k is the set of all instantiations of c such that k is the first non-zero element. For notational compactness, we define $E_0 = \{\mathbf{e}_0\}$ as the set whose only instantiation is the 0 vector. Defining the innermost sum as w_k for $k > 0$ and factoring independent cells:

$$w_k = \sum_{\mathbf{c} \in E_k} \prod_{i=1}^{N} p^2(c_i) = p^2(\mathbf{e}_k) \prod_{i=k+1}^{N} (o_i^2 + (1 - o_i)^2) \tag{6.16}$$

Where $o_i = p(c_i = 1)$. Defining $w_0 = p^2(\mathbf{e}_0)$ and applying the formula for the integral of

product of Gaussians:

$$\int \mathcal{N}(z-\mu_1,\sigma_1^2)\mathcal{N}(z-\mu_2,\sigma_2^2)\,dz = \mathcal{N}(\mu_1-\mu_2,\sigma_1^2+\sigma_2^2)$$

to (6.15) gives the first line in (6.8).

Turning to the squared product of the marginals, the two terms share no parameters and can be factored:

$$\begin{aligned}\sum_{\mathbf{c}}\int p^2(\mathbf{c})p^2(z)\,dz &= \sum_{\mathbf{c}} p^2(\mathbf{c})\int p^2(z)\,dz \\ &= \prod_{k=1}^{C}(o_k^2+(1-o_k)^2)\sum_{i=0}^{C}\sum_{j=0}^{C}p(\mathbf{e}_i)p(\mathbf{e}_j)\mathcal{N}(\mu_i-\mu_j,2\sigma^2)\end{aligned}$$

The sum is calculated in the same way as (6.16). The integral is simplified by plugging in the measurement density (6.7), squaring it, and applying the integral of Gaussians once more. This gives us the second line in (6.8).

Evaluating the product of the marginals and the joint is all that remains.

$$\begin{aligned}&\int p(z)\sum_{\mathbf{c}}p(z\mid\mathbf{c})p^2(\mathbf{c})\,dz \\ &= \int p(z)\sum_{k=0}^{C}p(z\mid\mathbf{e}_k)\sum_{\mathbf{c}\in E_k}p^2(\mathbf{c})\,dz \\ &= \int\left(\sum_{j=0}^{C}p(\mathbf{e}_j)p(z\mid\mathbf{e}_j)\right)\left(\sum_{k=0}^{C}w_k p(z\mid\mathbf{e}_k)\right)dz \qquad (6.17)\\ &= \sum_{j=0}^{C}\sum_{k=0}^{C}p(\mathbf{e}_j)w_k\mathcal{N}(\mu_i-\mu_j,2\sigma^2)\end{aligned}$$

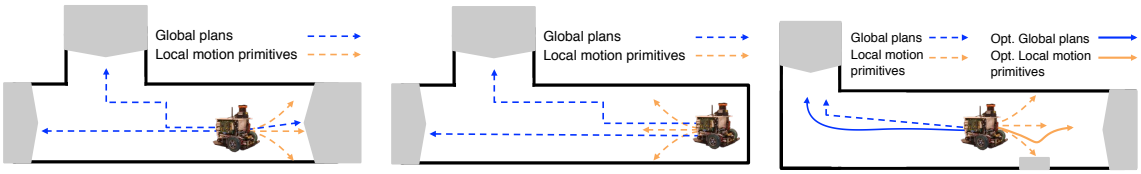
The sum in (6.17) is the same as (6.16). After that, plugging in the measurement density for $p(z)$ and applying the product of Gaussians formula gives the last line in (6.8).

Chapter 7

Planning with Trajectory Optimization for Mapping

Every control policy in this thesis so far has been formulated as the optimization of an information-theoretic objective over a discrete set of actions (e.g., a fixed set of motion primitives or the output of a path planner). While we have obtained many good results using this approach, it has two significant drawbacks. First, the actions are generated independently of the objective and there is no guarantee the selected action is locally optimal. Second, thus far, the generated actions have all been geometric paths (i.e., sequences of waypoints) as opposed to time parameterized trajectories that account for a robot's motion constraints. Consequently, a robot may not be able to follow the generated actions. While these issues can be addressed by generating larger numbers of actions and planning in higher-dimensional spaces, this substantially increases the computational complexity of the control policy. Third, and finally, even generating simple paths can be expensive, particularly in active mapping scenarios (Ch. 6) where paths must be generated throughout the entire environment.

There are several different strategies for generating control actions for the robot including local exploration strategies involving greedy gradient ascent [1, 112, 66] or local motion primitives [10, 15, 40]; and global strategies that use a variant of the frontier method to



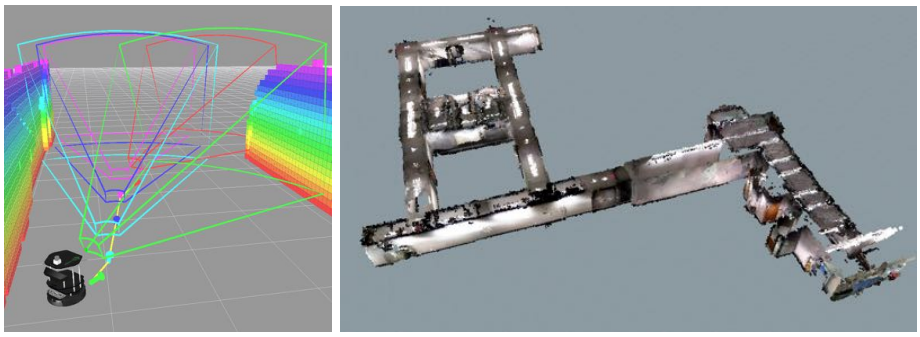
(a) (b) (c)

Figure 7.1: Local and global planning with trajectory optimization. (a) Local motion primitives are useful when a robot is close to uncertain portions of the map (gray blobs) and needs to react quickly to new information. (b) Global plans are useful when the robot is far from uncertain portions of the map as they can escape local minima in the information-theoretic objective. (c) Trajectory optimization can improve the informativeness of both local motion primitives and global plans while respecting a robot's motion constraints.

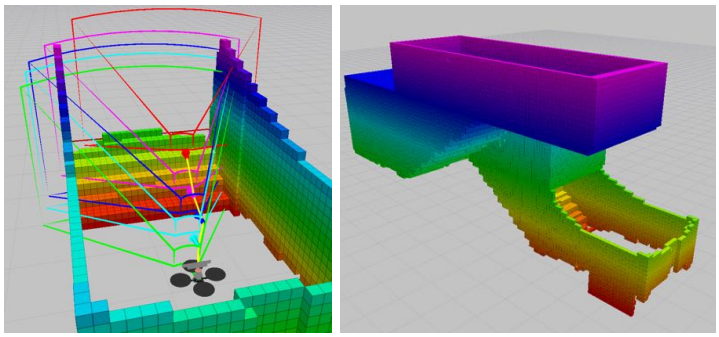
maximize the chosen objective [21, 105]. Local strategies are useful when the robot is close to uncertain portions of the map and constantly observing new areas (Fig. 7.1a), but are susceptible to local minima [123]. Global strategies can escape local minima when the robot is far from uncertain portions of the map (Fig. 7.1b), but are computationally expensive and may produce coarse paths that result in jagged robot motion [114].

We address all of these issues by extending the approach developed in Ch. 6. The primary contribution of this chapter is a two stage approach for generating and refining trajectories that substantially improves a robot's ability to quickly reduce the map's uncertainty. First, we generate trajectories using a combination of global planning and local motion primitives, enabling a robot to escape local minima and replan quickly as the map changes. Second, we contribute a method for improving both local motion primitives and global paths using gradient-based trajectory optimization that maximizes the CSQMI objective while satisfying the robot's motion constraints (Fig. 7.1c). This improvement is particularly important when mapping 3D environments with robots that have high-dimensional state spaces and non-trivial motion models.

We evaluated our approach through a series of simulations and experiments on a ground robot and an aerial robot mapping unknown 3D environments. In real-world experiments we show that by including global plans, local motion primitives, and trajectory optimization, a ground robot was able to construct a complete 3D map 3.3× faster than a strategy that



(a) Mapping an indoor environment with a ground robot



(b) Mapping a stairwell environment with an aerial robot

Figure 7.2: Results with ground and air robots. (a) Mapping an indoor environment with a ground robot equipped with a RGB-D sensor. (b) Mapping a stairwell environment with a quadrotor equipped with a RGB-D sensor. We contribute a two stage planning approach: 1) using a global planner and local motion primitives to choose trajectories that maximize an information-theoretic objective based on Cauchy-Schwarz quadratic mutual information (CSQMI) and 2) using trajectory optimization to further maximize the CSQMI objective.

drives to the closest frontier [142] and 2.3× faster than the information-based strategy in Ch. 6 that only considers global plans.

This chapter will appear in [19].

7.1 Problem Definition

Our basic problem formulation of the active mapping problem remains the same as in Ch. 6. To represent the map and measurements, we continue to use occupancy grids (Sect. 6.1) and a beam based model with Gaussian noise (Sect. 6.3.1).

The only difference in our problem definition for this chapter is the formulation of the control policy. Instead of only seeking a set of states for the robot to follow, we explicitly

include its control inputs and motion model. Specifically, given a map $\mathbf{m}$ at time t, we seek a set of control inputs for the robot so that it gathers measurements which reduce the uncertainty of the map as quickly as possible. To achieve this, we formulate an optimization problem over the time interval $\boldsymbol{\tau} \triangleq t+1 : t+T$ to find controls $\mathbf{u}_{\boldsymbol{\tau}} = [\mathbf{u}_t, \ldots, \mathbf{u}_{t+T-1}]$ that move the robot to states $\mathbf{x}_{\boldsymbol{\tau}} = [\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+T}]$ where it will obtain measurements $\mathbf{z}_{\boldsymbol{\tau}} = [\mathbf{z}_{t+1}, \ldots, \mathbf{z}_{t+T}]$ that reduce the map's uncertainty. The future states of the robot are determined by the deterministic dynamics of the robot, $\mathbf{x}_{i+1} = f(\mathbf{x}_i, \mathbf{u}_i)$. Our objective is to find controls that maximize the rate of some measure of information gain between the map, $\mathbf{m}$ and future measurements the robot will make, $\mathbf{z}_{\boldsymbol{\tau}}$:

$$\begin{aligned} \max_{\mathbf{x}_\tau,\ \mathbf{u}_\tau} \quad & \frac{\mathrm{I_{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]}{D(\mathbf{u}_{\boldsymbol{\tau}})} \\ \text{s. t.} \quad & \mathbf{x}_{i+1} = f(\mathbf{x}_i, \mathbf{u}_i), \\ \forall t<i<t+T-1 \quad & \mathbf{x}_i \in \mathcal{X}_{\text{feasible}}, \mathbf{u}_i \in \mathcal{U}_{\text{feasible}}, \end{aligned} \tag{7.1}$$

where $\mathrm{I_{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}]$ is CSQMI as defined in Ch. 6, $\mathcal{U}_{\text{feasible}}$ is the set of valid controls, $\mathcal{X}_{\text{feasible}}$ is the set states the robot can be in, and $D(\mathbf{u}_{\boldsymbol{\tau}})$ is how long it takes to execute all of the controls. As before, maximizing the rate of information gain is preferable to purely maximizing information, as it enables the robot to compare the value of actions over different time and length scales.

7.2 Approach

We propose a two stage planning approach. We construct a candidate set of trajectories by supplementing global plans with local motion primitives generated by control sampling. We then choose a trajectory that maximizes the objective given by (7.1). In doing so, we combine the benefits of both global planning and local exploration using primitives, as shown in Fig. 7.1. In the second stage, we optimize the chosen trajectory using gradient-based trajectory optimization over multiple time steps to maximize the CSQMI objective while satisfying the robot's motion model. The controls generated by the trajectory optimization routine are fed to the low level motion controller.

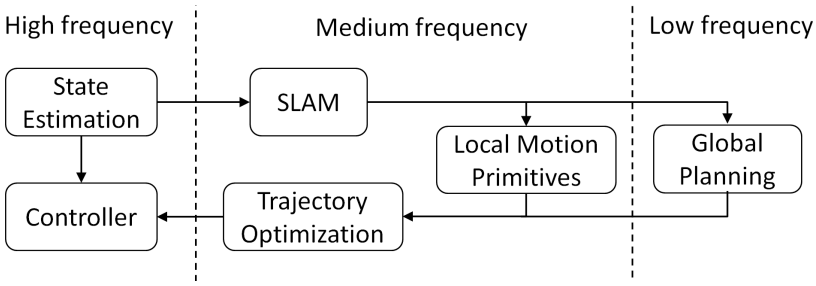


Figure 7.3: System architecture. We generate a candidate set of trajectories by combining global planning and local motion primitives and then select the one that maximizes the CSQMI objective. This selection is improved using trajectory optimization and then executed by the robot. Different system components run asynchronously at different frequencies.

A schematic of our system architecture is shown in Fig. 7.3.

7.2.1 Combining Global Planning and Local Motion Primitives

Global Planning: The global planner's task is to generate a set of paths from the robot's current location whose rate of information gain is high. These paths should extend to every portion of the known map, so that the robot can consider the utility of traveling far away from its current location. To generate these paths, we use the same algorithm for planning shortest paths to destinations where frontiers can be observed (Sect. 6.4), as illustrated in Fig. 7.4.

While the global planner is useful, it has the following shortcomings: 1) generating paths throughout the environment takes a long time; 2) the paths it generates are coarse and may not satisfy the robot's dynamics; and 3) even when paths can be followed, they are frequently not smooth, resulting in jagged robot motion [114].

Local motion primitives: The goal of local motion primitives is to quickly generate short trajectories. This is primarily useful when the robot is near an uncertain part of the map and constantly getting new information that affects where it should go (Fig. 7.1a). As it plans throughout the entire environment, in these cases, the global planner generates paths too slowly.

We evaluated the benefits of two types of control sampling methods: 1) lattice planners [78] that generate dynamically feasible trajectories using a fixed library of motion

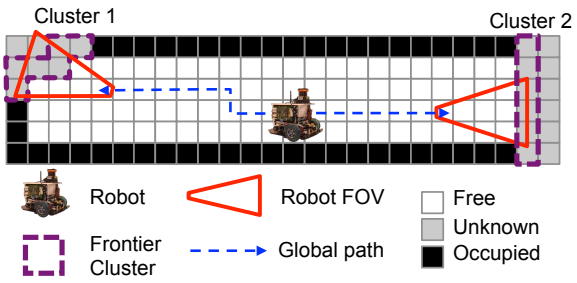


Figure 7.4: Global Planning. We generate global plans by 1) clustering frontier voxels and 2) planning paths to destinations that can view them. When a robot can view a cluster from multiple places, we select the one that can be reached the fastest.

primitives that are chained together to plan over multiple timesteps; and 2) randomized planners that generates trajectories by randomly sampling from the robot's control space over the local time horizon. In both cases, if the motion model predicts that the robot might enter an obstacle or unknown space in the environment, the set of controls is rejected. To specify "local," we heuristically generate motions whose total length is comparable to the maximum range of the sensor.

7.2.2 Trajectory Optimization for Refinement

Neither the global plans nor local motion primitives are locally optimal with respect to the rate of information gain as they do not consider the full control input space of the robot. To address this issue, we use gradient-based trajectory optimization that can refine a trajectory in the continuous control space. Ideally, we would like to optimize all the trajectories within the candidate set and select the one that maximizes the objective. However, optimizing all the candidate trajectories would be computationally expensive due to all the gradient computations involved. Instead, we first select the best trajectory that maximizes the CSQMI objective (7.1) and use that to initialize the optimization.

Gradient of Information: We consider gradient-based trajectory optimization methods. Because our underlying occupancy grid map representation and beam model for the RGB-D sensor are discrete, it is not immediately obvious that a meaningful gradient exists. Prior work [63] showed that the gradient of information of a single measurement can be calculated for a 2D mapping scenario if one assumes that individual beams are independent

and the sensor has a wide field of view. One important observation is that the reward surface is smoother if the position of the robot is restricted to the center of the grid cells. We extend this work by considering the gradient of information of multiple measurements with respect to the robot's full state and control inputs, even when beams are modeled as dependent.

To evaluate gradients, we view the information reward surface as a function of discrete robot positions along the occupancy grid and numerically evaluate gradients using finite differences. Note that the grid cell resolution is often small (e.g., 0.1 m) and that the robot's orientation and control inputs are not restricted to a discrete set.

Sequential Quadratic Programming (SQP): We use SQP [89] to locally optimize the non-convex, constrained optimization problem to find a set of controls over a finite horizon. Strictly speaking, SQP requires that the objective has a Lipschitz continuous second derivative to guarantee a quadratic convergence rate to a local optimum. We cannot formally do this due to nature of the discrete occupancy grid and the sensor model. However, prior work [1, 112, 66] has shown that it is possible to perform gradient ascent on such objectives even in the absence of theoretical guarantees.

To optimize a trajectory, we adapt (7.1) to a form that can be solved using SQP. In particular, we assume a discrete-time robot motion model with a fixed number of time steps, making the duration penalty a constant that can be removed from the objective. We also encourage the robot's states to be feasible by rewarding being sufficiently far away from obstacles:

$$\begin{aligned} \max_{\mathbf{x}_\tau,\ \mathbf{u}_\tau} \quad & \mathrm{I}_{\mathrm{CS}}[\mathbf{m}; \mathbf{z}_{\boldsymbol{\tau}} \mid \mathbf{x}_{\boldsymbol{\tau}}] + \alpha \sum_{i=t+1}^{t+T} \min\left(d_{\max}, \mathrm{dist}(\mathbf{x}_i, \mathbf{m})\right) \\ \text{s. t.} \quad & \mathbf{x}_{i+1} = f(\mathbf{x}_i, \mathbf{u}_i), \\ \forall t<i<t+T-1 \quad & \mathbf{x}_i \in \mathcal{X}_{\text{feasible}}, \mathbf{u}_i \in \mathcal{U}_{\text{feasible}}, \end{aligned} \tag{7.2}$$

where $\mathrm{dist}(\mathbf{x}, \mathbf{m}) \geq 0$ is the minimum unsigned distance from $\mathbf{x}$ to any obstacle in the map, $d_{\max}$ is a parameter that limits the distance reward, and $\alpha \geq 0$ is a weighting parameter

that balances the objective. After a parameter search with logged data, we obtained good results with $\alpha = 20$ and $d_{\max} = 0.5$ m. To support constant time distance queries, before the optimization starts, we construct the Euclidean Distance Transform [37] for the entire 3D map. Note that the time horizon can also be included as an optimization variable.

In our implementation, the innermost QP solver was generated by a numerical optimization code framework that generates QPs specialized for convex multistage problems such as trajectory optimization [34]. Even though it is possible to compute the entire Hessian, it is computationally very expensive to do. We used the symmetric rank 1 (SR1) update method to update the Hessian using the computed gradients [89].

Caching Information: Numerically calculating the objective's gradient is computationally expensive as each evaluation requires calculating the information gain resulting from all poses. The complexity of calculating information scales linearly in the number of poses in the trajectory, while the complexity of calculating information from all beams in a pose is determined by the time it takes to evaluate the information of individual beams, t_{Info}, and the time it takes to find a subset of beams that are independent t_{Indep} [21]. With T poses per trajectory, evaluating the gradient via central differences takes $O(T^2(t_{\text{Info}}+t_{\text{Indep}}))$ time.

Fortunately, we can calculate the gradient faster by caching redundant computations. Calculating each partial derivative of the gradient via central differences involves perturbing an individual optimization variable. Because we include the poses in the optimization, each perturbation of $\mathbf{x}$ or $\mathbf{u}$ only affects a pose at a single time step. Consequently, many of the information calculations across different perturbations are redundant and can be cached. Although this process is complicated by the fact that measurements are dependent – meaning information does not decompose into a sum over the information at each pose – we can cache the information gain from individual beams and the cells that the beams pass through using raycasting. This reduces the complexity of evaluating the gradient to $O(Tt_{\text{Info}} + T^2t_{\text{Indep}})$ which is a substantial speedup as $t_{\text{Info}} \gg t_{\text{Indep}}$. In experiments optimizing over 5 time steps, produces a 4.0× speedup.

7.3 Experiments

There are four primary questions that we seek to answer with our experiments— **Q1**: How much information gain is obtained by optimizing local motion primitives? **Q2**: Which scenarios do local motion primitives provide the most benefit in? **Q3**: How does our approach compare to previous work and a human teleoperator with respect to speed, distance traveled, and map completion? **Q4**: How well does trajectory optimization scale to high-dimensional systems?

We designed simulation and real-world experiments with a ground robot and simulations with an aerial robot to answer these questions.

7.3.1 Platforms and System Details

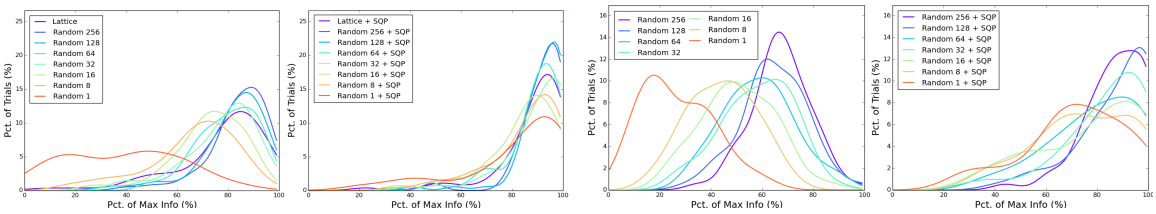
Ground Robot: We conducted experiments using a differential drive ground robot equipped with a 2D Hokuyo UTM-30LX laser range finder and an Asus Xtion Pro RGB-D camera. The robot's computer was an Intel i5 processor with 8 GB of RAM, enabling it to run our entire approach on-board.

We parameterized the robot's state $\mathbf{x}$ as a 3D vector that consists of the robot's 2D position and orientation. The control input $\mathbf{u}$ is a 2D vector of its left and right wheel speeds:

$$\mathbf{x} = \begin{bmatrix} x, & y, & \theta \end{bmatrix}^\intercal, \quad \mathbf{u} = \begin{bmatrix} v^l, & v^r \end{bmatrix}^\intercal$$
$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \\ \theta_{i+1} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \\ \theta_i \end{bmatrix} + \begin{bmatrix} \frac{1}{2}(v_i^l + v_i^r)\cos(\theta_i) \\ \frac{1}{2}(v_i^l + v_i^r)\sin(\theta_i) \\ (v_i^r - v_i^l)/\ell \end{bmatrix}, \tag{7.3}$$

where ℓ is the wheel separation. For trajectory optimization, we limited the wheel speeds to be at most 0.5 m/s and imposed the motion model's nonlinear equality constraints.

Aerial Robot: We simulated a quadrotor as an AscTec Pelican that is equipped with a UTM-30LX laser range finder and an Asus Xtion Pro RGB-D camera using Gazebo [70].



(a) **Ground Robot** (b) **Aerial Robot**

Figure 7.5: Information gain performance on recorded data. Over 157 situations for the ground robot and 63 situations for the aerial robot, we evaluate the optimality of the initial trajectory input into the trajectory optimization for various initializing methods (left column). We then evaluate the optimality of the trajectory output by trajectory optimization (right column). We see dominant modes shift towards 100% optimality when incorporating trajectory optimization, showing trajectory optimization improves overall information gain.

We parameterized the quadrotor's state $\mathbf{x}$ as an 8D vector consisting of its 3D position, 3D velocity, yaw, and yaw velocity. The control input $\mathbf{u}$ is a 4D vector of the linear accelerations and yaw acceleration. We assumed the quadrotor does not fly aggressively and that its state evolved with a constant acceleration model:

$$\mathbf{x} = \begin{bmatrix} x, & y, & z, & \theta, & \dot{x}, & \dot{y}, & \dot{z}, & \dot{\theta} \end{bmatrix}^\intercal$$

$$\mathbf{u} = \begin{bmatrix} \ddot{x}, & \ddot{y}, & \ddot{z}, & \ddot{\theta} \end{bmatrix}^\intercal$$

$$\mathbf{x}_{i+1} = \begin{bmatrix} I_4 & \Delta t \cdot I_4 \\ 0 & I_4 \end{bmatrix} \mathbf{x}_i + \begin{bmatrix} \frac{1}{2}\Delta t^2 \cdot I_4 \\ \Delta t \cdot I_4 \end{bmatrix} \mathbf{u}_i$$

where Δt is a user-defined time discretization value. For trajectory optimization, we bounded the linear acceleration between [-0.5,0.5] m/s^2, yaw acceleration between [$-\pi/2$,$\pi/2$] rad/s^2, the velocity in each dimension between [-1,1] m/s, and the yaw between [$-\pi/2$,$\pi/2$] rad/s^2. We also imposed linear constraints on the robot's motion model.

RGB-D Sensor: Both the ground and the aerial robot were equipped with a RGB-D sensor that was modeled after the Asus Xtion Pro sensor. We assume it has a minimum range of $z_{\text{min}} = 0.5$ m, maximum range of $z_{\text{max}} = 4.0$ m, and its noise is $\sigma = 0.03$ m. To predict future measurements and evaluate CSQMI, we discretized the 58° horizontal field of view and 48° vertical field of view into 20 separate values each, resulting in 400 separate

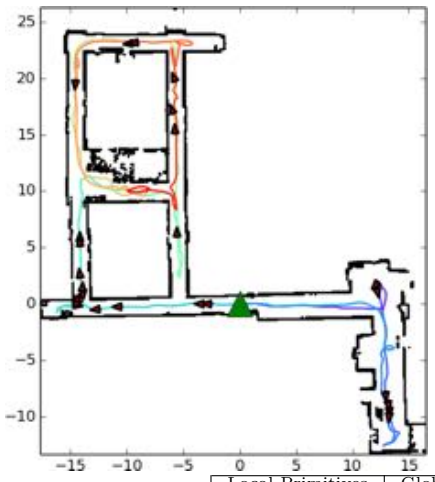
beams.

Occupancy Grid: For both platforms, the 3D occupancy grid was constructed with a resolution of 0.1 m using data from the RGB-D sensor. In experiments, the ground robot obtained position estimates from a 2D laser based SLAM system, while in simulation the quadrotor obtained a noisy pose estimate from the simulator.

7.3.2 Evaluating Trajectory Optimization

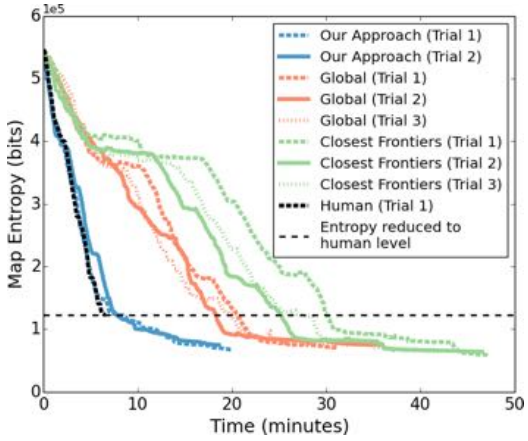
To quantify how much trajectory optimization can improve the information gain of local motion primitives (**Q1**), we looked at its performance in a variety of situations. To create these different situations, we sampled different maps and poses for the robots from our previous real-world experimental data in Ch. 6. For each situation, we generated a fixed number of motion primitives using a lattice planner or random sampling (Sect. 7.2.1). The motion primitive with the highest information was then optimized. Because information gain can vary substantially across different maps and poses, we normalize the information gain of each approach by the maximum information gain achieved by any approach with the same initial conditions. Results for the ground air and robot are shown in Fig. 7.5a and Fig. 7.5b. As expected, using larger number of motion primitives improves the performance of the unoptimized trajectories. However, regardless of the number of motion primitives used to find an initialization, trajectory optimization consistently produces trajectories whose information gain is close to the maximum achieved, demonstrating that it can substantially improve upon the search based strategies. Comparing the trajectory optimization information gain for the ground robot (Fig. 7.5a) with the aerial robot (Fig. 7.5b), we see the benefit of using trajectory optimization is greater in higher-dimensional systems (**Q4**).

We also examined the average computation time over all situations for each approach. Results are shown in Fig. 7.7a for the ground robot and Fig. 7.7b for the aerial robot. While evaluating a small number of motion primitives is fast, these trajectories tend to have small gains in information. However, using these trajectories to initialize optimization yields finds trajectories that are higher in information, with less time than searching over large numbers



	Local Primitives	Global Plan
Traj. Opt. Initialization (% of times)	66%	34%

(a) **Final map (2D projection), exploration path, and poses where trajectory optimization performed best**: The green triangle is the robot's start position. The colored path is its exploration path. The red arrows show the 25 poses where trajectory optimization provided the largest gain in information over the initializing trajectory.



	Averages when entropy reduced to human level		
	Time (min)	Distance (m)	Speed (m/s)
Human	6:36 (24.0%)	153 (86%)	0.41 (3.2×)
Our Approach	8:15 (29.9%)	156 (88%)	0.34 (2.6×)
Global	19:19 (71.2%)	156 (88%)	0.15 (1.2×)
Closest Frontiers	27:37 (100.0%)	178 (100%)	0.13 (1×)

(b) **Comparisons to previous work**: Comparing our approach with 1) Human teleoperator who knows the map apriori, 2) Information based planner that only plans globally [20], and 3) Closest frontiers [142].

Figure 7.6: Ground robot experiments. Mapping a real-world office environment. Our approach reduces the map's uncertainty faster than previous approaches and only does slightly worse than a human operator with a priori knowledge of the environment.

of motion primitives.

7.3.3 Mapping Experiments With a Ground Robot

We tested our approach by using a ground robot to map a 40 m × 35 m × 2 m portion of an office environment over two trials. For the local motion primitives, we sampled 32 random trajectories. Results and comparisons are in Fig. 7.6. The final 3D map from one trial is shown in Fig. 7.2a.

To assess the overall performance of our system (**Q3**), we compare it to three other approaches. The first is "Global", a strategy that evaluates information along paths to frontiers and does not use local motion primitives or trajectory optimization, while the second is a closest frontiers planner [142]. In order to obtain an approximate lower bound on achievable performance, we also compare to a human teleoperator who knows the entire

	Random 1	Random 8	Random 16	Random 32	Random 64	Random 128	Random 256	Lattice
Avg. Initial Time (s)	0.00 ± 0.0	0.04 ± 0.0	0.07 ± 0.1	0.15 ± 0.1	0.29 ± 0.3	0.58 ± 0.6	1.17 ± 1.2	0.26 ± 0.3
Avg. SQP Time (s)	0.78 ± 0.8	0.55 ± 0.6	0.43 ± 0.4	0.42 ± 0.4	0.40 ± 0.4	0.37 ± 0.4	0.34 ± 0.3	0.42 ± 0.4
Avg. Total Time (s)	0.79 ± 0.8	0.59 ± 0.6	0.51 ± 0.5	0.56 ± 0.6	0.69 ± 0.7	0.95 ± 1.0	1.51 ± 1.5	0.69 ± 0.7

(a) **Ground Robot**

	Random 1	Random 8	Random 16	Random 32	Random 64	Random 128	Random 256
Avg. Initial Time (s)	0.00 ± 0.0	0.02 ± 0.0	0.05 ± 0.0	0.09 ± 0.1	0.18 ± 0.2	0.37 ± 0.4	0.75 ± 0.7
Avg. SQP Time (s)	0.84 ± 0.8	0.61 ± 0.6	0.62 ± 0.6	0.60 ± 0.6	0.54 ± 0.5	0.56 ± 0.6	0.62 ± 0.6
Avg. Total Time (s)	0.84 ± 0.8	0.64 ± 0.6	0.67 ± 0.7	0.69 ± 0.7	0.73 ± 0.7	0.93 ± 0.9	1.37 ± 1.4

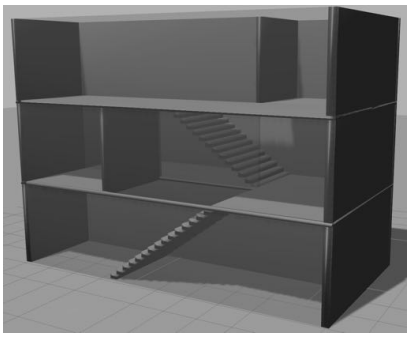
(b) **Aerial Robot**

Figure 7.7: Planning times on recorded data. Comparing the initialization, trajectory optimization, and total times for different initialization methods using the same data as in Fig. 7.5. The double line show when sampling controls becomes more computationally expensive than trajectory optimization.

environment a priori and manually drives the robot to gather measurements. For comparisons with the first two approaches, we use data from our previous experiments in Ch. 6, which were obtained using the same robot.

To measure the speed of each approach, we look at the time it takes to obtain a low uncertainty map measured using Shannon's entropy. The entropy of a map with $|\mathbf{m}|$ cells is $\sum_{i=1}^{|\mathbf{m}|} -f_i \log_2 f_i - (1-f_i)\log_2(1-f_i)$ bits where f_i is the probability that the i^{th} cell is free [27]. Fig. 7.6b shows the map's entropy over time for each approach. The human operator decreased the uncertainty of the map the fastest. However, our approach did not take substantially longer to achieve a similarly certain map, despite the robot's absence of prior knowledge about the environment. Our approach was also substantially faster than the other autonomous approaches. Note that the strategies that use information and the human all travel essentially the same distance. This suggests the difference in time is primarily due to how quickly each approach can determine an informative trajectory to follow.

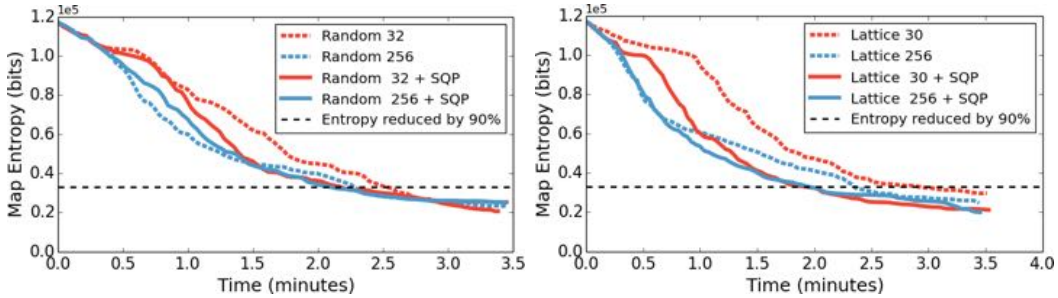
It is interesting that the human operator stopped once they believed they had obtained a low uncertainty map and that all autonomous approaches continue reducing the map's entropy beyond this point, as they continue until no frontiers are left. However, the final maps are qualitatively hard to differentiate, suggesting a better termination condition is needed.



(a) Stairwell with walls removed

	Averages when entropy reduced by 90%		
	Time (min)	Distance (m)	Speed (m/s)
Random 32	2:32 (87%)	36.2 (88%)	0.21
Random 256	2:14 (77%)	30.4 (73%)	0.20
Lattice 30	2:54 (100%)	41.4 (100%)	0.18
Lattice 256	2:38 (90%)	37.5 (91%)	0.18
Random 32 + SQP	2:11 (74%)	27.0 (65%)	0.19
Random 256 + SQP	2:09 (74%)	26.9 (65%)	0.19
Lattice 30 + SQP	1:57 (67%)	26.5 (64%)	0.19
Lattice 256 + SQP	1:53 (66%)	26.4 (64%)	0.20

(b) Comparison of distance traveled and time to reduce entropy



(c) Map entropy over time using random (left) and lattice (right) local motions with and without trajectory optimization

Figure 7.8: Aerial robot stairwell simulations. Trajectory optimization enables the quadrotor to map the environment more efficiently.

Fig. 7.6a shows a 2D projection of the final map, the path the robot followed, and poses where trajectory optimization provided the most improvement in information. These poses are predominantly in open spaces, junctions, approaching corners, and dead-ends (**Q2**). This agrees with intuition as these areas are more complicated than straight narrow hallways where finding an informative trajectory is simpler.

7.3.4 Mapping With an Aerial Robot

To further study the impact of trajectory optimization on the information of local primitives (**Q1**) and study how well our approach scales to robots with higher dimensional state (**Q4**), we evaluated it by having a quadrotor map a 10 m × 5 m × 7.5 m 3-story stairwell in simulation Fig. 7.8a. The final 3D map from one trial is shown in Fig. 7.2b.

We are interested in the effect of different motion primitives and whether the improvement in trajectory optimization results in different *overall* performance compared to control

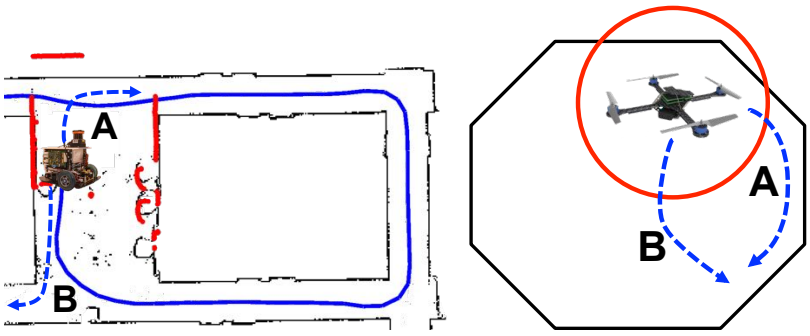
policies that use motion primitives and global plans. For these simulations, we generate local plans over 5 time steps and again use a random or lattice planner to generate motion primitives. The random planner repeatedly sampled from the robot's 4D control space to generate trajectories. The lattice planner generated straightline constant velocity trajectories whose directions were determined by discretizing polar and azimuthal angles in spherical coordinates. We chose this as even coarse discretizations of the control space result in thousands of trajectories, resulting in slow overall performance.

Fig. 7.8c shows the map's entropy over time using different numbers of initial trajectories averaged over 5 trials for each approach. The lattice planner generated 30 trajectories using 6 evenly spaced polar angles and 5 evenly spaced azimuthal angles. To generate 256 primitives, it discretized both angles into 16 distinct values. Overall, using trajectory optimization resulted in the entropy of the map decreasing faster (note that the simulator is asynchronous, so the reported difference in time includes computation time). This difference is particularly noticeable for the lattice planner, which had difficulty rounding corners due the limited number of directions it could travel.

Fig. 7.8b contains additional statistics about distance traveled and time to reduce entropy. We see that by using trajectory optimization – regardless of initialization strategy – the robot reduced most of the map's entropy faster, while traveling a shorter distance. This is due to the trajectory optimization's ability to find feasible trajectories that exploit the mobility of the quadrotor and execute informative turning actions (see the video submission for additional details).

7.4 Limitations and Discussion

Our work has three limitations that we briefly discuss. First, we assume that the robot is capable of localizing itself reliably using a SLAM system. Fig. 7.9 illustrates this issue. Fig. 7.9a shows a 2D occupancy grid which we created by artificially lowering the maximum range of the ground robot's UTM-30LX to 4.0 m and manually driving it along a trajectory (solid blue line). The most recent laser scan (red dots) is not aligned with the



(a) (b)

Figure 7.9: Limitations. In (a) and (b) executing action A is preferable over action B as the robot's uncertainty will be much lower. However, our approach only considers the map's uncertainty and selects action B in both cases.

map, showing accumulated position error. The robot should follow action A and close the loop to improve its state estimate and the map's consistency. However, our approach only considers the map's uncertainty and would select action B as the bottom of the map is more uncertain. Fig. 7.9b shows a different scenario where a robot is equipped with a short-range omnidirectional sensor. Our approach would select action B, as the robot will observe more of the map. However, during parts of this trajectory, all features will be outside the robot's FOV (red circle), precluding accurate localization; consequently action A is preferable. One avenue for future work is to extend our approach to active SLAM by accounting for robot state uncertainty [10, 129, 136, 69, 60, 98].

While modeling the robot's uncertainty in the control policy can be important, it is not always necessary to ensure good performance. For example, in our ground robot experiments, the accumulated state error was small. Although ground truth is not available, the final maps were consistent, meaning the corrections from the SLAM system provide insight into the robot's accumulated error. Over 8 experiments, the mean correction to the robot's 2D position and orientation was 0.28 m and 1.35°. These small corrections are due to the robot's accurate laser odometry [94], and demonstrate that it is possible to build good maps even when a robot does not explicitly take actions to reduce its own uncertainty.

Our work's second limitation is that we do not employ a principled termination condi-

tion, as is evident from the long flat tails of the entropy curves (Fig. 7.6b and Fig. 7.8c). Our experiments with the human operator provide insight into this process. It may be possible to take inspiration from prior work that uses reinforcement learning for active mapping [71] to determine appropriate termination conditions.

Finally, we use trajectory optimization for maximizing the CSQMI objective over a discrete occupancy grid. Apart from the lack of theoretical optimality guarantees, the optimization's performance also heavily depends on the initial trajectory [97].

The quality of paths produced by trajectory optimization using SQP also depends heavily on user-chosen parameters. To choose these SQP parameters, we used experiment data from prior work and evaluated the information gain using SQP over a grid of parameter values. We defined the quality of a parameter set as the mean information gain improvement using SQP. We then chose the parameter set with the highest improvement and used these parameters for real-world experiments.

7.5 Conclusion

In this work, we present an information-theoretic planning approach for dense 3D mapping of unknown environments with mobile robots. In contrast to prior work that either use only global plans or local motion primitives, our approach combines the benefits of the two by considering a candidate set of trajectories consisting of global plans and local motion primitives. We select the most promising trajectory from this candidate set and further optimize it to increase information gain while satisfying the robot's motion model. Through a series of simulations and real world experiments, we study the benefits and drawbacks of our system and compare its end to end performance with several other approaches. Our results indicate that the proposed approach has the potential to increase the efficiency of 3D mapping, particularly for robots with limited battery life that are equipped with noisy sensors that have short sensing ranges and limited fields of view.

Chapter 8

Heterogeneous Robot Routing for Cooperative Mapping

In Ch. 6 and Ch. 7 we developed a CSQMI based control policy that enables individual robots to build dense 3D maps. In this chapter, we extend this work to situations where air and ground robots must cooperatively map an environment.

While the information-theoretic policies in the preceding chapters perform well, they have some shortcomings. In particular, they can only be computed in a reasonable amount of time when they are restricted to a finite time horizon. This prevents the robots from determining long-term plans, which can lead to poor performance as illustrated in Fig. 8.1. The fundamental issue is that by planning over a finite time horizon the robot does not consider that it must eventually observe all areas of the map, biasing it towards actions that maximally reduce uncertainty over the time horizon which it is planning. This can result in the robot traveling far away from its current location even when the local area is not fully mapped, necessitating a long trip back later in the mapping process. This issue is partially addressed by maximizing the rate of information gain (Ch. 6 and Ch. 7), as it causes robots to prefer efficient actions, but this does not fully solve the problem. This issue is amplified when considering teams of heterogeneous robots (e.g., aerial and ground robots), as some areas may only be observable by one robot, while others can be visited by

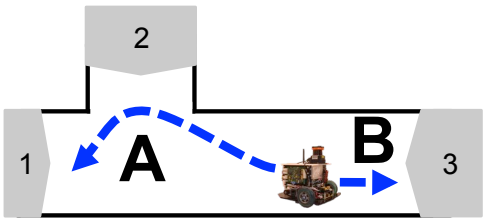


Figure 8.1: Limitations of planning over a finite time horizon. If a robot doesn't account for the fact that it must eventually view areas 1, 2, and 3, it may not plan time efficient paths. In this map, the fastest way to view all areas is to view area 3, followed by areas 2 and 1. However, when maximizing information over a finite time horizon and deciding between actions A and B, the robot may select A, as it will likely reduce the map's uncertainty more.

both. For example, if an area can only be observed by one robot, it may be beneficial to have it travel there immediately, even if it can reduce the uncertainty of the map more in other places.

While planning over long time horizons can improve the time to build a complete map, there is another issue facing our approach: the environment is only partially known. As soon as the team starts moving they might discover new areas that need to be observed, potentially invalidating previously generated paths. Unless the robots are capable of having some type of prior knowledge about the environment, or are able to accurately predict what unobserved parts of the environment will contain, this issue is unavoidable.

Consequently, instead of focusing on an algorithm that spends a large amount of time constructing detailed plans for each robot (e.g., a multi-agent watchman route problem [23]), we attempt to quickly plan coarse high-level paths. To do this, we formulate a problem where, given a set of features to observe in the current map, the robots' speed, and sensor models, we must find paths for each robot such that every feature gets observed and the maximum traversal time of any robot is minimized. While this is an NP-hard combinatorial optimization problem, the primary contribution of this chapter is showing that problem instances can be formulated and solved in a reasonable amount of time. To do this, we decompose the problem into two parts. First, we develop an approximation algorithm for finding shortest paths for multiple robots to visit a set of destinations. Second, we generate destinations that effectively summarize where each robot can go. By appropriately creating

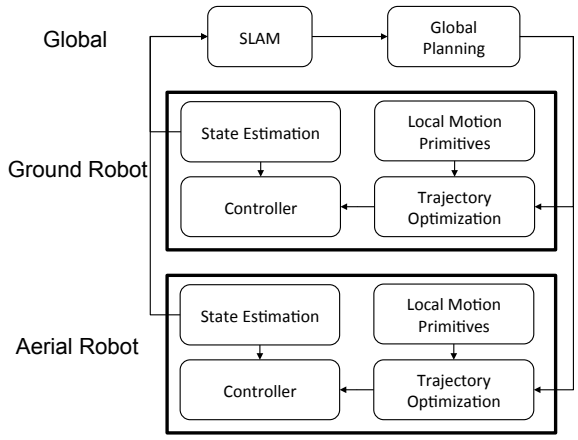


Figure 8.2: System architecture for a 2 robot team. By generating long paths for both robots that collectively observe all unobserved areas, the team can account for their mobility differences and map environments faster. Using the techniques in Ch. 7, individual robots locally refine and improve the informativeness of the paths they follow.

this set of destinations for each robot, we can find paths much faster.

While the paths generated by this approach will be coarse, they should enable the team to jointly determine which robot should go where, and can later be refined online using the information-theoretic policy from Ch. 7. While Ch. 7 focused on quickly generating and optimizing informative trajectories over short time and length scales for an individual robot, the focus of this chapter is on generating paths over long time and length scales to coordinate multiple heterogeneous robots. Together, these components form a hierarchy of planning and control systems for active mapping. Fig. 8.2 provides a centralized system architecture for a heterogeneous team comprised of a ground and aerial robot.

Many of the algorithms in this chapter scale poorly with the size of the team. Although we describe each algorithm in this chapter in terms of an arbitrary number of robots, our intended application is for reasonably teams (e.g., 2 to 3 robots).

8.1 Heterogeneous Robot Routing in Minimal Time

In this section, we formally describe the Heterogeneous Robot Routing Problem (HRRP) a method for solving it, and how to create an instance of the problem that is appropriate for generating paths for active mapping. Informally, the HRRP is to find paths for a team

of heterogeneous robots to a series of destinations such that each destination is visited at most once by each robot and the time to execute all paths is minimized.

This problem is closely related to the Heterogeneous, Multiple Vehicle, Heterogeneous, Multiple Depot, Multiple Traveling Salesman Problem (HMDMTSP) which Oberlin et al. [91] studied. The only difference is that we want to minimize the maximum time any robot spends traveling, while they minimize the sum of the time all robots spend traveling. However, in Sect. 8.1.1, we show that their approach can be used as an approximation algorithm for the problem we wish to solve. This is beneficial as Oberlin et al. developed an efficient reduction from the HMDMTSP to the Asymmetric Traveling Salesman Problem (ATSP) which can then be transformed to the Traveling Salesman Problem (TSP) [57].

Transforming a HRRP instance to a TSP instance may not sound like progress, as the TSP is a well known NP-hard problem. However, this transformation is quite useful, as a large amount of research and engineering effort has been devoted to solving TSPs. State of the art TSP solvers, like Concorde [3], can obtain optimal solutions to graphs with a few hundred vertices in seconds, even on a laptop computer. As we describe later, the cost of the corresponding solutions in ATSP, TSP, and HRRP are all the same, meaning that any approximation algorithm for ATSP or TSP, will yield an equivalent approximation guarantee for HRRP.

8.1.1 HRRP Formulation and Assumptions

Suppose we are given m distinct robots $R = \{r_1, \ldots, r_m\}$ and a set of n vertices $V = \{v_1, v_2, \ldots, v_n\}$, each of which must be visited by exactly one robot. Let S be the set of locations that each robot starts at $S = \{s_1, \ldots, s_m\}$ and $V_r \subseteq V$ be the subset of vertices that robot r can travel to. We assume that each destination can be visited by at least one robot, but the starting locations and visitable subsets need not be the same between robots. Let $E_r = \{(v_i, v_j) : v_i, v_j \in V_r \cup \{s_r\}\}$ be the set of edges that robot r can traverse and $c_r(v_i, v_j)$ be the non-negative cost of traversing that edge. The cost of traversing an edge need not be symmetric (i.e., $c_r(v_i, v_j) \neq c_r(v_j, v_i)$) and does not have to be the same across robots that can both visit it (i.e., $c_{r_k}(v_i, v_j) \neq c_{r_\ell}(v_i, v_j)$). A path for robot r is an

ordered sequence of vertices that begins at the robot's starting vertex, s_r, and goes to other vertices that the robot can visit (i.e., elements of V_r). We will denote a path for robot r as $P_r = [s_r, v_i, \ldots, v_j]$ and define its cost, $c_r(P_r)$, as the sum of edge weights between its constituent vertices.

The Heterogeneous Routing Robots Problem (HRRP) is to find a set of paths for each robot such that the maximum cost of any path is minimized and every vertex in V is visited by at most one robot:

$$\begin{aligned} \min_{P_1,\ldots,P_m} \quad & \max_r c_r(P_r) \\ \text{subject to} \quad & v \text{ is in exactly one of } P_r \ \forall v \in V \end{aligned} \tag{8.1}$$

Note that none of the robots are required to return to their starting position; the HRRP is over paths and not closed walks. This problem is NP-hard as it contains the weighted Hamiltonian path problem as a special instance. To see this, consider the situation where there is exactly one robot. The objective in (8.1) is then exactly the same as finding the minimum weight path that visits all vertices.

One major source of difficulty for this problem is that the objective is to minimize the maximum time. Unfortunately, we are unaware of any direct reduction to a single agent TSP. However, by relaxing the problem to minimizing the sum of the costs, we obtain the HMDMTSP:

$$\begin{aligned} \min_{P_1,\ldots,P_m} \quad & \sum_{r=1}^{m} c_r(P_r) \\ \text{subject to} \quad & v \text{ is in exactly one of } P_r \ \forall v \in V \end{aligned} \tag{8.2}$$

Technically, the HMDMTSP as defined by Oberlin et al. [91] is over tours for each robot and not paths (i.e., in the HMDMTSP each robot must return to its starting location, but in the HRRP they do not), but a small change that we describe in Sect. 8.1.2 resolves this issue.

Minimizing the sum of costs is less appealing when the objective measures time, because

robots can move simultaneously [135]. However, any approximate solution to HMDMTSP approximates the HRRP:

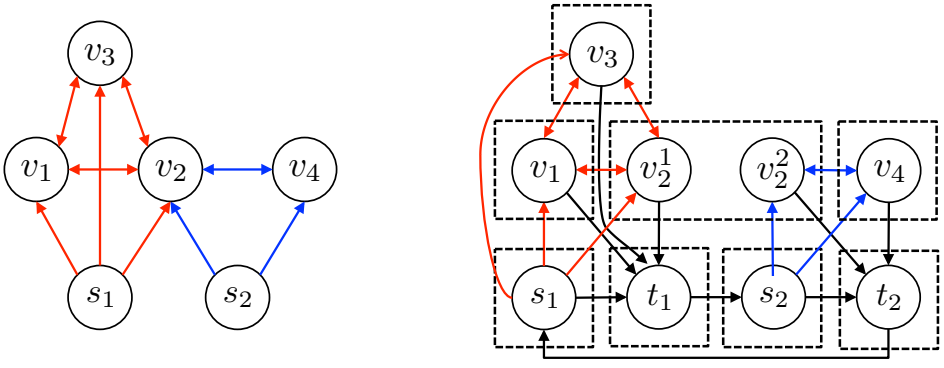
Theorem 4. *k-approximations of HMDMTSP* (8.2) *with m robots are km-approximations of HRRP* (8.1)*.*

Proof. Only the objective differs between the two problems, so a solution is feasible for one problem if and only if it is feasible for the other. Let $P = [P_1, \ldots, P_m]$ be a k-approximate solution for HMDMTSP and let P_1 be the path with maximum cost, so that the cost of P in HRRP is $c_1(P_1)$. If the approximation guarantee does not hold, then there must be a different set of feasible paths $P^* = [P_1^*, \ldots, P_m^*]$ such that $km \max_r c_r(P_r^*) < c_1(P_1)$ (i.e., km times the optimal cost in HRRP is strictly lower than the cost of the HMDMTSP solution). However, we know that $\sum_r c_r(P_r) \leq k \sum_r c_r(P_r^*)$ as P is a k-approximation to HMDMTSP. Additionally, $k \sum_r c_r(P_r^*) \leq km \max_r c_r(P_r^*)$ as each element of $c_r(P_r^*)$ is at most the maximum element. Chaining these inequalities implies that $\sum_r c_r(P_r) < c_1(P_1)$ which is a contradiction, establishing the approximation guarantee. □

Thm. 4 is not a particularly strong guarantee as the number of robots grows. However, we can often solve a HMDMTSP instance optimally, and this work is primarily focused on coordinating smaller teams of robots.

8.1.2 Transforming HRRP to TSP

We now describe how to transform an HRRP instance to a TSP instance. The first step is to transform HRRP to HMDMTSP, which doesn't actually require any work, as the only difference between the two problems is the objective. Next, we use the approach developed by Oberlin et al. [91] to transform the HMDMTSP instance to an instance of the Generalized Traveling Salesman Problem (GTSP), which can be transformed to an ATSP [5], which can be transformed into the symmetric TSP [61]. If an HRRP instance has n vertices and m robots, after the full series of transformations, the corresponding TSP will have at most $2m(n + 2)$ vertices. Although we did not develop this chain of transformations, we briefly describe them here for completeness, primarily focusing on the transformation from



(a) HMDMTSP (b) GTSP

Figure 8.3: HMDMTSP to GTSP. (a) An example instance of the HMDMTSP problem with 2 robots at different start locations (s_1 and s_2) and 4 destinations. Red arrows show directed edges for one robot and blue lines show edges for the other robot. The only vertex that can be visited by both robots is v_2. (b) The corresponding GTSP instance for a single agent that starts at s_1. Dashed line boxes show the disjoint sets, colors show where edges came from in the HMDMTSP instance, and all solid black arrows are 0-weight edges. v_2 can be visited by both robots in the HMDMTSP instance and it creates two vertices in the GTSP instance with a set constraint.

HMDMTSP to GTSP.

HMDMTSP to GTSP: This transformation was created by Oberlin et al. [91]. The GTSP is given a set of vertices V with a set of directed and non-negative weighted edges, and a partitioning of these vertices into non-overlapping subsets, $H = \{H_1, \ldots, H_m\}$ such that $V = \bigcup_{i=1}^{m} H_i$ and $H_i \cap H_j = \emptyset$ when $j \neq i$, find a minimal weight tour that visits an element of each element of H exactly once. The main idea behind the transformation from HDMTSP to GTSP is to create a graph that a single agent can traverse, but that maps back to paths that multiple robots can traverse in parallel in the original problem. Fig. 8.3 illustrates this process.

To create the GTSP instance, for each robot, we create a portion of the GTSP graph that corresponds to vertices the robot can travel to. Specifically, for each robot r, create 2 vertices s'_r and t'_r, that represent its starting and terminating location. For every $v \in V_r$ (i.e., each vertex r can visit) in the original HMDMTSP, add a vertex v' to the GTSP, with a directed weighted edge from s'_r to v' and a 0-weight edge from v' to t'_r. In the solution to the GTSP, the sequence of vertices between s'_r and t'_r forms the path that robot r should

follow in the HMDMTSP. Once vertices have been added for each robot, add a 0-weight cycle between all of the start and termination vertices: $[s_1, t_1, s_2, t_2, \ldots, s_r, t_r, s_1]$

Having 0-weight edges from each vertex to the termination vertices is what causes us to compute minimum cost paths. To compute minimum cost cycles (i.e., paths that return to the starting destination), as Oberlin et al. [91] did, we would make the weight of the returning edge equal to the cost between the current vertex and the robot's starting vertex in the HMDMTSP instance.

To complete the transformation, we need to specify H, the disjoint subsets of the vertices in the GTSP. The goal in designing H is to guarantee that after solving the GTSP the corresponding paths for each robot in the HMDMTSP collectively visit each destination exactly once. To start, we add a disjoint subset around each of the starting and terminating vertices. Next, if a vertex can be visited by multiple robots in the HMDMTSP, we add a disjoint subset around the corresponding vertices that were created in the GTSP.

If there are n vertices and m robots in the original problem, there are at most $m(n+2)$ vertices in the GTSP. It is worth noting that this transformation relies heavily on the costs in the HMDMTSP being additive across different robots. If this were not the case, as in the original HRRP problem where the cost is a maximum over all robots, then it is unclear how to reformulate the problem as a single agent GTSP.

GTSP to ATSP: This transformation is due to Behzad and Modarres [5] and the resulting ATSP has the same number of vertices as the GTSP instance.

ATSP to TSP: This transformation is due to Jonker and Volgenant [61] and doubles the number of vertices that the agent must visit. The primary reason for performing this transformation is a pragmatic one: current generic solvers for symmetric TSPs are faster than those for ATSPs [57].

8.1.3 Solution Methods

Given a TSP instance, there are a large number of ways to solve it. For a more thorough discussion of these choices, we refer the interested reader to the summary given by Hornik and Grün [57]. In experiments and simulations with two robots, we obtained optimal

solutions in a reasonable amount of time using the Concorde solver [3]. For larger problem instances, this approach becomes infeasible. However, it is also possible to obtain a 2-approximation to solve the TSP using the Minimum Spanning Tree algorithm if we allow an individual robot to revisit destinations that it has already visited. From Thm. 4, this would result in a $2m$-approximation algorithm for the HRRP.

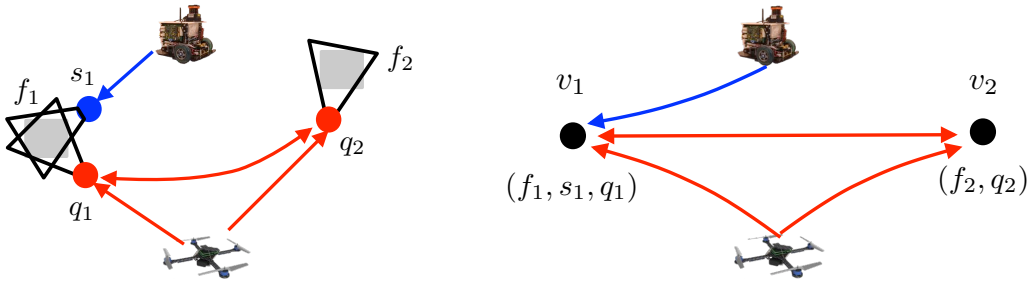
8.2 Generating the HRRP

In this section, we describe a method for generating an HRRP instance whose solution results in good paths for active mapping (Fig. 8.4). The basic approach is to identify frontiers and poses that can observe them as in Ch. 6 and Ch. 7. These destinations serve as a reasonable basis for the destinations for a robot in the HRRP. However, even in relatively small office environments, there may be hundreds of destinations for each robot, and after performing the series of transformation in Sect. 8.1, the resulting TSP could have thousands of vertices, making it difficult to solve quickly. To address this issue, in Sect. 8.2.1 we describe a hierarchical agglomerative clustering approach for grouping destinations that are similar for an individual robot. To determine how destinations across robots should be grouped as a single goal in the HRRP problem, we describe a matching procedure in Sect. 8.2.2 that determines which destinations are effectively viewing the same features.

8.2.1 Clustering Destinations for Individual Robots

We wish to generate a small set of destinations for a robot that accurately capture what features it can observe. The input to this algorithm is a set F of features in the map: $F = \{f_1, f_2, \ldots, f_{|F|}\}$. The output should consist of a set of destinations the robot can travel to and which features it can observe from each one. Although the algorithms described in this section do not require a specific type of feature, they do require a way of testing whether or not a feature is visible from a specific pose. We define the binary function $\text{Vis}_r(x, f)$ which returns true if feature f is visible by robot r at pose x and false otherwise. For features, our implementation uses groups of frontier voxels, similar to Ch. 6 and Ch. 7.

For each feature $f \in F$, we use the Dijkstra based search described in Sect. 6.4 to try



(a) Destinations that view frontiers

(b) Corresponding HRRP instance

Figure 8.4: Creating an HRRP instance. (a) Gray blobs are frontiers, black boxes show the field of view for each robot, and blue and red dots show destinations where the robots can observe a frontier. The quadrotor can observe both frontiers, while the ground robot can only view frontier f_1. (b) Planning paths that view all frontiers can be formulated as an HRRP where each vertex corresponds to a frontier and the edge weights between vertices are the travel times between the associated destinations.

to find a pose x such that $\text{Vis}_r(x, f)$ is true. This process maps each visible feature to a single pose that can view it (e.g., s_1 and q_1 viewing f_1 in Fig. 8.4a). We define D^r as the set of poses generated for robot r by this process. As discussed earlier, D^r could serve as a basis for the HRRP, except that the number of features, $|F|$, is potentially large, making the problem expensive to solve.

Fortunately, interesting features in an environment are often close to each other, and a robot may be able to observe multiple features from a single pose. To see when this is the case, we determine all features that are observable from each pose in D^r. This creates a mapping from reachable poses for the robot to sets of features that it can observe; we update D^r with this information so that $D^r = \{(x_i, F^r(x_i)\}$ where $F^r(x) = \{f : f \in F \cap \text{Vis}_r(x, f))\}$.

To make D^r smaller, we cluster destinations that observe similar sets of features and are not too far from each other. More precisely, we use a greedy agglomerative (i.e., bottom-up) algorithm that starts with a group of clusters and repeatedly merges the two that are "closest" to each other [31]. There are several variations of this approach [51]. To measure the distance between two clusters, we use the Jaccard distance between the sets of feature IDs. Formally, if $d_i = (x_i, F_i)$ and $d_j = (x_j, F_j)$ are destinations in D^r, with $F_i, F_j \subseteq F$, we

define their similarity as:

$$\rho(d_i, d_j) = 1 - \frac{F_i \cap F_j}{F_i \cup F_j} \tag{8.3}$$

The Jaccard distance is a metric on finite sets [76] and measures their similarity on a scale between 0 and 1. It is 0 if and only if the sets are identical, and 1 if and only if the two sets have no elements in common.

We use the single-linkage criterion to measure the distance between clusters. Given two clusters $D_k, D_\ell \subseteq D^r$ we define the distance between them as $\min\{\rho(d_i, d_j) : d_i \in D_k, d_j \in D_\ell\}$. A well known problem with the single-linkage criterion is that it tends to "chain" several points together into a single cluster, even when most of them are far apart from each other. To mitigate this issue, we do not merge clusters when the maximum distance (or similarly time) the robot must travel between any two destinations exceeds a user-specified threshold.

The agglomerative clustering algorithm produces several clusters each comprised of destinations with their associated features. To reduce the complexity of this data, we summarize each cluster with a single representative destination for the robot along with all features that are viewed at any destination in the cluster. The time to travel between clusters can then be determined by the time to travel between the representative destinations. This time serves as an approximation of the time to view all features in the two clusters, since within a cluster a robot can travel to all of the destinations – and truly observe all of the features – in a bounded amount of time. In our implementation, we selected a cluster's representative destination by finding the destination that was closest to the average of all destinations in the cluster.

8.2.2 Matching Destinations Between Robots

At this point, for each robot, we have a set of destinations with their associated features. By construction, each robot can travel between all of the destinations. However, in order to fully specify the HRRP, we need to determine which destinations between different robots are viewing equivalent features. For example, in Fig. 8.4a, both robots can observe feature f_1,

Algorithm 4: Construct an HRRP instance

Input: Destinations $D^r = \{(x_i^r, F_i^r)\}$ for each robot r.
Output: An HRRP instance.

1: Construct a undirected unweighted graph G whose vertices correspond to a single destination for one robot. Add an edge between vertices if they correspond to destinations from different robots and their overlap coefficient is sufficiently large.
2: Find all maximal cliques in G and add each one as a destination in the HRRP.
3: For each pair of destinations in the HRRP instance that robot r belongs to, add an edge whose weight is the travel time between r's corresponding destinations.
4: For each robot, add a vertex to the HRRP for its starting pose. The weight from the start to each vertex the robot can visit is the travel time between the robot's current pose and the vertex's corresponding destination.

but from different poses. Determining that the robot's destinations – s_1 and q_1 – that should correspond to the same destination in the HRRP is easy in this case, but more difficult in general as the sets of features robots observe may only partially overlap. To address this issue, we define a concrete measure of when two robot destinations are equivalent, and describe an algorithm for finding matching destinations between multiple robots.

Intuitively, if one robot observes features at a destination that are a strict subset of features another robot observes at a different destination, then there is little benefit in both robots visiting their destinations. To formalize this idea, let $d_i^{r_1} = (x_i^{r_1}, F_i^{r_1}) \in D^{r_1}$ and $d_i^{r_2} = (x_i^{r_2}, F_i^{r_2}) \in D^{r_2}$ be destinations with features for robots r_1 and r_2. We measure the similarity between these destinations as the overlap coefficient between the feature sets:

$$S(d_i^{r_1}, d_j^{r_2}) = \frac{|F_i^{r_1} \cap F_j^{r_2}|}{\min\{|F_i^{r_1}|, |F_j^{r_2}|\}} \tag{8.4}$$

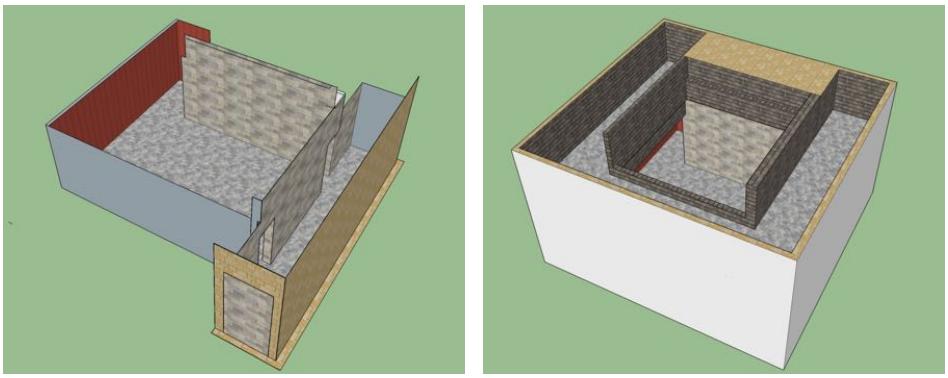
This value is 1 if and only if $F_i^{r_1} \subseteq F_j^{r_2}$ or vice-versa, and is 0 when the two sets have no elements in common. We say that two sets are equivalent when the overlap coefficient is above a user-specified threshold. High values of this threshold ensure that every feature in the environment will be observed, with the potential downside that multiple robots may travel to similar places. Conversely, lower values will cause more destinations to be considered equivalent, with the potential that some features may not be observed.

The overlap coefficient, (8.4), only determines when *pairs* of destinations are equivalent. However, with larger teams, it is possible that destinations for 3 or more robots should all be considered equivalent. Consequently, we extend our notion of equivalence by saying that sets of destinations are equivalent if every pair of those destinations satisfies the overlap criterion.

Given this notion of similarity, we can define an instance of the HRRP by finding sets of destinations across robots that are equivalent. To do this, we define a graph G in such a way that maximal cliques in the graph represent destinations for the HRRP. Specifically, vertices in G are destinations from all of the robots and we add edges between vertices if the overlap criterion is satisfied. This makes sets of equivalent destinations the same as cliques in G (i.e., sets of vertices where each pair is connected), which can then be used to create vertices in the HRRP. We add weighted edges weights between vertices in the HRRP, when a robot is present in both of the corresponding cliques in G. An edge's weight is the time it takes the corresponding robot to travel between the destinations within the associated cliques. Alg. 4 describes this process in detail. Finding all maximal cliques in a graph is exponential in the number of vertices in the graph [141], limiting the practicality of this approach to small teams of robots.

8.3 Results

To assess the performance of the algorithms developed in this chapter, we design a simulation in which a ground robot and quadrotor must map two floors of a building that is similar to Skirkanich Hall at the University of Pennsylvania (Fig. 8.5). The simulation setup and parameters are the same as the simulations in Ch. 6 and Ch. 7. A single global planner creates and solves an instance of the HRRP every 15 seconds. To cluster destinations for a single robot, we require a minimal Jaccard index of 0.5, and that every destination in the set is within 3.0 m of each other (Sect. 8.2.1). To match goals between the robots, we use a cutoff of 0.6 for the overlap coefficient (Sect. 8.2.2). Upon receiving a new path, each robot travels to the first destination and uses the CSQMI based trajectory optimization



(a) 1st floor (b) 2nd floor

Figure 8.5: Simulated Skirkanich Hall. The quadrotor can travel to both floors, but cannot fit through the doors on the first floor. The ground robot can travel to any part of the first floor.

from Ch. 7 to make the path locally optimal with respect to information. If another path is not generated by the time a robot reaches its first destination, they use the same strategy as Ch. 7, and compare local motion primitives to the remainder of the global path, selecting whichever maximize the rate of information gain.

We ran three trials with the quadrotor and ground robot both starting in the middle of the first floor. Fig. 8.7 shows the decrease in entropy of the map. As in previous chapters, the robots took actions that effectively reduce the map's uncertainty. Fig. 8.6 illustrates Trial 2, showing the generated paths at various points in time. Overall, the paths effectively coordinated the robots, leading them to map separate parts of the environment.

A major concern for this cooperative mapping approach was formulating a problem that can be solved sufficiently quickly. Fig. 8.8a shows the total time it took to generate a full plan for both robots throughout Trial 2 and Fig. 8.8a gives a breakdown of this for each of these plans. By clustering destinations for each robot, the resulting HRRP instance was small enough that the corresponding TSP could be solved optimally and quickly using Concorde [3]. Over all trials, the total planning time never exceeded 5.0 seconds. The single largest contributor to the overall planning time was the time to plan shortest paths between destinations for the quadrotor. This is not entirely surprising as planning in 3D is relatively expensive, and generating the HRRP instance requires a large number of paths

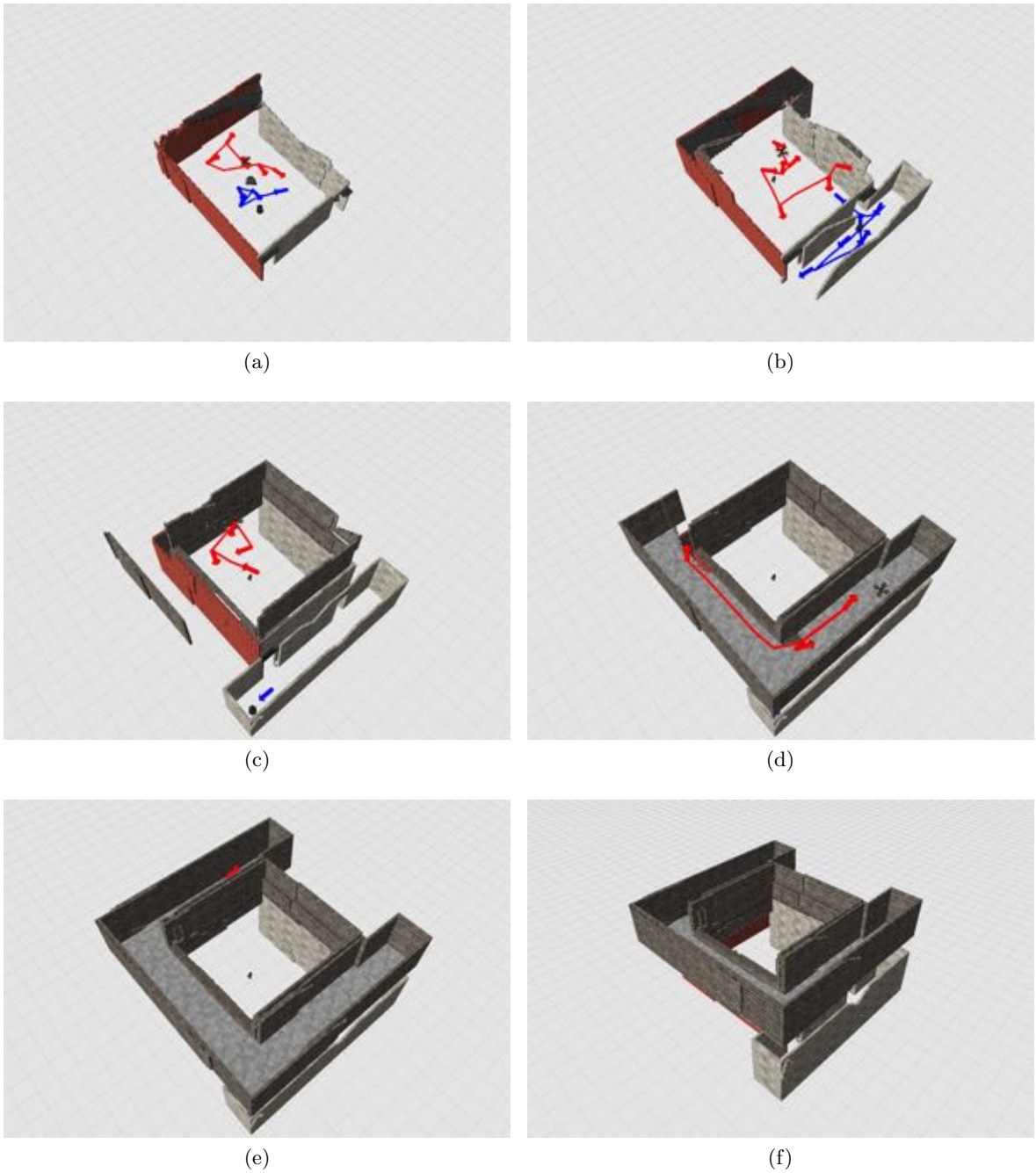


Figure 8.6: Cooperative mapping. The red line shows the quadrotor's path, while the blue line shows the ground robot's path. (a) Both robots start exploring. (b) The quadrotor begins flying up, while the ground robot enters the room the quadrotor cannot. (c) The ground robot completely maps the room and can no longer reduce the map's uncertainty. (d) and (e) the quadrotor maps the second floor. (f) The final 3D map.

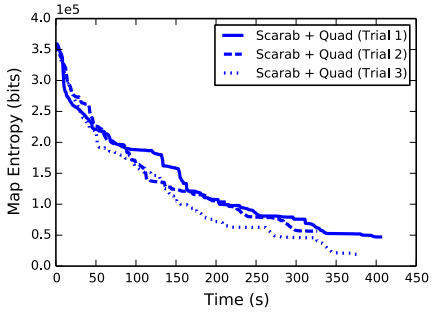
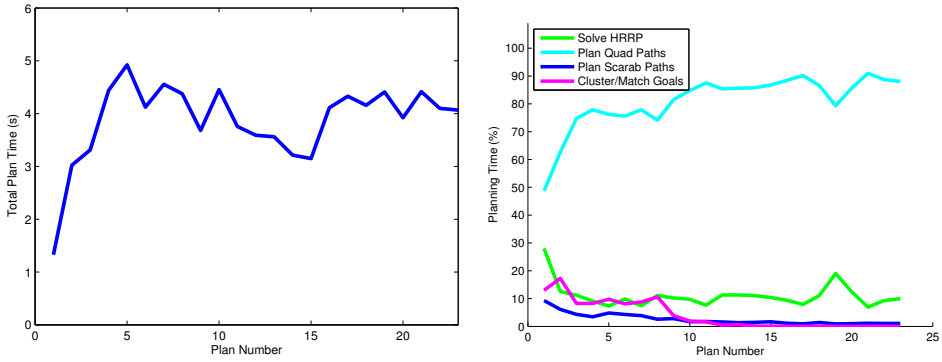


Figure 8.7: Entropy Reduction. By coordinating their actions, the robots consistently built a low uncertainty map.



(a) Total Time (b) Time Breakdown

Figure 8.8: Time to solve HRRP. (a) Total time to generate and solve the HRRP; a new problem was formulated and solved every 15 seconds. (b) Breakdown of time by process. The time it takes to plan paths between destinations for the quadrotor tends to dominate the time to generate and solve the HRRP instance.

to be planned.

8.4 Conclusion

In this chapter we presented a method for coordinating heterogeneous robots for active mapping. The primary contribution was the formulation of a combinatorial optimization problem that attempts to find paths for robots that minimizes the time it takes for them to collectively view all frontiers. These paths can serve as the global plan for the information-theoretic control policies developed in Ch. 6 and Ch. 7. We demonstrated the viability of this approach by showing simulation results in which a single ground robot and quadrotor cooperatively mapped a two-story building.

Chapter 9

Conclusion and Future Work

9.1 Contributions

Designing active perception policies for multi-robot teams is difficult. The key challenges are: (1) Choosing an objective function that can quantify the impact of multiple nonlinear measurements on non-Gaussian belief distributions, (2) appropriately designing the set of actions the team can execute, and (3) developing approximate representations so that the control policy can be computed in a reasonable amount of time.

This thesis has made a number of contributions in each of these areas and substantially extends our ability to quickly compute control policies that cause robots to gather information and reduce uncertainty. Towards (1), we argued for the use of holistic measures like mutual information that account for the complete evolution of the belief distribution and showed that new measures like Cauchy-Schwarz quadratic mutual information offer many of the same benefits and are faster to compute. We also developed a new method for calculating the information from multiple measurements, without relying on a strong independence assumption. To aid in the design of control actions, (2), we proved error bounds showing that for noisy low-information sensors, nearby actions must have similar information gain, but that in applications like active mapping, locally refining trajectories can substantially increase performance. We also demonstrated the benefit of a hierarchy of planning approaches over multiple time and length scales, particularly when coordinating hetero-

geneous robots. To compute these policies online and quickly, (3), we developed several new methods, including approximations for the belief, approximations for the information-theoretic objectives, and caching for numerical gradient evaluation. In several cases, we were able to develop analytic error bounds for these approximations.

In addition to the theoretical contributions of this thesis, we demonstrated the efficacy of our proposed control policies through experiments and simulations. For range-only localization, we showed how our approach enables small teams of robots to track a mobile target in real time and how larger teams can localize an unknown number of static targets. For active mapping, we designed a control policy that lets air and ground robots build dense 3D maps of indoor environments, outperforming several baseline approaches, and nearly achieving the performance of a knowledgeable human operator. We also extended these results to scenarios where air and ground robots must cooperatively map an environment and account for their differences in mobility.

9.2 Future Work

There are a number of interesting directions for future work. One immediate question is to what extent the methods developed in this thesis are sufficient to enable rapid information gathering with highly dynamic vehicles. For example, commercially available quadrotors can fly at 10 m/s while carrying informative sensors like stereo cameras. However, doing this autonomously in cluttered environments remains a significant challenge. By creating an objective that constrains how much information a robot can make in terms of its motion model, the work on trajectory optimization in Ch. 7 makes progress on this problem, but it is unclear whether additional improvements are necessary to travel at these high speeds.

Concerning heterogeneity, the work in Ch. 8 represents a good high-level approach for coordinating a small number of robots, but it scales poorly in the size of the team. In the scenarios we examined, both robots were equipped with similar sensor packages. One question is what kinds of control policies could enable hundreds or thousands of different types of robots to cooperatively gather information. These types of questions will likely

become more relevant for large numbers of small robots that exist on micro- or nano-scales.

As the size of the team grows, the assumptions we made about communication and the centralized nature of the control policies in this thesis are more likely to fail. One way to address this issue would be to directly incorporate models of communication into the control policy. Doing is difficult, as RF signals can change suddenly depending on the environment the robots are in (Ch. 3), but the potential gain in robustness would be large. A particularly interesting outcome would be a policy that let the team take actions that temporarily broke communication between all robots, so that some of them could gather information and then rejoin the rest of the team. Other researchers have made progress on these types of problems [29, 68, 39, 130], but significant challenges still remain.

Another avenue for future research is to apply information gathering to different sensing modalities. While the laser range finders, structured light cameras and range-sensors used in this thesis are different, we modeled all of them in terms of the geometric information they provide (e.g., distance to target or bearing and distance to an occupied part of the environment). Other types of sensors like microphones (sound) or vibration sensors (touch), provide an interesting alternative source of information about environments that could be relevant in security or medical applications.

Finally, the information based control policies in this thesis are all designed to select actions that reduce the uncertainty of a belief distribution. This is a sensible choice when the robots' task is finding a target or mapping its environment, but what if their goal is something else like moving an object? One challenging direction for future research would be to develop a notion of "actionable" information that enables robots to reason about what they need to know in order to achieve a particular goal. Recent work by Kaelbling and Lozano-Pérez [67] studies this problem, but properly formulating it is still quite difficult.

Appendix A

Platforms

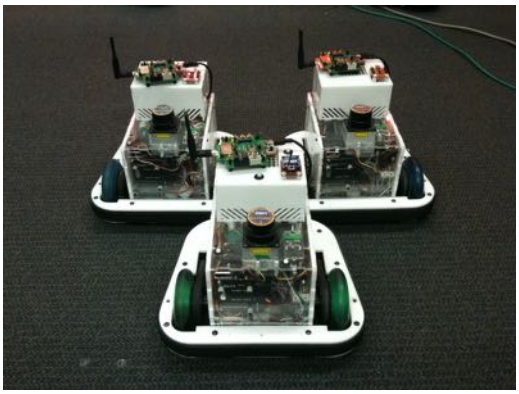
A.1 Scarabs

Over the past 7 years, MRSL has been designing and building the Scarab mobile robot, a simple and relatively cheap robotic platform that is useful for conducting multi-robot experiments. The Scarabs are simple differential drive robots that are equipped with laser scanners for localization and 802.11s wireless mesh cards for communication. The overall motivation for their development as well as the first three generations are described by Michael et al. [82].

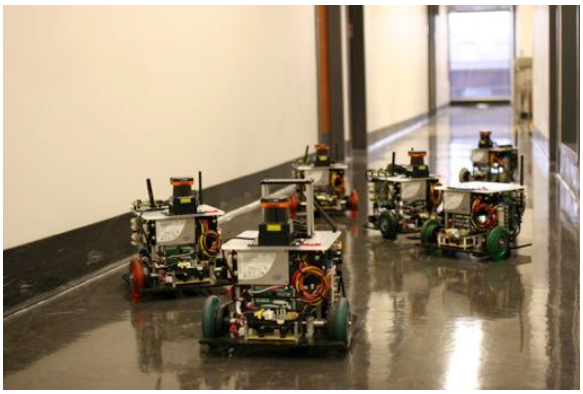
For results described in this thesis, we used the 3^{rd} generation of the Scarabs (Ch. 3 and Ch. 4) and the 4^{th} (Ch. 6 and Ch. 7).

The 3^{rd} generation Scarabs are equipped with a Hokuyo-URG04LX laser range finder, wheel encoders for odometry estimates, and a 2GHz computer with 512 MB RAM. Despite the high accuracy of the URG04LX, it has a relatively short maximum range of $4 - 6$ m, occasionally making it difficult for the Scarabs to reliably and accurately estimate their position in environments containing large open spaces (e.g., long hallways). The maximum speed of these Scarabs is 0.5 m/s.

The 4^{th} generation Scarabs are more powerful than the third, and designed to travel outside of the lab more easily. They are equipped with a UTM-30LX laser range finder, wheel encoders, and an Intel Core i5 3.4 GHz CPU processor with 8 GB RAM. The larger



(a) Generation 3



(b) Generation 4

Figure A.1: Scarab mobile robot.

maximum range of the laser, $28-30$ m, enables highly accurate odometry estimates using a correlative scan matching algorithm [94]. The faster computer also enables this generation to run all of the algorithms presented in this thesis on board. The 4^{th} generation is also more modular than previous generations as they have a generic sensor plane for easily adding and removing sensors. Their maximum speed is 0.9 m/s with a maximum payload of 4 kg.

Both the 3^{rd} and 4^{th} generation Scarabs are powered by lithium-ion batteries. When constantly driving and fully utilizing the available CPUs, they can drive for about 1.5 hours without needing to replace the batteries.

A.2 Quadrotors

The quadrotor used in Ch. 6 of this thesis was an AscTec Pelican equipped with a UTM-30LX and Asus Xtion Pro. It performs laser-based localization onboard using the system by Shen et al. [116]. Its computer is an Intel NUC, which has an i5 1.8GHZ processor with 8 GB RAM.

Bibliography

[1] F. Amigoni and V. Caglioti. An Information-based Exploration Strategy for Environment Mapping with Mobile Robots. *Robotics and Autonomous Systems*, 58(5): 684–699, 2010.

[2] C. Anderson. Agricultural drones. http://www.technologyreview.com/featuredstory/526491/agricultural-drones/, April 2014.

[3] D. Applegate, R. Bixby, V. Chvatal, and W. Cook. Concorde tsp solver. http://www.math.uwaterloo.ca/tsp/concorde/, 2006.

[4] R. Bajcsy. Active perception. Technical Report 619, University of Pennsylvania, 1988.

[5] A. Behzad and M. Modarres. A new efficient transformation of the generalized traveling salesman problem into traveling salesman problem. In *International Conference of Systems Engineering*, 2002.

[6] T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

[7] J. Binney, A. Krause, and G. S. Sukhatme. Optimizing waypoints for monitoring spatiotemporal phenomena. *Int. J. Robot. Research*, 32(8):873–888, 2013.

[8] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[9] B. Boots and G. J. Gordon. A spectral learning approach to range-only slam. In *International Conference on Machine Learning*, pages 19–26, 2013.

[10] F. Bourgault, A. A. Makarenko, S. B. Williams, B. Grocholsky, and H. F. Durrant-Whyte. Information based adaptive robotic exploration. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 540–545, 2002.

[11] F. Bullo, J. Cortés, and S. Martínez. *Distributed Control of Robotics Networks: A Mathematical Approach to Motion Coordination Algorithms*. Princeton University Press, 2009.

[12] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider. Coordinated multi-robot exploration. *IEEE Trans. on Robot.*, 21(3):376–386, 2005.

[13] S. Carpin, D. Burch, and T. H. Chung. Searching for multiple targets using probabilistic quadtrees. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 4536–4543, Sept. 2011.

[14] B. Charrow, N. Michael, and V. Kumar. Cooperative multi-robot estimation and control for radio source localization. In *Proc. of the Int. Sym. on Exp. Robot.*, Québec, Canada, June 2012.

[15] B. Charrow, V. Kumar, and N. Michael. Approximate representations for multi-robot control policies that maximize mutual information. In *Proc. of Robot.: Sci. and Syst.*, Berlin, Germany, June 2013.

[16] B. Charrow, V. Kumar, and N. Michael. Approximate representations for multi-robot control policies that maximize mutual information. *Auton. Robots*, 37(4):383–400, 2014.

[17] B. Charrow, N. Michael, and V. Kumar. Cooperative multi-robot estimation and control for radio source localization. *Int. J. Robot. Research*, 33(4):569–580, 2014.

[18] B. Charrow, N. Michael, and V. Kumar. Active control strategies for discovering and localizing devices with range-only sensors. In *Proc. Workshop on Algorithmic Foundations of Robotics*, Aug. 2014.

[19] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar. Information-theoretic planning with trajectory optimization for dense 3d mapping. In *Proc. of Robot.: Sci. and Syst.*, Rome, Italy, 2015. To appear.

[20] B. Charrow, S. Liu, V. Kumar, and N. Michael. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. Technical Report MS-CIS-15-02, University of Pennsylvania, 2015.

[21] B. Charrow, S. Liu, V. Kumar, and N. Michael. Information-theoretic mapping using cauchy-schwarz quadratic mutual information. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Seattle, USA, 2015.

[22] S. Chen, Y. Li, and N. M. Kwok. Active vision in robotic systems: A survey of recent developments. *Int. J. Robot. Research*, 30(11):1343–1377, 2011.

[23] W.-P. Chin and S. Ntafos. Optimum watchman routes. *Information Processing Letters*, 28(1):39–44, 1988.

[24] H. L. Choi and J. P. How. Continuous trajectory planning of mobile sensors for informative forecasting. *Automatica*, 46(8):1266–1275, 2010.

[25] T. Chung, G. Hollinger, and V. Isler. Search and pursuit-evasion in mobile robotics. *Auton. Robots*, 31(4):299–316, 2011.

[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.

[27] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley Online Library, 2004.

[28] A. Dame and E. Marchand. Mutual Information-based Visual Servoing. *IEEE Trans. on Robotics*, 27(5):958–969, 2011.

[29] P. Dames and V. Kumar. Cooperative multi-target localization with noisy sensors. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Karlsruhe, Germany, May 2013.

[30] A. Das, M. Diu, N. Mathew, C. Scharfenberger, J. Servos, A. Wong, J. S. Zelek, D. A. Clausi, and S. L. Waslander. Mapping, planning, and sample detection strategies for autonomous exploration. *Journal of Field Robotics*, 31(1):75–106, 2014.

[31] S. Dasgupta and P. M. Long. Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4):555–569, 2005.

[32] J. Djugash and S. Singh. Motion-aided network slam with range. *Int. J. Robot. Research*, 31(5):604–625, 2012.

[33] J. Djugash, S. Singh, G. Kantor, and W. Zhang. Range-only SLAM for robots operating cooperatively with sensor networks. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 2078–2084, Orlando, USA, 2006.

[34] A. Domahidi. FORCES: Fast optimization for real-time control on embedded systems. http://forces.ethz.ch, Oct. 2012.

[35] C. Dornhege and A. Kleiner. A Frontier-Void-based Approach for Autonomous Exploration in 3D. *Advanced Robotics*, 27(6):459–468, 2013.

[36] B. Englot and F. S. Hover. Three-dimensional coverage planning for an underwater inspection robot. *Int. J. Robot. Research*, 32(9-10):1048–1073, 2013.

[37] R. Fabbri, L. Costa, J. C. Torelli, and O. M. Bruno. 2D Euclidean Distance Transform Algorithms: A Comparative Survey. *ACM Computing Surveys*, 40(1):2, 2008.

[38] M. Fannes. A continuity property of the entropy density for spin lattice systems. *Communications in Mathematical Physics*, 31(4):291–294, 1973.

[39] J. Fink. *Communication for Teams of Networked Robots*. PhD thesis, University of Pennsylvania, 2011.

[40] C. Forster, M. Pizzoli, and D. Scaramuzza. Appearance-based Active, Monocular, Dense Reconstruction for Micro Aerial Vehicle. In *Proc. of Robot.: Sci. and Syst.*, 2014.

[41] D. Fox. Adapting the sample size in particle filters through KLD-sampling. *Int. J. Robot. Research*, 22(12):985–1003, 2003.

[42] L. Freda, G. Oriolo, and F. Vecchioli. Sensor-based Exploration for General Robotic Systems. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 2157–2164, 2008.

[43] K. Fu. https://spqr.eecs.umich.edu/moo/apps/concrete/, Mar. 2014.

[44] A. L. Gibbs and F. E. Su. On choosing and bounding probability metrics. *Int. Statistical Review*, 70(3):419–435, 2002.

[45] S. Gil, M. Schwager, B. J. Julian, and D. Rus. Optimizing communication in air-ground robot networks using decentralized control. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 1964–1971, Anchorage, USA, May 2010.

[46] D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. 42(1):427–486, 2011.

[47] H. H. González-Banos and J.-C. Latombe. Navigation Strategies for Exploring Indoor Environments. *Int. J. Robot. Research*, 21(10-11):829–848, 2002.

[48] L. Greengard and J. Strain. The fast gauss transform. *SIAM Journal on Scientific and Statistical Computing*, 12(1):79–94, 1991.

[49] B. Grocholsky. *Information-theoretic control of multiple sensor platforms*. PhD thesis, University of Sydney, 2002.

[50] T. Hahn. Cuba. http://www.feynarts.de/cuba/, Jan. 2013.

[51] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2009.

[52] G. Hoffmann and C. Tomlin. Mobile sensor network control using mutual information methods and particle filters. *IEEE Trans. Autom. Control*, 55(1):32–47, 2010.

[53] G. Hollinger and G. Sukhatme. Sampling-based motion planning for robotic information gathering. In *Proc. of Robot.: Sci. and Syst.*, Berlin, Germany, June 2013.

[54] G. Hollinger, J. Djugash, and S. Singh. Target tracking without line of sight using range from radio. *Auton. Robots*, 32(1):1–14, 2011.

[55] G. Hollinger, B. Englot, F. S. Hover, U. Mitra, and G. S. Sukhatme. Active planning for underwater inspection and the benefit of adaptivity. *Int. J. Robot. Research*, 32 (1):3–18, 2013.

[56] D. Holz, N. Basilico, F. Amigoni, and S. Behnke. A Comparative Evaluation of

Exploration Strategies and Heuristics to Improve Them. In *European Conf. on Mobile Robots (ECMR)*, pages 25–30, 2011.

[57] K. Hornik and B. Grün. Tsp-infrastructure for the traveling salesperson problem. *Journal of Statistical Software*, 23(2):1–21, 2007.

[58] M. Huber and U. Hanebeck. Progressive gaussian mixture reduction. In *Intl. Conf. on Inform. Fusion*, 2008.

[59] M. Huber, T. Bailey, H. Durrant-Whyte, and U. Hanebeck. On entropy approximation for gaussian mixture random vectors. In *Multisensor Fusion and Integr. for Intell. Syst.*, pages 181–188, Seoul, Korea, Aug. 2008.

[60] V. Indelman, L. Carlone, and F. Dellaert. Planning Under Uncertainty in the Continuous Domain: A Generalized Belief Space Approach. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, Hong Kong, China, June 2014.

[61] R. Jonker and T. Volgenant. Transforming asymmetric into symmetric traveling salesman problems. *Operations Research Letters*, 2(4):161–163, 1983.

[62] D. Jourdan, J. Deyst, M. Win, and N. Roy. Monte carlo localization in dense multipath environments using UWB ranging. In *IEEE Intl. Conf. on Ultra-Wideband*, pages 314–319, Zurich, Switzerland, Sept. 2005.

[63] B. J. Julian. *Mutual Information-based Gradient-Ascent Control for Distributed Robotics*. PhD thesis, Massachusetts Institute of Technology, 2013.

[64] B. J. Julian, M. Angermann, M. Schwager, and D. Rus. A scalable information theoretic approach to distributed robot coordination. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 5187–5194, San Francisco, USA, Sept. 2011.

[65] B. J. Julian, S. Karaman, and D. Rus. On mutual information-based control of range sensing robots for mapping applications. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 5156–5163, Nov. 2013.

[66] B. J. Julian, S. Karaman, and D. Rus. On Mutual Information-based Control of Range Sensing Robots for Mapping Applications. *Int. J. Robot. Research*, 33(10): 1375–1392, 2014.

[67] L. P. Kaelbling and T. Lozano-Pérez. Integrated task and motion planning in belief space. *Int. J. Robot. Research*, 32(9–10), 2013.

[68] A. Kassir, R. Fitch, and S. Sukkarieh. Decentralised information gathering with communication costs. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 2427–2432, Saint Paul, USA, May 2012.

[69] A. Kim and R. M. Eustice. Perception-driven Navigation: Active Visual SLAM for

Robotic Area Coverage. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 3196–3203, 2013.

[70] N. Koenig and A. Howard. Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 2149–2154, 2004.

[71] T. Kollar and N. Roy. Trajectory Optimization using Reinforcement Learning for Map Exploration. *Int. J. Robot. Research*, 27(2):175–196, 2008.

[72] A. Krause and C. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Uncertainty in AI*, pages 324–331, 2005.

[73] H. Kretzschmar and C. Stachniss. Information-theoretic compression of pose graphs for laser-based slam. *Int. J. Robot. Research*, 31(11):1219–1230, Sept. 2012.

[74] P. Lazik and A. Rowe. Indoor pseudo-ranging of mobile devices using ultrasonic chirps. In *ACM Conference on Embedded Network Sensor Systems*, page 99, New York, USA, Nov. 2012.

[75] J. Leonard and H. F. Durrant-Whyte. Simultaneous Map Building and Localization for an Autonomous Mobile Robot. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 1442–1447, 1991.

[76] M. Levandowsky and D. Winter. Distance between sets. *Nature*, 234(5323):34–35, 1971.

[77] D. Levine, B. Luders, and J. How. Information-theoretic motion planning for constrained sensor networks. *Journal of Aerospace Information Systems*, 10(10):476–496, 2013.

[78] M. Likhachev and D. Ferguson. Planning Long Dynamically Feasible Maneuvers for Autonomous Vehicles. *Int. J. Robot. Research*, 28(8):933–945, 2009.

[79] W. Maddern, A. Harrison, and P. Newman. Lost in translation (and rotation): Rapid extrinsic calibration for 2d and 3d lidars. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 3096–3102, May 2012.

[80] R. Marchant and F. Ramos. Bayesian Optimisation for Informative Continuous Path Planning. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, 2014.

[81] S. Martínez and F. Bullo. Optimal sensor placement and motion coordination for target tracking. *Automatica*, 42(4):661–668, 2006.

[82] N. Michael, J. Fink, and V. Kumar. Experimental testbed for large multirobot teams. *IEEE Robot. and Autom. Mag.*, 15(1):53–61, Mar. 2008.

[83] N. Michael, S. Shen, K. Mohta, Y. Mulgaonkar, V. Kumar, K. Nagatani, Y. Okada, S. Kiribayashi, K. Otake, K. Yoshida, et al. Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, 29(5): 832–841, 2012.

[84] S. J. Moorehead. *Autonomous Surface Exploration for Mobile Robots*. PhD thesis, Carnegie Mellon University, 2001.

[85] C. Morelli, M. Nicole, V. Rampa, and U. Spagnolini. Hidden markov models for radio localization in mixed LOS/NLOS conditions. *IEEE Trans. Signal Process.*, 55(4): 1525–1542, 2007.

[86] nanoPAN 5375 Development Kit. http://nanotron.com/EN/pdf/Factsheet_nanoPAN_5375_Dev_Kit.pdf, Sept. 2013.

[87] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functionsi. *Mathematical Programming*, 14(1):265–294, 1978.

[88] P. Newman and J. Leonard. Pure range-only sub-sea slam. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, volume 2, pages 1921–1926, 2003.

[89] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Verlag, 1999.

[90] J. L. Ny and G. J. Pappas. On Trajectory Optimization for Active Sensing in Gaussian Process Models. In *Proc. IEEE Int. Conf. on Decision and Control (CDC)*, pages 6286–6292, 2009.

[91] P. Oberlin, S. Rathinam, and S. Darbha. A transformation for a heterogeneous, multiple depot, multiple traveling salesman problem. In *American Control Conf.*, pages 1292–1297, 2009.

[92] Occupancy Grid of Intel Lab. Intel Lab Occupancy Grid http://www.informatik.uni-freiburg.de/~stachnis/datasets, Sept. 2013.

[93] E. Olson. *Robust and Efficient Robotic Mapping*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, June 2008.

[94] E. Olson. Real-time correlative scan matching. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 4387–4393, Kobe, Japan, June 2009. IEEE.

[95] E. Olson, J. J. Leonard, and S. Teller. Robust range-only beacon localization. *IEEE Journal of Oceanic Engineering*, 31(4):949–958, Oct 2006.

[96] D. Owen. A table of normal integrals. *Communications in Statistics-Simulation and Computation*, 9(4):389–419, 1980.

[97] J. Pan, Z. Chen, and P. Abbeel. Predicting Initialization Effectiveness for Trajectory Optimization. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, 2014.

[98] S. Patil, Y. Duan, J. Schulman, K. Goldberg, and P. Abbeel. Gaussian Belief Space Planning with Discontinuities in Sensing Domains. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, 2014.

[99] N. Patwari, J. Ash, S. Kyperountas, A. H. III, R. Moses, and N. Correal. Locating the nodes: cooperative localization in wireless sensor networks. *IEEE Signal Process. Mag.*, 22(4):54–69, 2005.

[100] J. C. Principe. *Information Theoretic Learning: Renyi's Entropy and Kernel Perspectives*. Information Science and Statistics. Springer, 2010.

[101] A. Prorok and A. Martinoli. Accurate localization with ultra-wideband: Tessellated spatial models and collaboration. In *Proc. of the Int. Sym. on Exp. Robot.*, pages 321–335, 2012.

[102] S. Rao. *Unsupervised Learning: An Information Theoretic Framework*. PhD thesis, University of Florida, 2008.

[103] T. S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 1996.

[104] A. Renyi. On measures of entropy and information. In *Fourth Berkeley Symposium on Mathematical Statistics and Probability*, pages 547–561, 1961.

[105] R. Rocha, J. Dias, and A. Carvalho. Cooperative Multi-Robot Systems: A Study of Vision-based 3D Mapping using Information Theory. *Robotics and Autonomous Systems (RAS)*, 53(3):282–311, 2005.

[106] ROS. http://wiki.ros.org, Oct. 2014.

[107] A. Rowe, M. E. Berges, G. Bhatia, E. Goldman, R. Rajkumar, J. H. Garrett, J. M. F. Moura, and L. Soibelman. Sensor andrew: Large-scale campus-wide sensing and actuation. *IBM Journal of Research and Development*, 55:6:1–6:14, Jan. 2011.

[108] A. Runnals. Kullback-leibler approach to gaussian mixture reduction. *IEEE Trans. on Aero. and Elec. Sys.*, 43(3):989–999, July 2007.

[109] RVO2. http://gamma.cs.unc.edu/RVO2/, Jan. 2013.

[110] A. Ryan and J. K. Hedrick. Particle filter based information-theoretic active sensing. *Robot. and Autonomous Syst.*, 58(5):574–584, May 2010.

[111] B. M. Sadler, N. Liu, Z. Xu, and R. Kozick. Range-based geolocation in fading environments. In *Allerton Conf. on Comm., Control, and Comput.*, pages 15–20, Allerton House, USA, 2008.

[112] M. Schwager, P. Dames, D. Rus, and V. Kumar. A Multi-Robot Control Policy for Information Gathering in the Presence of Unknown Hazards. In *Proc. of the Int. Sym. on Robot. Research*, 2011.

[113] M. Schwager, D. Rus, and J.-J. Slotine. Unifying geometric, probabilistic, and potential field approaches to multi-robot deployment. *Int. J. Robot. Research*, 30(3): 371–383, 2011.

[114] R. Shade and P. Newman. Choosing where to go: Complete 3d exploration with stereo. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 2806–2811, May 2011.

[115] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:623–656, October 1948.

[116] S. Shen, N. Michael, and V. Kumar. Autonomous multi-floor indoor navigation with a computationally constrained mav. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 20–25, Shanghai, May 2011.

[117] S. Shen, N. Michael, and V. Kumar. Stochastic differential equation-based exploration algorithm for autonomous indoor 3d exploration with a micro-aerial vehicle. *Int. J. Robot. Research*, 31(12):1431–1444, Nov. 2012.

[118] J. F. Silva and P. Parada. Sufficient conditions for the convergence of the shannon differential entropy. In *IEEE Info. Theory Workshop*, pages 608–612, Paraty, Brazil, Oct. 2011.

[119] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser. Efficient informative sensing using multiple robots. *J. of AI Research*, 34(1):707–755, 2009.

[120] A. Singh, F. Ramos, H. Durrant-Whyte, and W. J. Kaiser. Modeling and decision making in spatio-temporal processes for environmental surveillance. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 5490–5497, 2010.

[121] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *IEEE International Conference on Computer Vision Workshops*, pages 1154–1160, Nov. 2011.

[122] J. Snape, J. van den Berg, S. J. Guy, and D. Manocha. Smooth and collision-free navigation for multiple robots under differential-drive constraints. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 4584–4589, Oct. 2010.

[123] S. Soatto. Steps Towards a Theory of Visual Information: Active Perception, Signal-to-Symbol Conversion and the Interplay between Sensing and Control. *arXiv preprint arXiv:1110.2053*, 2011.

[124] P. Sokkalingam and Y. Aneja. Lexicographic bottleneck combinatorial problems. *Operations Research Letters*, 23(1):27–33, 1998.

[125] J. Spletzer and C. J. Taylor. A framework for sensor planning and control with applications to vision guided multi-robot systems. In *Computer Vision and Pattern Recognition*, 2001.

[126] J. Spletzer and C. J. Taylor. Sensor planning and control in a dynamic environment. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 676–681, 2002.

[127] J. Spletzer and C. J. Taylor. A bounded uncertainty approach to multi-robot localization. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, Las Vegas, USA, Oct. 2003.

[128] C. Stachniss. *Robotic Mapping and Exploration*, volume 55. Springer, 2009.

[129] C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robot.: Sci. and Syst.*, Cambridge, MA, USA, 2005.

[130] J. Stephan, J. Fink, B. Charrow, A. Ribeiro, and V. Kumar. Robust routing and multi-confirmation transmission protocol for connectivity management of mobile robotic teams. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, 2014.

[131] J. Strom and E. Olson. Multi-sensor attenuation estimation (MATTE): Signal-strength prediction for teams of robots. In *Proc. of the IEEE/RSJ Int. Conf. on Intell. Robots and Syst.*, pages 4730 – 4736, Algarve, Portugal, Oct. 2012.

[132] E. Stump, V. Kumar, B. Grocholsky, and P. Shiroma. Control for localization of targets using range-only sensors. *Int. J. Robot. Research*, 28(6):743, 2009.

[133] S. Thrun. Robotic Mapping: A Survey. *Exploring Artificial Intelligence in the New Millennium*, pages 1–35, 2002.

[134] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2008.

[135] M. Turpin. *Safe, Scalable, and Complete Motion Planning of Large Teams of Interchangeable Robots*. PhD thesis, University of Pennsylvania, 2014.

[136] R. Valencia, M. Morta, J. Andrade-Cetto, and J. M. Porta. Planning Reliable Paths with Pose SLAM. *IEEE Trans. on Robotics*, 29(4):1050–1059, 2013.

[137] J. Vander Hook, P. Tokekar, and V. Isler. Cautious greedy strategy for bearing-only active localization: Analysis and field experiments. *Journal of Field Robotics*, 31(2): 296–318, April 2014.

[138] R. Vidal, O. Shakernia, H. J. Kim, D. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Trans. on Robot. and Autom.*, 18(5):662–669, 2002.

[139] A. Visser and B. Slamet. Balancing the information gain against the movement cost for multi-robot frontier exploration. In *European Robotics Symposium*, pages 43–52, 2008.

[140] P. Whaite and F. P. Ferrie. Autonomous Exploration: Driven by Uncertainty. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 19(3):193–205, 1997.

[141] D. R. Wood. On the maximum number of cliques in a graph. *Graphs and Combinatorics*, 23(3):337–352, 2007.

[142] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proc. of the IEEE Int. Conf. on Robot. and Autom.*, pages 146–151, July 1997.

[143] P. L. Yu, J. N. Twigg, and B. M. Sadler. Radio signal strength tracking and control for robotic networks. In *Proc. SPIE*, volume 8031, Orlando, USA, Apr. 2011.