Active Collaborative Planning and Sensing in Human-Robot Teams

by

Charles Luke Burks

B.S., University of Arkansas, 2015

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Ann and H.J. Smead Department of Aerospace Engineering Sciences

2020

Committee Members:

Nisar Ahmed

Eric Frew

Danielle Szafir

Torin Clark

Bradley Hayes

Zachary Sunberg

Burks, Charles Luke (Ph.D., Aerospace Engineering Sciences)

Active Collaborative Planning and Sensing in Human-Robot Teams

Thesis directed by Prof. Nisar Ahmed

As they become increasing capable, autonomous robots can further benefit from human-provided semantic characterizations of uncertain task environments and states. However, the development of integrated strategies which let robots model, communicate, and act on such 'soft data' remains challenging. This work presents a framework for active semantic sensing and planning in human-robot teams which addresses these gaps by developing algorithms and techniques to allow formally integrated semantic sensing and planning in human robot teams, leveraging advances in POMDP approximation, multi-modal semantic interaction, and Bayesian data fusion. This approach lets humans opportunistically impose model structure and dynamically extend the range of semantic soft data in uncertain environments, which otherwise yield little information to a lone robot. It also lets robots actively query humans for new semantic data which update understanding and beliefs of unknown environments for improved online planning. Dynamic target search simulations show that active collaborative semantic sensing leads to significant improvements in time and belief state estimates required for interception versus conventional planning, which relies on robotic sensing only. This thesis contains several contributions advancing the state of the art in human-robot collaborative planning. Chapter 3 derives and implements the VB-POMDP algorithm, which provides for continuous state POMDP planning under hybrid continuous-to-discrete semantic sensor observations modeled by softmax functions. Compared to previous methods, this algorithm scales construction of observation models to previously unreachable integrated planning and control problems. Chapter 4 scales the work of Chapter 3 even further, applying a hierarchical framework for the efficient solution of POMDPs in complex continuous state spaces. It also incorporates active human sensing into such problems, using a semantic dictionary of potential robotic queries to facilitate human-robot information transfer. Finally, Chapter 5 extends collaborative

human-robot target search to unknown a priori environments, and presents a novel sketch-based approach to multi-level active semantic sensing which allows the transfer of both model and state information without the need for retraining.

Dedication

To those on whose shoulders I have been privileged to stand, and those whom I may have the privilege to someday support in turn. Only together can we build a better world.

Acknowledgements

At the end of the journey this thesis represents, it has become clear that the efforts of many besides myself were required to not only complete the journey, but also make it worth completing.

First and foremost, I'd like to thank my family for their support and encouragement. From my loving wife Jennifer who keeps my feet on the ground while my head's in the clouds, to my sister Hope who first tread the graduate path and inspired me to follow, to my parents and younger sister who endured countless conversations about robotics at family gatherings. Thank you.

Immense thanks are also due to my advisor Nisar Ahmed, who over the course of my graduate work played the roles of teacher, mentor, and friend. He was as unfailing in his efforts to mold me into a better researcher as he was in his support at every step of the way.

Next I'd like to express my wholehearted appreciation for the members of the COHRINT lab whom I've had the privilege to work with and beside over the years. It's likely that each and every idea, innovation, and algorithm in this thesis started out being discussed at a white board with a fellow lab member, while their conversations and friendship ensured a truly enjoyable graduate experience. Thank you all.

Finally, I'm grateful to the Center for Unmanned Aircraft Systems (C-UAS), which provided both funding and a regular presentation forum enabling my research.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Humanity keeps time, at the broadest scales, by technology. From the Stone Age, to the Bronze, Iron, and Industrial Ages, a rearward facing perspective emphasizes the tools of the day. It is fitting then, that the modern world is often referenced as the Digital or Computer Age. Computers, in the form of both stationary boxes and mobile robots, have arguably changed and continue to change the human condition more than any technology before them. As proof, it may suffice to point out that in colloquial conversation even the word "technology" itself has come to imply the presence of digital computation.

Autonomy however, the concept of machines thinking and performing tasks independent of human control, predates the existence of computers. From Talos, the mythical bronze guardian of Crete, to folkloric depictions of golems and clay giants, humans had conjured ideas of automatons long before the term "robot" was first used in Karel Čapek's 1920 play, *Rossum's Universal Robots*.

With the rise of computers, actual robots were not only possible, but imminent. Coupled with an incorrigible tendency to anthropomorphize the world, this led people to imagine computers like us, robots gifted with all of the same skills and faults of their creators. Little thought was given to the constant challenges human brush aside on a second by second basis. From keeping our balance on uneven ground to visually identifying one face from another, humans have a remarkable and underappreciated skill set when viewed from the perspective of robotics. Few who saw the original Terminator movie appreciated how difficult it would actually be for the Terminators to stride triumphantly across the ruins of California without falling, or what it would require for Lt.

Commander Data to know who he was stiltedly addressing in Star Trek.

Real artificial intelligence proved difficult, and not for the reasons we thought it would. Games like chess, for all their complexity, ended up producing triumph after triumph for AI [28], while distinguishing different species of birds in photos remains an active research question for which scientists now invoke powerful deep learning models [107]. One core difference emerged in the concept of uncertainty. AI could plan super-humanly well, as long as the rules of the game and their results were predictable. But dealing with uncertainty, whether in movements, sensors, or the "rules" themselves became one of the defining challenges for robotics and autonomy. Learning to deal with this challenge, as well as learning to characterize and model the nearly unbelievable complexity of the world around us, remains one of the foremost opportunities to unlock an ever-greater world of robotic possibilities.

While it is inevitable some robots will face the challenge of uncertainty alone, perhaps in deep space or locked away in a darkened factory, the majority will likely inhabit the world alongside humans [19]. Indeed, it has been said [19] that human-robot teamwork is the inevitable next leap forward in the design of autonomous systems. Even the (thus far) mythical "general artificial intelligence" will need to interact with humans if it is to exist alongside them. Given the difficulties autonomy has faced to date, it would seem to be a step too far to assume such AI will just naturally play well with others. Even humans themselves require extensive training on human interaction in infancy before then can be considered competent teammates [64]. Nonetheless, this additional challenge may well hold the key to the one posed above with regard to uncertainty. Humans have a broad spectrum of capabilities at their disposal for mitigating uncertainty, and possess several strengths difficult to replicate in their autonomous creations. They can recognize the world through multi-modal senses such as sight, sound, and smell. They pre-process vast quantities of information and even intuit heuristic approximations of Bayesian uncertainty regarding their decisions [3]. As these strengths are leveraged for human-robot teaming, they may enable autonomy to finally reach the heights which have so long been imagined.

In this thesis, the problem of uncertainty is addressed in the context of autonomous robotic

planning and sensing, specifically as applied to human-robot teams. With the techniques and advances detailed below, it becomes more possible than ever to acknowledge our differences as complimentary strengths, to allow the human to see and the robot to plan, all in pursuit of a common goal.

1.1 The Technicalities of Planning under Uncertainty

Many applications of planning under uncertainty require autonomous agents to reason over outcomes in continuous dynamical environments using imprecise but readily available semantic observations. For instance, in search and tracking applications, autonomous robots must efficiently reacquire and localize mobile targets that remain out of view for long periods of time. Planning algorithms must therefore generate vehicle trajectories that optimally exploit 'detection' and 'no detection' data from onboard sensors [17, 82], as well as semantic natural language observations that can be provided by human supervisors [6]. However, it remains quite challenging to achieve tight optimal integration of vehicle motion planning with non-linear sensing and non-Gaussian state estimation in large continuous dynamic problem domains. In robots such as small unmanned aircraft systems (UAS), human operators and users can play valuable roles as 'human sensors' that contribute information beyond the reach of autonomous air vehicle sensors. For instance, operators in search and tracking missions can provide 'soft data' to narrow down possible survivor locations using semantic natural language observations (e.g. 'Nothing is around the lake'; 'Something is moving towards the fence'), or provide estimates of physical quantities (e.g. masses/sizes of obstacles, distances from landmarks) to help autonomous vehicles better understand search areas and improve online decision making with limited computational resources. This research focuses on the development of intelligent operator-UAS frameworks that not only leverage combined robotic sensing and semantic human sensing, but are also tightly integrated with vehicle motion planning in large continuous dynamic spaces. This naturally raises the question of how autonomous reasoning can actively and opportunistically engage human reasoning to improve its own performance.

This casting of human data hints at the need for an intelligent human-autonomy interaction

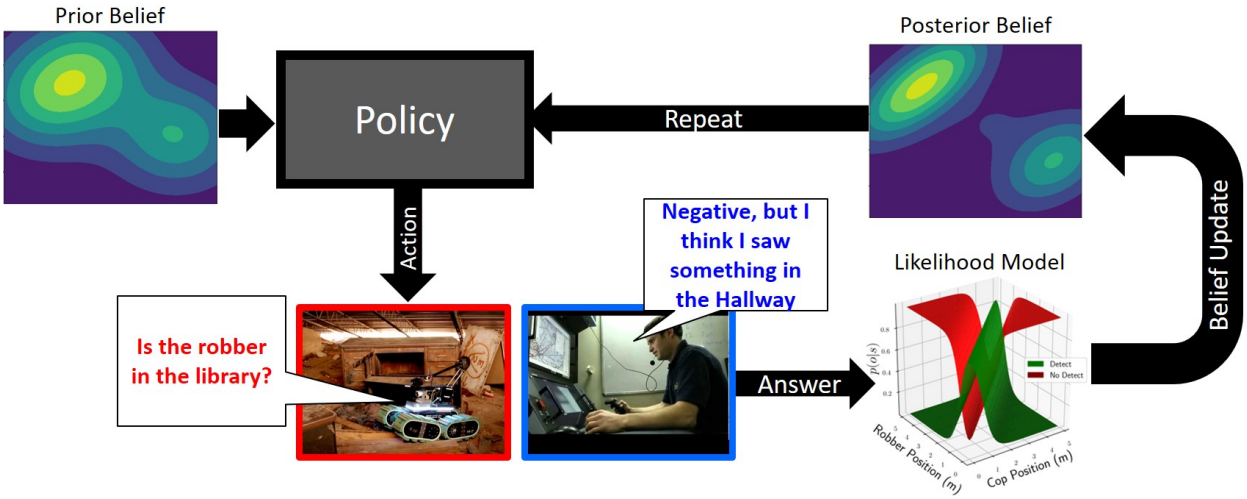


Figure 1.1: Closed-loop collaborative Bayesian target search using a non-myopic policy for simultaneous semantic querying and sensor vehicle motion planning.

that not only leverages combined robot-human sensing, but is also tightly integrated with dynamic platform decision making and planning. Such an approach can use Bayesian data fusion to exploit soft data in such a way as to interface with existing autonomous sensing frameworks, while also enabling accessible and understandable interaction with the system on the part of the human. Ideally, such systems could accept human input as it becomes available, rather than either depending on it to function or having to reconfigure computational architectures to accommodate it. Such 'plug and play' human sensing for robot state estimation was explored in [48, 18] for restricted types of human observations, and has received increased attention in recent years [50, 33, 39, 62].

In this work, recent research on Bayesian semantic natural language human data fusion [6, 100] is combined with concepts from optimal active sensing, in order to develop a new framework for *interactive* human-robot semantic sensing. Here the focus is on the challenging problem of non-myopic decision making for *simultaneous (tightly coupled) vehicle motion planning and human sensor querying* in continuous dynamic search environments.

As shown in Figure 1.1, this approach allows for joint action-query *policies* (i.e. control laws) over a continuous target search space. A policy tells the robot how to respond to target location uncertainty, so that it simultaneously makes optimal decisions about how to move/sense on its own in the environment and about which semantic natural language questions it should ask human

sensors in order to 'pull' useful information. The human only needs to act as a (voluntary) sensor hence, the robot does not "depend" on the human but can opportunistically gain information or adapt to whatever soft information is provided by the human; furthermore, the policy lets the robot conduct an optimal search with complex non-Gaussian uncertainties, even without human input.

Such a policy needs to operate at scale, dealing with large continuous spaces, while requiring no more computational power than a small mobile platform can provide. Algorithms exist to pre-compute the policy offline [47, 69, 72], sidestepping computational constraints and allowing the online execution to take the form of a lookup table mapping uncertainty to actions. Such algorithms, with computation happening remotely and prior to use, are ill-suited for certain real world applications in which sensors and dynamics may change unpredictably or not be analytically well modelled beforehand. In contrast, algorithms which compute such a policy on the fly during execution [89, 92, 99] are more amenable to adaptation in uncertain or changing environments. However, their use of online policy calculations risks over-running the computational or power budgets of smaller mobile robotic platforms.

A variety of techniques based on partially observable Markov decision processes (POMDPs) have been developed to calculate decision policies, both offline and online. These include methods which preserve the continuous dynamical nature of the problem through suitable function approximations, rather than discretizing the continuous state space. Of particular interest here are approximations based on Gaussian mixture (GM) models, which flexibly represent complex policy functions and non-Gaussian probability density functions (pdfs) [24, 72]. These techniques enable closed-form manipulation and recursions for producing accurate approximations of optimal POMDP policies.

However, these state of the art methods suffer from major drawbacks when dealing with semantic observations and real-time robotics. First, they rely on computationally expensive and non-scalable hybrid probabilistic likelihood models for capturing the relationship between discrete semantic sensor data and continuous states. Second, POMDP approaches are typically restricted to highly simplified problems. While nothing stops a complex, high dimensional, human-robot

sensing and planning problem from being *framed* as a single monolithic POMDP, in practice the computational complexity of such algorithms almost always outstrips available resources. Absent new truly revolutionary computing advances, new ways of framing such problems are required to render them accessible. Finally, the majority of state of the art methods depend on a static environment, and build policies assuming their internal model of the world need not drastically change during policy execution. However, this fact proves fundamentally limiting in problem domains such as human-robot target tracking, where new information unattainable a priori can prove decisive.

1.2 Research Questions

This work addresses three fundamental research questions. Each progressively builds on the previous in describing robotic autonomy with active semantic planning and sensing in human-robot teams.

- How can a robot tractably approximate an optimal POMDP policy exploiting semantic information in continuous spaces?
- How can environmental structure and human natural language information be leveraged by a robot to solve complex POMDP planning problems?
- How can human sketch information be used to identify and communicate about salient environmental features such that a robot can adaptively update its planner?

1.3 Contributions

Answering these questions, this work adds several novel contributions to the field of robotics. These contributions include:

- The VB-POMDP Algorithm, a variational Bayes policy approximation which allows the use of continuous state POMDPs with hybrid continuous-to-discrete semantic sensor observations modeled by softmax functions

- A scalable hierarchical framework for efficiently solving CPOMDPs in complex continuous state spaces
- A novel sketch-based approach to multi-level active semantic sensing and planning in human-robot teams, allowing instant information transfer of both model and state transformation from human to robot without the need for policy retraining

Data from both simulations and live hardware experiments are provided showcasing the validity and significance of these contributions.

1.4 Chapter Summaries

Chapter 2 of this work covers the high-level general problem statement associated with active collaborative planning and sensing in human-robot teams. It also summarizes and reviews relevant existing and supporting work on semantic sensing, optimal planning, human-robot collaboration, and language-based data fusion.

Chapter 3 begins the technical discussion of this works contributions, starting with the development and application of the VB-POMDP algorithm to a robotic semantic sensing task. Comprehensive results are examined showcasing the ability of this algorithm to address problems of increasing difficulty, as compared with previous state-of-the-art approaches.

With the VB-POMDP algorithm development accomplished, Chapter 4 introduces live human data into the mix, expanding from the toy problems of Chapter 3 into the more complex Cops and Robots scenario. A novel scalable hierarchical POMDP framework is developed and validated in hardware experiments.

Chapter 5 removes the explicit assumption of Chapters 3 and 4, allowing for adaptable problem modeling in previously unknown/partially known environments. Sketch based methods are developed for knowledge transfer about both environment and problem state from human to robot, and the problem scope is expanded to large outdoor tracking scenarios.

Finally, Chapter 6 summarizes the previous work and presents several opportunities to extend

the work of previous chapters.

Chapter 2

Background

This chapter discusses relevant background literature and research necessary to build up to the technical contributions of Chapters 3, 4 and 5. It begins with a generalized motivating problem in the domain of target tracking before introducing related work on semantic sensing and continuous state planning under uncertainty. A formal mathematical problem statement is then given in the form of a Partially Observable Markov Decision Process (POMDP) along with discussion of existing POMDP approximation methods. The chapter ends with a survey of previous work on algorithms for human-robot sensing and planning, both independent of and as applied to POMDPs[1] .

2.1 Motivation and High Level Problem Description

While this work primarily addresses dynamic target search and interception tasks, the concepts developed here readily generalize to other problems involving decision making under uncertainty in continuous dynamic state spaces, e.g. mobile robot self-localization and motion planning [23, 11, 104]; inventory control [110]; unmanned aircraft collision avoidance [9]; population dynamics modeling [67]; multi-target radar scheduling [55]; robot arm motion planning and coordinated grasping-based manipulation [38, 10, 54, 71]; and autonomous driving [22, 10].

The dynamic target search and interception problems considered here consist of a single autonomous mobile robot platform (the seeker) which must seek out, localize and capture another single mobile entity (the target). The seeker and target dynamics are each described by a finite-

[1] Parts of this chapter consist of material published in IEEE Transactions on Robotics [26]

dimensional continuous state space dynamics model. The seeker robot receives a limited set of noisy sensor observations and can use these to make informed decisions about its own movements, which in turn lead to new future observations and possible interception of the target.

It is assumed throughout that the seeker has perfect (or near-perfect) knowledge and observability of its own state, although its actions may result in uncertain state transitions. This can be relaxed to allow for uncertain seeker states, though it is assumed regardless that the seeker states are observable and the search environment is known, such that obstacles and other known hazards are mapped ahead of time. The seeker also has a (possibly imperfect) state space model of the target, as well as an initial prior belief over target states. The seeker's sensor observations consist of semantic data types that are generated in the continuous space as discrete categorical observations, i.e. positive information in the form of 'target detected' and negative information in the form of 'no target detected' reports from a visual sensor. Continuous sensor observations may also be present (e.g. relative range and bearing measurements), though the semantic/discrete observations are the distinguishing feature and focus for this work.

Given this setup, the seeker must reason about how to intercept the target in some optimal sense. This work focuses on the problem of safe minimum time capture; that is, the seeker must intercept the target as quickly as possible. Alternative performance measures could be optimized, e.g. maximum probability of capture, minimum mean squared error target localization error, minimum power consumption, etc. Regardless, optimal planning requires the seeker to map the target's state, its own state, and its set of possible actions and observations to the maximization of an overall utility for some planning horizon.

Previous work in controls, data fusion, and robotics has expansively addressed target search and tracking and interception for continuous spaces [58, 16, 68, 82]. However, optimal planning under uncertainty in continuous spaces with semantic observations remains quite challenging. The hybrid probabilistic nature of this application presents challenging data fusion and control problems with highly non-Gaussian uncertainties, which are not present in other approaches that rely on continuous measurements and Gaussian uncertainties. In such a problem, the typical approach is

to apply the separation principle, relying on the observability of the state space and properties of the sensor and dynamics models to ensure Gaussian uncertainties over the long-term. However, the separation principle is not guaranteed to produce optimal results for planning under uncertainty with semantic observations, since these are typically non-linear and can lead to highly non-Gaussian uncertainties. Fusion approaches applied to this problem must be able to accommodate arbitrary uncertainties using non-Gaussian sensor and dynamics models.

2.2 Semantic Sensing and Data Fusion

Sensors typically provide continuous numeric observations, such as a range or bearing measurement. However, some sensors provide categorical observations, e.g. the output of a visual object detection algorithm that reports when an object is in a camera sensor's field of view, or a human-generated report that a target is west of a landmark. Such semantic observations map to discrete regions in a continuous space, where the regions are not necessarily exclusive. For example, placing a target at the edge of a camera's view could generate either a positive (true detection) or negative (false miss) observation from the identification algorithm, with some probability for each outcome. When these probabilities are cast in a likelihood model, they provide useful negative information in Bayesian reasoning for target tracking [52] [108], as well as probabilistic generative models of semantic observations for planning problems [89].

In collaborative human-robot search problems, non-robot sensors such as surveillance cameras, unattended ground sensors, or human teammates who generate natural language data can be modeled as semantic data sources [6, 13, 100]. The seeker robot can then augment its decision making process by incorporating actions to actively point and poll these sensors in a closed-loop manner, resulting in a fully integrated hybrid sensing and planning problem. Refs. [49, 61] also consider this problem from the standpoint of myopic Value of Information (VOI) reasoning to determine whether querying a particular sensor will result in a better decision in the long run (i.e. improved utility), despite the cost of using the sensor and regardless of the sensor's observation. These approaches require online optimization and inference for decision-making within a probabilistic graphical model,

and hence decouple the planning and sensing problems to ensure computational tractability. The approach described in this paper can be used to solve for combined motion planning and human querying policies offline, thus avoiding high computational cost and achieving tighter integration of planning and sensing with complex uncertainties. Application to the full semantic active sensing problem with human-robot teams is not treated here, but has been implemented and examined in related work [27].

Semantic data fusion has major consequences for online state estimation and representation. The negative information carried by such data can change the state belief in highly non-linear ways via the 'scattering effect' [16]. This temporarily increases the differential entropy of a continuous target state probability density function (pdf) by introducing non-Gaussian features like holes, skewness, and multiple peaks. Yet, many data fusion and estimation approaches rely heavily on Gaussian state pdfs and likelihood models; these can lose significant information relative to the true target state distribution and thus lead to suboptimal closed-loop search/localization policies. Extensions of these methods generally rely on approximations of pdfs and observation likelihoods via normalized and unnormalized Gaussian mixture (GM) functions, respectively [15, 24]. These methods exploit the fact that, for recursive Bayesian updates, the product of GM state prior distributions and GM semantic likelihood functions is always guaranteed to be another GM. It is fairly well-established that normalized GMs provide highly flexible models for non-Gaussian state estimation, especially if GM condensation techniques are applied to control mixand growth across successive mixture operations [81, 83]. However, the number of parameters required for unnormalized GMs to model semantic data likelihoods in 2 or more continuous state dimensions scales quite poorly and quickly becomes computationally impractical for optimal planning. Previous work also showed that semantic observations could be modeled via softmax functions and fused into (normalized) GM pdfs for recursive Bayesian state estimation [6, 100]. This concept is significantly extended in Chapter 3 for updating unnormalized GM policy functions, which also accounts for the tight coupling between optimal sensing and planning. Further extensions are then considered in Chapter 5 which construct softmax functions in an ad-hoc manner to augment particle filter

estimation in unknown/uncertain environments.

2.3 POMDPs: Planning under Uncertainty

Dynamic target search and interception problems feature many types of stochastic uncertainty, including dynamic process noise and sensor errors. Planners based on Partially Observable Markov Decision Processes (POMDPs) are well-suited to handle such uncertainties. POMDPs can theoretically support arbitrary dynamics, state beliefs (probability distributions), and sensor models, and thus encode a broad range of general optimal decision making problems when specified with an appropriate reward function. POMDP policies can operate on arbitrary target state pdfs, as long as Bayesian belief updates to the target state pdf are carried out. POMDPs also naturally account for VOI when integrated with sensor tasking and information gathering actions. In practice, however, POMDPs can be unworkable due to the curse of dimensionality in discrete spaces and problems with tractability and representation in continuous spaces. The key challenge is the need to solve a Markov Decision Process (MDP) over the state belief space to obtain optimal decision making policies. This is impractical to do exactly for all but the most trivial problems [47]. Hence, it is generally necessary to resort to approximate solutions.

Formally, a POMDP is described by the 7-tuple $(S, A, T, R, \Omega, O, \gamma)$, where: S is a set of states; A is a set of $|A|$ discrete actions a; T is a discrete time probabilistic transition mapping from state s_t at time t to state s_{t+1} at time $t+1$ given some a; R is the immediate reward mapping over (s, a) pairs; Ω is a set of observations o with $N_o = |\Omega|$ possible outcomes; O is the likelihood mapping from states to observations; and $\gamma \in [0, 1]$ is a discount factor. An agent whose decision making process is modeled by a POMDP seeks to maximize a utility function defined by the expected future discounted reward: $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$, where $s_t \in S$ is the state at discrete time t, and $a_t \in A$. The expectation operator $\mathbb{E}[\cdot]$ reflects that the agent lacks full knowledge of s_t. It must instead rely on the noisy process model T and observation model O to update a Bayesian belief function $b(s_t) = p(s_t|a_{1:t}, o_{1:t})$, which summarizes all available information for reasoning about present and possible future states. An optimal decision making policy $\pi(b(s_t)) \rightarrow a_t$ must

therefore be found for any possible belief $b(s_t)$. Since POMDPs are equivalent to Markov Decision Processes (MDPs) over state beliefs $b(s_t)$, exact policies are impossible to compute for all but the simplest problems.

Let $s, s' \in S$ be arbitrary states to which any (approximate) policy must apply such that $s \mapsto s'$ via T (i.e. for any $t \to t{+}1$). One well-known family of techniques for computing approximate POMDP policies offline is Point-Based Value Iteration (PBVI) [69]. These methods approximate π at a finite set of beliefs $\mathcal{B}_0 = \{b_1(s), ..., b_{N_B}(s)\}$, for which explicit finite-horizon Bellman equation recursions can be performed to obtain locally optimal actions in the neighborhood of each $b_i(s)$, $i = 1, ..., N_B$. When S is a set of discrete states with N possible outcomes, then $b(s) \in \mathbb{R}^N$ such that $\sum_{s=1}^{N} b(s) = 1$. In this case, PBVI policies are represented by a set Γ of N_α vectors $\alpha \in \mathbb{R}^N$. The α vectors mathematically represent hyperplanes that encode value functions for taking particular actions at a given belief. The action a recommended by the policy for a given $b(s) \in \mathbb{R}^N$ is found as the action associated with $\arg\max_{\alpha \in \Gamma} < \alpha, b(s) >$, where $< \cdot >$ is the inner product. A number of methods exist for generating typical sample beliefs, e.g. starting with a large set of $b_i(s)$ sampled from the reachable belief space by random simulation [69] (as in this work), or propagating a small initial belief set in between recursive Bellman updates for α vector computations to approximate optimal reachable belief sets [56].

When $\mathcal{S}$ contains a "natural" decomposition into non-overlapping regions, or subspaces, hierarchical POMDP methods can be used. Such algorithms generally maintain a high level POMDP which chooses actions from among the policies of multiple lower level planners. These low level algorithms can be either additional hierarchical POMDPs [70, 46], or specialized algorithms such as computer vision for identifying areas of interest to bound future planning [96]. In many problem domains, subspace decomposition can be carried out by hand, though several automated methods have also been developed [29, 103].

When s is a continuous random vector such that $s \in \mathbb{R}^N$ with support $\mathcal{S}(S)$, it is natural to represent $b(s)$ as a pdf, where $\int_{\mathcal{S}(S)} b(s)ds = 1$. In such cases, continuous state POMDPs (CPOMDPs) can be formulated by specifying T, R, O and $\alpha(s)$ as suitable continuous functions

over s. Although $b(s)$ can sometimes be represented by simple parametric models such as Gaussian pdfs [1], $b(s)$ is in general analytically intractable for arbitrary T and O models that represent non-linear/non-Gaussian dynamics and semantic sensor observations. Therefore, $b(s)$ must also be approximated to derive a suitable set Γ of $\alpha(s)$, such that the (approximate) optimal PBVI policy $\pi(b(s))$ is defined by the action associated with $\arg\max < \alpha(s), b(s) >$.

2.3.1 Continuous State POMDPs

Discrete space POMDP approximations have been regularly applied to target search and interception in prior work; benchmark applications include 'tag/avoid' [69, 56] and laser tag [75]. These approximations generate offline policies for target interception based on a discretization of the continuous state space. However, solving the belief space MDP for these problems carries the curse of dimensionality. For a problem with N discrete states, policies must be found over the continuous space of all $N-$dimensional probability distributions, and thus become intractable to represent. Approximations based on Point-Based Value Iteration (PBVI) [69] attempt to solve the POMDP at specific 'tentpole' beliefs, allowing the policy to be interpolated at other beliefs [95, 56]. Other methods attempt to compress the belief space onto a lower dimensional manifold [80], approximate the POMDP as a single step MDP with observations, e.g. Q-MDP [60], build hierarchical policy structures to scale semi-independent state spaces [70, 96], or use sample states to build trees of potential histories [32] or scenarios [92] in an online fashion during runtime. One of these online sampling methods, Partially Observable Monte Carlo Planning (POMCP) [89], has also been adapted recently to continuous state spaces [40].

Recent years have seen the development of several POMDP policy approximations for continuous state, action, and observation spaces. These include a variety of belief representations, and address the combination of continuous states with discrete or continuous actions and observations. Several continuous POMDP approximation approaches rely on sampling methods [21, 9], in a similar or extended version of the discrete space sampling approaches. Local policy approximations for continuous observations have been applied using Gaussian state beliefs [105, 38]. Belief space

roadmap techniques [73] have had success in computing policies as paths directly in belief space for linear Gaussian systems with continuous observations. Also related are policy approximations inspired by linear-quadratic-Gaussian (LQG) optimal control for motion planning under uncertainty [104, 74].

One family of continuous POMDP approximations extends the PBVI discrete approach to continuous spaces [72]. This approach uses Gaussian mixtures (GMs) to approximate arbitrarily complex pdf beliefs, state transitions, and observation likelihoods of the POMDP. Beliefs are updated via the Gaussian Sum filter, exploiting the fact that GMs become universal function approximators as the number of mixing components (mixands) becomes large. Refs. [24] and [59] extended this idea to address hybrid dynamical systems, where the state transition model switches in different parts of the state space.

These existing techniques are generally ill-suited for complex planning problems in continuous state spaces with semantic observations, such as the target search and interception task considered here. Discretization-based methods lead to an undesirable tradeoff between state space size and fidelity of system dynamics, with larger spaces requiring coarse discretizations that fail to capture subtleties in the target model.

Continuous state policy approximations such as [24, 9] have either relied on the assumption of continuous observations (and are thus not amenable to semantic observations), or constructed semantic observation likelihoods out of GMs. In the latter case, such models are chosen to facilitate calculations and maintain closed form recursions, but drive up computation cost of policy searches and scale poorly with state dimension N due to the number of mixands required to accurately specify likelihoods. Chapter 3 of this thesis develops a more scalable alternative likelihood representation using softmax models for PBVI type policy solutions with GM belief representations. Existing GM-based policy approximations for semantic observations also make the simplifying assumption that state transitions are independent of the current state. This limits their applicability to dynamic search and tracking problems, where target dynamics are often described by flexible linear time invariant models. Techniques developed in Chapter 3 relax this assumption and thus

generalize. Unlike Monte Carlo-based approximations [11] or learning-based belief compression methods [80], the approach developed in Chapter 3 provides deterministic policy approximations for a given set of 'tentpole' beliefs. However, these offline policies still require static environments and observation models, leading to the work in Chapter 5 extending [89] in adapting to unknown environments with semantic observations.

2.3.2 Offline POMDP Approximations

In addition to separating POMDP approximation algorithms out into discrete and continuous state space representations as discussed above, an even more fundamental divide exists between so called offline and online algorithms. Offline approaches are those which calculate a policy prior to run-time, such that the act of executing a policy is often reduced to a lookup table which maps belief distributions to actions.

Most offline approaches allow for "full-width" policy calculation, where policies can be applied to *any* valid belief to produce an action, not only beliefs considered prior to policy calculation. They achieve this property in a variety of ways, from the QMDP algorithm's application of whatever the current belief happens to be to a fully observable MDP policy [60], to PBVI's use of transition and observation models to evolve a representative set of reachable beliefs [69]. Some approaches such as SARSOP [56] and [91] use value-function bounds and heuristics to attempt to constrain considered beliefs. Others, such as the Perseus algorithm [95], approximate the policy using a static pre-selected belief set. Many of these algorithms also carry the "anytime" property, such that computation can be cut off at any point prior to full policy convergence return whatever the best policy found to that moment was.

One prevalent disadvantage of offline methods is their need to analytically model transition, observation, and reward functions during policy computation. While some methods [101, 11] leverage Monte-Carlo style generative models or Gaussian Mixture policies [72] to adapt well to continuous spaces, most maintain discrete state space models to retain tractability. Furthermore, although the ability to pre-compute a policy implies limitless time, in practice offline policies can

take unreasonably long to compute, from hours to weeks as scaled by complexity. These policies, once computed, are often inflexible and only useful for the very specific problem formulation for which they were solved. Even algorithms which anticipate adaptation and learning during execution [78] are limited to specified learn-able variables.

2.3.3 Online POMDP Approximations

In contrast to offline methods, online POMDP approximations calculate a policy in real-time, interleaving stages of execution and planning. For example, one popular online solver is Partially Observable Monte Carlo Planning (POMCP), shown in Algorithm 1.

Online algorithms also rely on generative "black-box" models for planning, which need not

Algorithm 1 POMCP

Function: $SEARCH$
Input: History h, Belief B
repeat
  $s \sim B(s)$
  $SIMULATE(s, h, 0)$
until TIMEOUT( )
return $argmax_a V(ha)$

Function: $SIMULATE$
Input: State s, History h, Depth d
if $d > d_{max}$ **then**
  return 0
end if
if $h \notin T$ **then**
  for $a \in A$ **do**
    $T(ha) \leftarrow (N_{init}(ha), V_{init}(ha), \emptyset)$
  end for
  return γ ESTIMATE_VALUE(s,h,d)
end if
$a \leftarrow argmax_{a'} \left[V(ha') + c\sqrt{\frac{log(N(h))}{N(ha')}} \right]$
$s' \sim p(s'|s, a), \;\; o \sim p(o|s', a), \;\; r \sim R(s, a)$
$R \leftarrow r + \gamma SIMULATE(s', hao, d + 1)$
$h \leftarrow h \cup s$
$N(h) \leftarrow N(h) + 1$
$V(ha) \leftarrow V(ha) + \frac{R - V(ha)}{N(ha)}$
return R

be analytically complete. Such generators can be used to determine a policy from individual beliefs [89], sampled state sequence 'scenarios" [92], or sampled action sequences [45, 43]. This structure allows the approximation of an optimal policy for only the current belief during execution, increasing efficiency at the expense of brittleness to model errors and risking highly subpotimal worst case behavior with high regret [31]. It also lends itself to adaptation to changing models [57], as all computation can be completed on the problem as specified at execution time. This flexibility incurs additional computation cost for embodied robots however, creating a trade-off for platforms with limited time or power.

2.3.4 Additional POMDP Resources

For readers interesting in additional resources on POMDPs in general, the books Probabilistic Robotics [102] and Decision Making under Uncertainty [53] contain excellent derivations and explorations of both fundamental and practical POMDP concepts, while [88] and [79] contain detailed surveys of Point-Based and Online algorithms respectively. For aid in the practical use of POMDPs, the POMDPs.jl library detailed in [36] implements many common POMDP algorithms in the Julia programming language.

2.4 Human-Robot Sensing and Planning

Algorithms for robotic autonomy, when applied on physically embodied agents, face several fundamental difficulties in the form of physical and computational constraints. Firstly, autonomous robotic vehicles are subject to constraints on motion, size, weight, power, and cost; this limits their computing and sensing payloads as well as their operating time and range. Secondly, the sensing and planning horizons for approximate optimal search algorithms are inherently limited. This not only restricts the ability to correctly detect and sense targets, but also the ability to execute adaptive long term information gathering strategies in complex dynamic environments. Finally, sensing platforms may only have access to imperfect/highly uncertain target behavior models. This can lead to non-Gaussian probability distributions over target states, and make online planning and

sensing/data fusion even more difficult.

Formal integration of robotic and human perception can greatly improve the efficiency and robustness of autonomous decision making, especially in situations where uncertainties cannot be well-characterized in advance and must be adapted on the fly. Soft data can be broadly related to either 'abstract' phenomena that cannot be measured by robotic sensors (e.g. labels for occupied/unoccupied rooms, object categories and behaviors) or measurable dynamical physical states that must be monitored continuously (object position, velocity, attitude, temperature, size, mass, etc.) [42]. In Chapter 4, this work examines the problem of *active soft data fusion*, and builds on methods for addressing the following key issues: (i) soft semantic data modeling for recursive state estimation; and (ii) active semantic sensing for intelligent planning under uncertainty.

2.4.1 Algorithms for Human-Machine Interaction

The application of humans as sensors to the core target tracking problems of this work provides unique challenges from both theoretical and implementation perspectives. Ultimately, the human must be modeled, and interacted with, as a multi-purpose semi-opaque probabilistic sensor. As an example, in a real world implementation a human might be able to reason about the future trajectory of the target regarding which areas have high value, or be able to discern observational deficiencies from cross-referencing camera feeds. This imperfect natural inference can help bound the search area for the robot, using information the robot may not have thought to ask for. This form of reasoning both mirrors and complements robotic logic such as recursive Bayesian estimation (discussed in the next section). Thus, the methods by which the human and robot interact would ideally allow for both the useful but imprecise nature of human sensor observations, as well as their and capacity to offer surprising or unexpected information.

While existing literature has explored the idea of learning how individual humans offer information to machines in repeated tasks [66, 77], these primarily focus on learning the parameters of a specific human collaborator. Tasks such as the motivating problem can also benefit from more generalized human models. These "plug and play" human models, as explored in [48] and [18],

allow a system to understand information originating from a non-specific human. These ideas have been explored using varying mathematical frameworks [50], and adapted to account for human factors such as cognitive load and physiological state [33]. However, such frameworks have typically focused on the data fusion aspect of human sensing. In order to incorporate human data into target tracking such that informational trajectories can be planned for it, a method of decision making which takes data fusion framework into account is also required.

A desirable characteristic of an integrated human is the ability of the robot to initiate a "Robot-Pull" event to query the human for specific information. This ability gives rise to the issue of deciding which question should be asked. Put another way, given a multitude of options regarding the various rooms, objects, and directions the robot can generate queries from, which combination leads to the greatest increase in utility? In general, this problem can be framed as a Value of Information (VOI) [49] problem. Previous work has grappled with this active sensing problem by positing a direct link between state uncertainty and utility. This leads to the implicit assumption that perfect knowledge of a target's location will lead to maximal utility through rapid interception, and the correct question to ask is that which leads to the greatest expected decrease in uncertainty. Unfortunately, even in cases where the calculation of VOI can be accomplished easily for the current timestep, determining the optimal *sequence* of questions becomes intractable due to the combinatoric increase in questions trajectories over time. Thus VOI aware planning has generally been used in myopic implementations [100], though work has been done on training machine-learning algorithms to recognize non-myopic VOI from current uncertainty levels [61]. However, even these cases still calculate VOI according to expected changes in uncertainty rather than on expected changes in the probability of success. The framework proposed in this work instead directly links information gathering with a reward function representing desired behaviors. Thus active sensing explicitly becomes a matter of choosing information gathering actions that lead most certainly and quickly to success.

Solutions to the problem of integrating the knowledge of respective members of a human-robot team can take many forms. For many tasks, humans can serve as information providers not

only about the state of the task, but also about potential policies or strategies to accomplish the task. Techniques such as Apprenticeship Learning [2] or Learning from Demonstration [44, 30], allow the human to provide examples which allow the robot to learn an appropriate policy. These examples can be direct and initiated by the human [65], or a result of robot actions to evoke a response by the human [35]. A core assumption of these methods is that the human knows what they're doing, or at the least is capable of providing useful information about the optimal policy for the task. In cases where the human cannot serve as a reasonably effective teacher, the robot must rely on other means to calculate a policy.

In tasks allowing language based communication, natural language methods enable robust communication between human and robot team members, but introduces the problem of facilitating dialog on the part of the robot. Such dialog can handled within optimal planning frameworks to help accomplish shared goals [41], and account for language uncertainties [34]. However, many previous approaches applying natural language dialog to shared tasks assume the human as a direct collaborator, assuming they will perform their own actions cooperatively or independently from the robot, or that the human is in some way supervising a task and directing the robots actions. In cases where the human acts as a sensor, rather than an on-the-ground partner or commander, there arises a need to account for both temporary unavailability and inaccuracy on the part of human, such that a robot can accomplish a task fully independently in the worse case.

2.4.2 Active Semantic Sensing for Planning with Human Collaborators

A major challenge for problems like target localization is that dynamics and uncertainties can quickly become quite non-linear and non-Gaussian, particularly given the types of semantic information available for fusion (e.g. negative information from 'no detection' readings [51]). As a result, typical stovepiped approaches to control/planning and sensing/estimation can lead to poor performance, since they rely on overly simplistic uncertainty assumptions. Constraints on human and robot performance also place premiums on when and how often collaborative data fusion can occur. For example, it is generally important to balance situational awareness and mental

workload for a human sensor (who might also need to switch between tasks constantly). Likewise, it is important for the robot to know how and when a human sensor can be exploited for solving complex planning problems, which would otherwise be very inefficient to tackle using only its own local sensor data

One approach to POMDP approximation, the QMDP algorithm [60], attempts to use a fully observable MDP policy to compute the optimal action in a partially observed step. As QMDPs are only exactly optimal assuming the state will indeed become fully observable after a single timestep, they are generally unsuitable for information gathering dependent problems such as the one addressed in this work. However, the introduction of the oracular POMDP (OPOMDP) formulation [7, 8] built on a QMDP policy and enabled the use of a human sensor to provide "perfect" state information at a fixed cost. Further work resulted in Human Observation Provider POMDPs (HOP-POMDPs) [76], which allow the consideration of oracular humans who are not always available to answer a robotic query. HOP-POMDPs calculate a cost of asking, which is then weighed against the potential information value, similar to VOI aware planning in [49]. When augmented with the Learning the Model of Humans as Observation Providers (LM-HOP) algorithm [77], HOP-POMDPs can estimate both the accuracy and availability of humans, thus treating them as probabilistic sensors. A primary drawback to using either OPOMDPs or HOP-POMDPs to address target tracking problems is that while they both enable a QMDP based policy to consider information gathering actions, these actions consist of a single self-localization query. Such formulations ignore the rich information set available in the motivating problem thanks to the presence of semantically labeled objects.

Chapter 3

Optimal Continuous State POMDP Planning with Semantic Observations: A Variational Approach

This chapter begins to address active collaborative sensing and planning in human-robot teams by first removing the human element of the problem. By simplifying to a target search scenario involving wholly robotic sensing, algorithms and techniques are developed which will later be applied to the more general human-robot case[1] .

3.1 Target Search Example with Semantic Observations

In order to ground the discussion of semantic planning, consider a 2-state CPOMDP in which an autonomous robot 'cop' attempts to localize and catch a mobile 'robber', where each moves along a single dimension (see Figure 3.1a). Here, $S = \mathbb{R} \times \mathbb{R}$ consists of $N = 2$ bounded continuous random variables at each discrete time t, $s_t = [Cop_t, Rob_t]^T$, $Cop_t \in (0, 5)$, $Rob_t \in (0, 5)$, where Cop and Rob denote the state variable for the respective agents. The robber executes a random walk,

$$p(Rob_{t+1}) = \phi(Rob_{t+1}|Rob_t, 0.5) \tag{3.1}$$

Where $\phi(s|\mu, \Sigma)$ denotes the evaluation of a Gaussian defined by mean μ and variance Σ at state s. The cop chooses a movement direction from among 3 noisy actions $A \in \{\text{left}, \text{right}, \text{stay}\}$, such

[1] This chapter was consists of material published in IEEE Transactions on Robotics [26]

that,

$$p(Cop_{t+1}|Cop_t, \text{left}) = \phi(Cop_{t+1}|Cop_t - 0.5, 0.01), \tag{3.2}$$

$$p(Cop_{t+1}|Cop_t, \text{right}) = \phi(Cop_{t+1}|Cop_t + 0.5, 0.01) \tag{3.3}$$

$$p(Cop_{t+1}|Cop_t, \text{stay}) = \delta(Cop_{t+1}, Cop_t) \tag{3.4}$$

The cop is rewarded for remaining within a set distance of the robber's position, and penalized otherwise,

$$r(|Rob_t - Cop_t| \leq 0.5) = 3, \tag{3.5}$$

$$r(|Rob_t - Cop_t| > 0.5) = -1. \tag{3.6}$$

The cop receives binary semantic observations from an onboard visual detector,

$$o_t \in \{\text{'robber detected'}, \text{'robber not detected'}\} \tag{3.7}$$

Figures 3.1b and 3.1c show unnormalized GM models for the semantic 'detection' and 'no detection' likelihoods, which are respectively parameterized by 8 and 200 isotropic Gaussian components. These models follow the specification of $O = p(o_t|s_t)$ suggested by refs. [24] and [72], and require 624 parameters total. Since it is expected that the cop will gather mostly 'no detection' observations in a typical scenario, the number of mixing components for $\alpha^i_{a,o}(s)$ will grow by a factor of at least 600 on a majority of the intermediate Bellman backup steps for offline policy approximation, as will be shown in eq. (3.19). Likewise, as will be shown in eq. (3.21) implies that the number of mixture components for $b(s_{t+1})$ will grow by a factor of at least 600 on each update of the Bayes' filter whenever the target is not detected. This example shows that, even for relatively small problems, unnormalized GM likelihood models are inconvenient for approximating or evaluating optimal policies in continuous state spaces.

3.1.1 Semantic Observation Model Synthesis and Use

As discussed in [6] and mentioned in [24], semantic observation likelihoods are ideally modeled by self-normalizing functions like the softmax model,

$$p(o_t = j|s_t) = \frac{\exp(w_j^T s_t + b_j)}{\sum_{c=1}^{N_o} \exp(w_c^T s_t + b_c)} \tag{3.8}$$

where $w_1, ..., w_{N_o} \in \mathbb{R}^N$ and $b_1, ..., b_{N_o}$ are the vector weight parameters and scalar bias parameters for each categorical outcome of o_t given s_t. In addition to ensuring $\sum_j p(o_t = j|s_t) = 1 \; \forall s_t \in \mathcal{S}(S)$, softmax functions require relatively few parameters compared to GM likelihoods, and scale well to higher dimensional spaces. Figure 3.1b shows how the cop's semantic observation likelihood can be easily modeled with a softmax function featuring 3 semantic categorical classes (two of which collectively represent the 'no detect' observation via the generalized 'multimodal softmax' (MMS) formulation [100]). Unlike the GM likelihood function approximation in Fig. 3.1c, the softmax model only requires 9 parameters.

In general, softmax parameters are easily synthesized to conform to a priori sensing geometry information and quickly calibrated with training data [100, 5]. Building on prior work, this assumes linear boundaries between softmax classes. However, since the product of Gaussian and softmax functions is analytically irreducible, modeling $p(o_t|s_t)$ via softmax functions breaks the recursive nature of the α function updates for GM-based PBVI approximations. This chapter addresses this issue in a novel way using a variational Bayes (VB) inference approximation. The VB approximation allows the product of each Gaussian term within a GM and a softmax likelihood function to be approximated as a GM. This restores the closed-form Bellman recursions for GM α approximations while keeping the resulting number of mixands in the result to a minimum. This VB approximation is inspired by a very similar technique developed in [6] to approximate eq. (3.21) for the Bayesian filtering problem, when $b(s_{t+1})$ is a GM pdf and $p(o_{t+1}|s_{t+1})$ is a softmax model.

3.2 Chapter Summary

In this chapter, the approximate VB inference technique is generalized to the dual problems of Bayesian filtering and optimal action selection under uncertainty for CPOMDPs. This technique is then extended for non-random walk transition functions to permit use of linear time-invariant (LTI) state space models in the GM-based Bellman recursions. Additionally, comparisons to the state-of-the-art online planning technique are discussed, and an analysis of planner performance under transition model discrepancies is presented.

3.3 Gaussian Mixture CPOMDPs

Finite GM models provide a general and flexible way to approximate arbitrary functions $f(s)$ of interest for CPOMDPs, where

$$f(s) = \sum_{m=1}^{G} w_m \phi(s|\mu_m, \Sigma_m) \tag{3.9}$$

is a GM defined by G weights $w_m \in \mathbb{R}_{0+}$, means $\mu_m \in \mathbb{R}^N$, and symmetric positive semi-definite covariances $\Sigma_m \in \mathbb{R}^{N\times N}$ for the multivariate normal component pdf ('mixand') $\phi(s|\mu_m, \Sigma_m)$, such that $\sum_{m=1}^{G} w_m = 1$ to ensure normalization when $f(\cdot)$ represents a pdf (this condition need not apply otherwise). Ref. [72] showed the following: let A describe a discrete action space with finite realizations a, $T = p(s_{t+1}|s_t, a) = p(s'|s, a)$ a Gaussian state transition pdf, $O = p(o_{t+1}|s_{t+1}) = p(o'|s')$ a GM observation likelihood, and $R(s_t, a_t = a) = r_a(s_t) = r_a(s)$ a continuous GM reward function for each a, such that

$$T = p(s'|s, a) = \phi(s_{t+1}|s_t + \Delta(a), \Sigma^a) \tag{3.10}$$

$$O = p(o'|s') = \sum_{l=1}^{M_o} w_l \phi(s_{t+1}|\mu_l, \Sigma_l) \tag{3.11}$$

$$R = r_a(s) = \sum_{u=1}^{M_r} w_u \phi_u(s_t|\mu_u^a, \Sigma_u^a) \tag{3.12}$$

(where $\phi(\mu, \Sigma)$ is a Gaussian with mean μ and covariance matrix Σ, and $\Delta(a)$ is the change in $s = s_t$ due to action a); then PBVI approximations to $\pi(b(s))$ can be found from closed-form GM Bellman

recursions for a finite set of GM functions $\alpha(s_t)$ defined over some initial set of GM beliefs $b(s)$. Note that $r_a(s_t)$ is generally an unnormalized GM, with possibly negative mixture weights such that $\int_{\mathcal{S}(s_t)} r_a(s_t)ds_t \neq 1$. This allows the CPOMDP to penalize certain configurations of continuous states with discrete actions and thus discourage undesirable agent behaviors. However, T must obey the usual constraints for pdfs, such that $\int_{\mathcal{S}(s_{t+1})} p(s_{t+1}|s_t, a)ds_{t+1} = 1$. The observation likelihood must also obey $\sum_{o_{t+1}} p(o_{t+1}|s_{t+1}) = 1$ for all s_{t+1}, where o_{t+1} is a discrete random variable describing a semantic observation. As such, $p(o_{t+1}|s_{t+1})$ can be a GM with strictly positive weights but unnormalized weights (i.e. which do not sum to 1) to model the conditional probability for o_{t+1}'s outcomes as a continuous function of s_{t+1}.

If the belief pdf $b(s)$ is a J-component GM,

$$b(s) = \sum_{q=1}^{J} w_q \phi(s|\mu_q, \Sigma_q), \tag{3.13}$$

then it is possible to arrive at a finite set $\Gamma_n = \{\alpha_n^1, \alpha_n^2, ...\alpha_n^{N_\alpha}\}$ of $\alpha(s)$ functions for an n-step look-ahead decision starting from $b(s)$, such that

$$\alpha_n^i(s) = \sum_{k=1}^{M} w_k^i \phi(s|\mu_k^i, \Sigma_k^i) \quad (\alpha_n^i \in \Gamma_n) \tag{3.14}$$

and the optimal value function $V_n^*(b(s))$ is approximately

$$V^*(b(s)) \approx \max_{\alpha_n^i} < \alpha_n^i, b(s) >, \tag{3.15}$$

$$< \alpha_n^i, b(s) >=$$

$$\int_{\mathcal{S}(S)} \left[\sum_{k=1}^{M} w_k^i \phi(s|\mu_k^i, \Sigma_k^i)\right] \left[\sum_{q=1}^{J} w_q \phi(s|\mu_q, \Sigma_q)\right] ds, \tag{3.16}$$

$$= \sum_{k,q}^{M\times J} w_k^i w_q \phi(\mu_q|\mu_k^i, \Sigma_q + \Sigma_k^i) \int_{\mathcal{S}(S)} \phi(s|c_1, c_2)ds, \tag{3.17}$$

$$= \sum_{k,q}^{M\times J} w_k^i w_q \phi(\mu_q|\mu_k^i, \Sigma_q + \Sigma_k^i), \tag{3.18}$$

$$c_2 = [(\Sigma_k^i)^{-1} + (\Sigma_q)^{-1}]^{-1},$$

$$c_1 = c_2[(\Sigma_k^i)^{-1}\mu_k^i + (\Sigma_r)^{-1}\mu_q],$$

which follows from the fact that the product of two Gaussian functions is another Gaussian function. The n-step look-ahead approximation is commonly in PBVI methods, where n is large enough such that V_n^* does not change appreciably and thus converges closely to the infinite horizon V^*.

The $\alpha_n^i \in \Gamma_n$ functions are computed using n-step policy rollouts, starting from N_B different initial GM beliefs $\mathcal{B}_0 = \{b_1(s), ..., b_{N_B}(s)\}$. In each step, for each $b \in \mathcal{B}_0$, each α_{n-1}^i function's value is updated via the 'Bellman backup' equations, which perform point-wise value iteration to capture the effects of all possible observations and actions on the accumulated expected reward for future time steps $0, ..., n$. These lead to the recursions

$$\alpha_{a,j}^i(s) = \int_{s'} \alpha_{n-1}^i(s') p(o' = j|s') p(s'|s, a) ds', \tag{3.19}$$

$$\alpha_n^i(s) = r_a(s) + \gamma \sum_{j=1}^{N_o} \arg\max_{\alpha_{a,j}^i} (< \alpha_{a,j}^i, b(s) >), \tag{3.20}$$

where $\alpha_{a,j}^i(s)$ is an intermediate function corresponding to a value for a given action-observation pair (a, j) at step n, and $\alpha_n^i(s)$ is the discounted marginalization over all observations of the intermediate function that maximizes the belief being backed up, summed with the reward function. The action then associated with each α_n^i is the one which maximized the value marginalized over observations. Since $p(s_{t+1}|s_t, a)$ is Gaussian, $p(o_{t+1}|s_{t+1})$ is a GM, and $r_a(s_t)$ is a GM for each action a, the Bellman backups yield closed-form GM functions for $\alpha_n^i(s_t)$. Since the GM function for $r_a(s_t)$ can have negative weights and values, it follows that each GM function $\alpha_n^i(s_t)$ can also take on negative weights and values.

This GM formulation scales well for continuous state spaces where $N \geq 2$, and naturally handles highly non-Gaussian beliefs $b(s)$ stemming from non-linear/non-Gaussian state process and observation models in a deterministic manner. In contrast to approximations that discretize S to transform the CPOMDP into a standard discrete state POMDP (and thus scale badly for large N), the complexity of the CPOMDP policy (i.e. the required number of mixture terms for each $\alpha_n^i(s)$) depends only on the complexity of the dynamics for the state belief pdf $b(s)$, rather than the number of continuous states N. Furthermore, since the Bellman backup equations can be

performed entirely offline using a set of initial beliefs $\mathcal{B}_0$, the resulting policy induced by the final set of $\alpha_n^i(s)$ functions can be quickly and easily computed online: as the agent obtains new state beliefs $b(s_t) \rightarrow b(s_{t+1})$ over time via the standard Bayes' filter equations (which yield the general Gauss sum filter in this case),

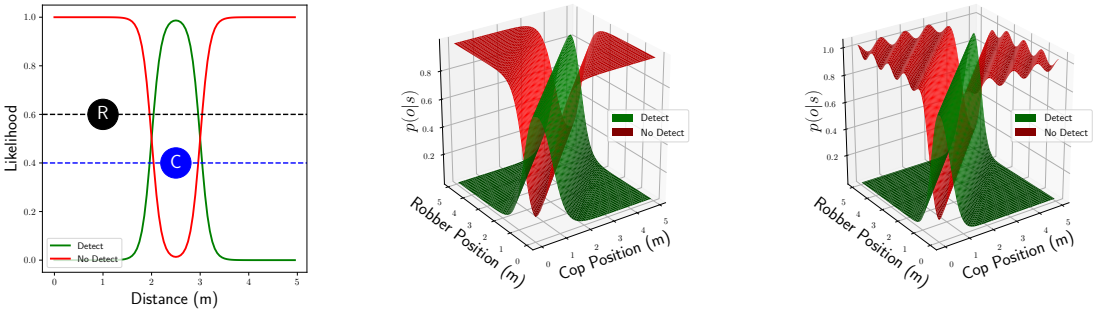
$$b(s_{t+1}) \propto p(o_{t+1}|s_{t+1}) \int_{S(s_t)} p(s_{t+1}|s_t, a) b(s_t) ds_t, \tag{3.21}$$

the optimal action a to take for $b(s_{t+1})$ is the one associated with the $\alpha_n^i \in \Gamma_n$ satisfying $\arg\max_{\alpha_n^i} < (\alpha_n^i), b(s_{t+1}) >$.

3.4 Limitations for Hybrid Continuous-Discrete Reasoning

The likelihood model $O = p(o_t|s_t)$ must describe a valid hybrid (continuous-discrete) probability distribution, such that $\sum_{o_t} p(o_t|s_t) = 1 \ \forall s \in \mathcal{S}(S)$. For each possible outcome o_t, this is typically modeled by an unnormalized GM [72], $p(o_t|s_t) \approx \sum_{l_o=1}^{L_o} w_o \phi(s_t|\mu_{l_o}, \Sigma_{l_o})$, such that $\sum_{o_t} p(o_t|s_t) \approx 1$ everywhere. Such models preserve the closed-form Bellman updates required for PBVI, but are difficult and labor intensive to specify. In particular, for state dimension $N \geq 2$, L_o must be very large for each possible o_t outcome to ensure that the normalization requirement is satisfied for all s_t and that the desired probabilities $p(o_t|s_t)$ are modeled accurately. This effectively turns $p(o_t|s_t)$ into a 'soft grid' discretization and severely restricts the scalability of GM policy approximation.

Another issue is the fact that the number of multiplications and summations in each step of the Bellman recursions (3.19)-(3.20) grows with the number of resulting GM components from J mixands in $b(s)$ and K mixands in $\alpha_n^i(s)$ to $K \times J$ mixands. The number of terms in $\alpha_n^i(s)$ also grows with each summation over N_o observations. GM condensation methods are thus needed to control the size of the policy functions $\alpha_n^i(s)$ between backup steps for offline policy approximation, as well as the size of the state belief GM pdf $b(s)$ between Bayes' filter updates for online policy evaluation. Refs. [24, 72] propose different general methods for condensing GM functions in CPOMDP policy approximations, although in principle any number of GM merging algorithms developed in the



(a) Colinear Model (b) Softmax Likelihood Model (c) GM No Detect

Figure 3.1: A model for two Colinear Robots, one a cop and the other a robber. The cop's robber detection observation likelihood can be modeled as a 9 parameter MMS model (b), or a 624 parameter Gaussian Mixture model (c).

target tracking and data fusion literature could also be applied [106, 81]. However, for large-scale problems such as dynamic target search and tracking, it is not uncommon for offline Bellman backups and online policy evaluations to rapidly produce hundreds or even thousands of new mixands in just one backup step or Bayes' filter prediction/measurement update. As discussed in Sec. III.B, existing GM merging methods tend to be computationally expensive and slow for such large mixtures. This issue is exacerbated by use of dense unnormalized GM models for $p(o_t|s_t)$, which introduce additional policy approximation errors if normalization is not guaranteed for all $s_t \in \mathcal{S}(S)$. Offline policy approximation and online policy evaluation thus become expensive.

Existing GM-based CPOMDP approximation methods also use simplified random walk state transition models $p(s_{t+1}|s_t, a)$ where s_{t+1} is modeled as the result of shifting s_t by a specific distance in $\mathcal{S}(S)$ for a given action a plus some additional random noise component. It is thus not obvious how these methods can be used with more sophisticated state dynamics models, e.g. kinematic linear Nearly Constant Velocity (NCV) models commonly used in dynamic target tracking [12], where the position components of s_{t+1} depend on velocity components in s_t. This severely limits the applications for GM-based CPOMDP approximations.

3.5 Variational PBVI for Softmax Semantic Likelihoods

To use softmax $p(o'|s')$ functions in the GM-based PBVI CPOMDP policy approximation described in eqs. (3.19)-(3.20), the local VB approximation for hybrid inference with softmax models developed in [6] is used to approximate the product of a softmax model and a GM as a variational GM,

$$\begin{aligned}
&\alpha_{n-1}^i(s')\ p(o'=j|s') \\
&\quad = \left[\sum_{k=1}^{M} w_k^i \phi(s'|\mu_k^i, \Sigma_k^i)\right]\left[\frac{\exp(w_j^T s' + b_j)}{\sum_{c=1}^{N_o}\exp(w_c^T s' + b_c)}\right] \\
&\quad \approx \sum_{h=1}^{M} w_h \phi(s'|\mu_h, \Sigma_h)
\end{aligned} \tag{3.22}$$

Figure 3.2 shows the key idea behind this VB approximation using a toy 1D problem. The softmax function (blue curve, e.g. representing $p(o'|s')$ in (3.22)) is approximated by a lower bounding variational Gaussian function (black curve). The variational Gaussian is optimized to ensure the product with another Gaussian function (green, e.g. representing a single mixand of α_{n-1}^i) results in a good Gaussian approximation (red dots) to the true non-Gaussian (but unimodal) product of the original softmax function and Gaussian functions (solid magenta). More formally, the VB update derived in [6] for approximating the product of a normalized Gaussian (mixture) pdf $p(s')$ and a softmax function $p(o'|s')$ can be adapted and generalized to approximate the product of an **unnormalized** Gaussian (mixture) $\alpha_{n-1}^i(s')$ (from the intermediate Bellman backup steps) and softmax likelihood. In the first case, consider the posterior Bayesian pdf for a Gaussian prior $p(s')$ given $o' = j$,

$$p(s'|o') = \frac{p(s')p(o'|s')}{p(o')} = \frac{1}{C}\phi(s'|\mu, \Sigma)\frac{\exp(w_j^T s' + b_j)}{\sum_{c=1}^{N_o}\exp(w_c^T s' + b_c)}$$

$$C = \int_{-\infty}^{\infty} \phi(s'|\mu, \Sigma)\frac{\exp(w_j^T s' + b_j)}{\sum_{c=1}^{N_o}\exp(w_c^T s' + b_c)} ds'$$

By approximating the softmax likelihood function as an unnormalized variational Gaussian function

$f(o', s')$, the joint pdf and normalization constant C can be approximated as:

$$p(s', o') \approx \hat{p}(s', o') = p(s')f(o', s')$$
$$C \approx \hat{C} = \int_{-\infty}^{\infty} \hat{p}(s', o')ds'.$$

The key trick here is that, for any $j \in \Omega$, it is possible to ensure $f(o' = j, s') \leq p(o' = j|s')$ by construction [6], using the variational parameters y_c, γ, and ξ_c such that

$$f(o' = j, s') = \exp\left\{g_j + h_j^T s' - \frac{1}{2}s'^T K_j s'\right\} \tag{3.23}$$

$$g_j = \frac{1}{2}[b_j - \sum_{c \neq j} b_c] + \gamma(\frac{N_o}{2} - 1) + \sum_{c=1}^{N_o} \frac{\xi_c}{2} + \lambda(\xi_c)[\xi_c^2 - (b_c - \alpha)^2] - \log(1 + \exp\{\xi_c\}) \tag{3.24}$$

$$h_j = \frac{1}{2}[w_j - \sum_{c \neq j} w_c] + 2\sum_{c=1}^{N_o} \lambda(\xi_c)(\alpha - b_c)w_c \tag{3.25}$$

$$K_j = 2\sum_{c=1}^{N_o} \lambda(\xi_c)w_c w_c^T \tag{3.26}$$

Since $f(o' = j, s') \leq p(o' = j|s')$ for any choice of the variational parameters, it follows that $\hat{C} \leq C$. As such, the variational parameters which produce the tightest lower bound $\hat{C}$ can be found through an iterative expectation-maximization algorithm, which requires alternately re-estimating $\hat{p}(s'|o')$ given new values of the variational parameters, and then re-computing the variational parameters based on new expected values of s' from $\hat{p}(s'|o')$. Upon convergence of $\hat{C}$ to a global maximum, the product $p(s', o' = j) = p(s')p(o' = j|s')$ becomes well-approximated by the product $\hat{p}(s', o' = j) = p(s')f(o' = j|s')$, which is another (unnormalized) Gaussian,

$$\hat{p}(s', o' = j) = \exp\left\{(g_p + g_j) + (h_p + h_j)s' - \frac{1}{2}s'^T(K_p + K_j)s'\right\}. \tag{3.27}$$

Normalizing this joint distribution by $\hat{C}$ gives the posterior Gaussian pdf approximation $\hat{p}(s'|o') = \phi(s'|\mu_h, \Sigma_h)$. The approximation of the product of a GM pdf with a softmax model follows imme-

Algorithm 2 VB-POMDP Backup

Input: $b \in B_0$, Γ_{n-1}, $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma\}$
$\Gamma_n = \emptyset$
for each $\alpha_{n-1} \in \Gamma_{n-1}$,$a \in A$, $j \in \Omega$ **do**
  $\alpha_{a,j}(s) \leftarrow \sum_h w_h \phi(s|\mu_h - \Delta(a), \Sigma^a + \Sigma_h)$
  $\alpha_n(s) = r_a(s) + \gamma \sum_j \arg\max_{\alpha_{a,j}}(< \alpha_{a,j}, b >)$
  $\Gamma_n = \Gamma_n \cup \alpha_n(s)$
end for
return: Γ_n

diately from fact that this product is a sum of weighted products of individual Gaussians with the softmax model, where each individual product term can be approximated via variational Bayes.

This approximation is now adapted to the case where the 'prior' GM pdf over s' is an unnormalized GM function $\alpha^i_{n-1}(s')$. The results from the above derivation must simply be multiplied by the normalizing constant $\hat{C}$ to obtain the approximate joint $\hat{p}(s', o' = j)$ for each mixture term instead. This allows eq. (3.19) for the intermediate α function update in the PBVI backup to be approximated as

$$\alpha^i_{a,j}(s) = \int_{s'} \alpha^i_{n-1}(s') p(o' = j|s') p(s'|s, a) ds' \tag{3.28}$$

$$= \int_{s'} \left[\sum_{k=1}^{M} w^i_k \phi(s'|\mu^i_k, \Sigma^i_k) \right] \left[\frac{\exp(w_j^T s' + b_j)}{\sum_{c=1}^{N_o} \exp(w_c^T s' + b_c)} \right] \times \left[\phi(s'|s + \Delta(a), \Sigma^a) \right] ds', \tag{3.29}$$

$$\approx \int_{s'} \sum_{h=1}^{M} w_h \phi(s|\hat{\mu}_h, \hat{\Sigma}_h) \phi(s'|s + \Delta(a), \Sigma^a) ds' \tag{3.30}$$

$$\rightarrow \alpha^i_{a,j}(s) \approx \sum_{h=1}^{M} w_h \phi(s|\hat{\mu}_h - \Delta(a), \hat{\Sigma}_h + \Sigma^a), \tag{3.31}$$

where $(\Delta(a), \Sigma^a)$ are known constants for each a. In practice, the intermediate $\alpha_{a,j}(s)$ functions do not depend on the belief being backed up, and can therefore be calculated once per iteration over all beliefs. Algorithm 2 summarizes how the GM-based PBVI updates developed in Section II are thus modified to use the VB approximation for softmax semantic observation likelihoods. The VB algorithm does introduce some approximation error. The possibility of using the bound on the log-likelihood to estimate this error across backups will be explored in future work.

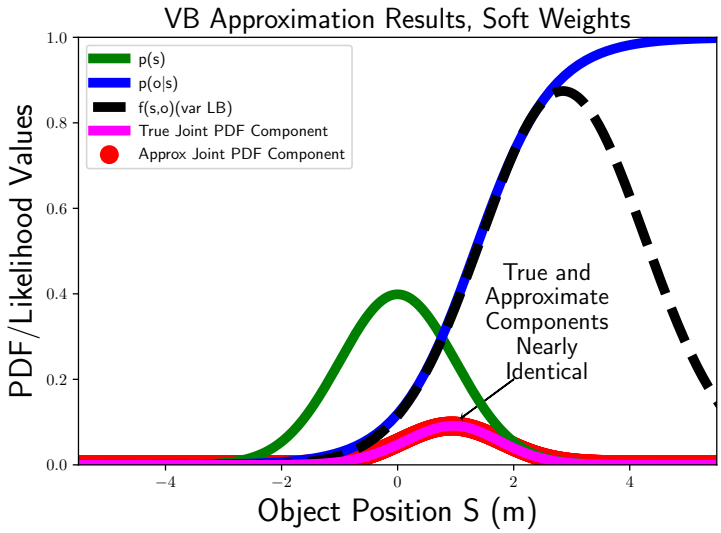


Figure 3.2: 1D VB approximation example: approximate pdf-likelihood product (red) aligns closely with true product (pink).

Following [6], recursive semantic observation updates to GM $b(s_{t+1})$ pdfs can also be carried out online during execution of these policies using softmax likelihoods with the VB approximation, as shown in Fig. 3.3,

$$b(s_{t+1}) \propto p(o_{t+1} = j|s_{t+1}) \int_{s_t} p(s_{t+1}|s_t, a) b(s_t) ds_t \tag{3.32}$$

$$= \left[\sum_{q=1}^{J} w_q \phi(s_{t+1}|\mu_q + \Delta(a), \Sigma^a + \Sigma_q) \right] \times \left[\frac{\exp(w_j^T s_{t+1} + b_j)}{\sum_{c=1}^{N_o} \exp(w_c^T s_{t+1} + b_c)} \right]$$

$$\approx \sum_{q=1}^{J} w_q \phi(s_{t+1}|\mu_q, \Sigma_q). \tag{3.33}$$

In this example, the resulting posterior GM pdf for the 'no detection' update has only 4 components [2] , thus demonstrating that parametrically simpler softmax models drastically reduce the complexity of inference compared to unnormalized GM likelihood functions.

[2] the 2 prior components in this example are evaluated against separate categories for 'no detection left' and 'no detection right', which together make up a non-convex 'no detection' semantic observation class via an MMS model

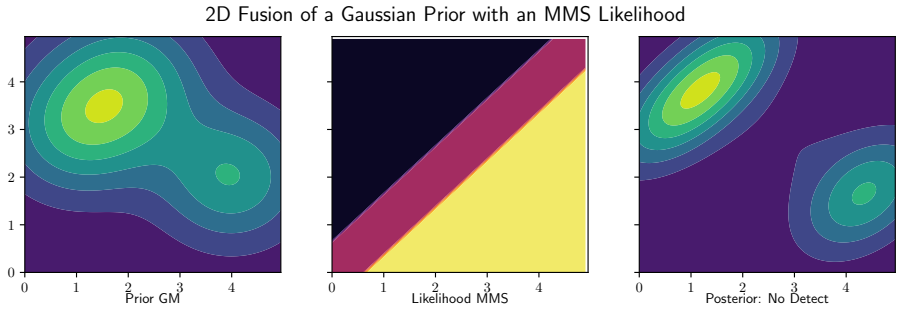


Figure 3.3: GM belief update with a MMS observation likelihood model. The negative observation of "No Detection" causes the posterior to split further into a bimodal distribution.

3.5.1 Bellman Backups with Arbitrary LTI State Dynamics

As in previous work in [72], the Bellman backups used so far assume random walk state transitions from s_t to s_{t+1}. On the other hand, many problems such as target search and tracking require modeling more sophisticated dynamic behaviors, e.g. via linear time-invariant (LTI) state space models. In discrete time, such dynamics are represented by a state transition matrix (STM) $F \in \mathbb{R}^{N \times N}$ and action effect $\Delta(a_t) \in \mathbb{R}^N$, such that

$$s_{t+1} = Fs_t + \Delta(a_t).$$

In this case, the state transition pdf takes the form,

$$T = \phi(s_{t+1}|Fs_t + \Delta(a_t), \Sigma^a).$$

Using this altered transition model to re-derive eq. (3.31), the new intermediate alphas are

$$\alpha^i_{a,o}(s) \approx \sum_{h=1}^{M} w_h \phi(Fs|\hat{\mu}_h - \Delta(a_t), \hat{\Sigma}_h + \Sigma^a).$$

However, CPOMDP policy approximation requires that the α functions depend on the 'current' state s, not on mappings of s. Therefore the Gaussian dependency on Fs must be converted to a dependency on s,

$$\phi(Fs|\mu, \Sigma) \propto \phi(s|\tilde{\mu}, \tilde{\Sigma})$$

Expanding the left hand side of this last expression,

$$\phi(Fs|\mu,\Sigma) = \\ |2\pi\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(Fs-\mu)^T\Sigma^{-1}(Fs-\mu)\right)$$

the STM can be factored within the exponential[3] ,

$$\phi(Fs|\mu,\Sigma) = \\ |2\pi\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(s-F^{-1}\mu)^T F^T\Sigma^{-1}F(s-F^{-1}\mu)\right).$$

The exponential term then resembles a Gaussian,

$$\phi(s|\tilde{\mu},\tilde{\Sigma}), \quad \tilde{\mu} = F^{-1}\mu, \quad \tilde{\Sigma} = F^{-1}\Sigma F^{-T}.$$

To address the normalization in front of the exponential, a weighting term ω is introduced,

$$\omega = |F^{-1}F^{-T}|^{-\frac{1}{2}}$$

Multiplying and dividing by ω gives

$$\begin{aligned}&\phi(Fs|\mu,\Sigma) = \\ &\frac{\omega}{\omega}|2\pi\Sigma|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(s-F^{-1}\mu)^T F^T\Sigma^{-1}F(s-F^{-1}\mu)\right) \\ &= \frac{1}{\omega}|2\pi F^{-1}\Sigma F^{-T}|^{-\frac{1}{2}} \\ &\quad \times \exp\left(-\frac{1}{2}(s-F^{-1}\mu)^T F^T\Sigma^{-1}F(s-F^{-1}\mu)\right).\end{aligned}$$

Finally, a weighted Gaussian can be recognized as

$$\phi(Fs|\mu,\Sigma) = \frac{1}{\omega}\phi(s|\tilde{\mu},\tilde{\Sigma}) \\ \rightarrow \alpha_{a,o}(s) \approx \sum_{h=1}^{M} w_h \cdot \frac{1}{\omega} \cdot \phi(s|\tilde{\mu}_{h,a},\tilde{\Sigma}_a)$$

These equations reduce to the original VB-POMDP backup equations when $F = I$ (identity). This transformation can also be applied to the original CPOMDP approximation with unnormalized GM observation models [72].

[3] assuming F is invertible; this is always the case for LTI systems since F comes from the corresponding matrix exponential

3.6 Clustering-based GM Condensation

The number of GM mixands for α functions and $b(s)$ can still become significantly large over iterations/time even with the VB approximation. This section describes a novel GM condensation algorithm to help reduce the computational overhead and enable faster policy computation and online belief updates. Numerical studies comparing the effectiveness of different Gaussian clustering metrics are also presented, showing that a Euclidean distance measure between Gaussian means provides the best overall balance between computational speed and accuracy in terms of speeding up the widely used Runnalls' condensation algorithm [81] for large GMs. The Runnalls' algorithm uses upper bounds on the Kullback-Leibler divergences between uncondensed GMs and condensed GMs to select successive pairs of mixands for mixture moment-preserving mergers, and as such is better able to retain information from uncondensed GMs compared to other similar condensation methods [83, 106] while also requiring little additional computational overhead.

3.6.1 Clustering-based Condensation Algorithm

To remain computationally tractable, the GMs representing each α function must also be condensed such that,

$$\alpha_n^i = \sum_{k=1}^{M} w_k \phi(s|\mu_k, \Sigma_k) \approx \hat{\alpha}_n^i = \sum_{k=1}^{\tilde{M}} \hat{w}_k \phi(s|\hat{\mu}_k, \hat{\Sigma}_k),$$

where $\tilde{M} < M$ (mixture terms in $b(s)$ must also be compressed following dynamics prediction and Bayesian observation updates). Existing GM condensation algorithms perform myopic pairwise merging of the M components in α_n^i, such that the resulting $\tilde{M}$ components in $\hat{\alpha_n^i}$ minimize some information loss metric [72, 24]. Naïve pairwise merging tends to be very expensive and slow when $M \geq 100$ (which is often the case for long horizon Bellman recursions with $N \geq 2$).

To improve condensation speed, a novel 'divide and conquer' strategy is employed which first pre-classifies the mixture indices into K local clusters (submixtures), and then condenses each cluster to some pre-determined number of components ψ via pairwise merging, before recombining the results to a condensed mixture with the desired size $\tilde{M} < M$. For merging within submixture

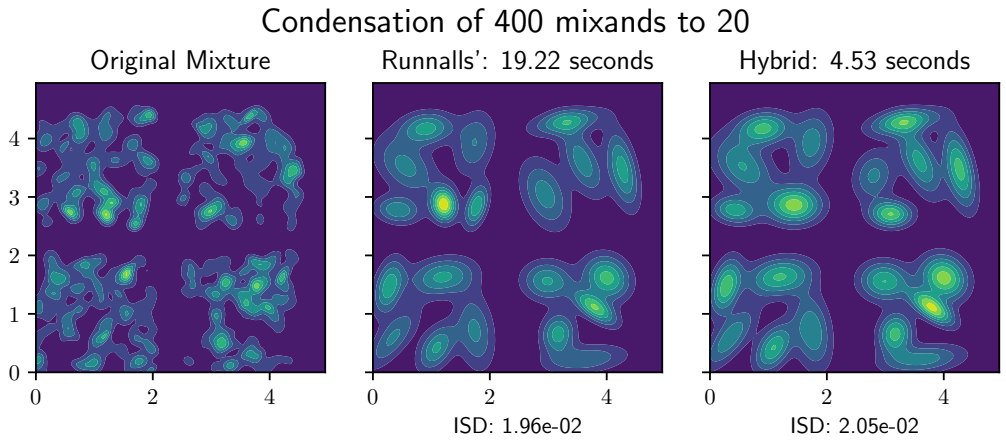


Figure 3.4: Condensation comparison of Runnalls' method to pre-clustering hybrid method: an initial mixture of 400 mixands is condensed to 20 mixands; the hybrid method results in a similar ISD as Runnalls' alone, but significantly faster.

clusters, the Runnalls' algorithm [81] is used, which uses an upper bound on the KL divergence between the pre-merge and post-merge submixture to select the least dissimilar component pairs merging. This process is outlined in Algorithm 3.

Since submixtures may have different pre-condensation sizes depending on the clustering method used, this approach is prone to overcondensation when each submixture is naively condensed to the same final size. To avoid this, each submixture is condensed according to the proportion of mixands it contains with respect to the original mixture. This means a submixture containing h mixands would be condensed to $\psi = floor(\frac{h\tilde{M}}{M})$. This can still result in overcondensation if $\frac{h\tilde{M}}{M}$ is not an integer for at least one submixture, but the difference between the desired size and the resulting size is strictly upper-bounded by the chosen number of submixtures.

Empirical tests indicate that this new hybrid method achieves approximately the same accuracy for condensation performance as classical full scale pairwise merging, although the hybrid method is considerably cheaper and faster (e.g. 22.16 secs vs. 5.69 secs for $M = 400 \rightarrow \tilde{M} = 20$ with $N = 2$, in Python on a 2.6 GHz Intel i7 processor running Windows 10 with 16 GB of RAM). Figure 3.4 shows a comparison of the classical full-mixture Runnalls' condensation method to our hybrid cluster-then-condense method for a GM with $M = 400$ components, with $K = 4$ and $\psi = 5$. The Integral Square Difference (ISD) metric [106] is used to assess the accuracy of each method,

Algorithm 3 Clustering-Based Condensation Algorithm

Input: Mixture, K, ψ
Clusters = K-means(Mixture,K);
Create empty NewMixture;
for $C \in$ Clusters **do**
  $\hat{C} = \text{Runnalls}\left(C, floor\left(\frac{\text{size}(C)K\psi}{\text{size(Mixture)}}\right)\right)$;
  NewMixture.add($\hat{C}$);
end for
return NewMixture;
Subfunction: Runnalls
Input: C, max
while size(C) $>$ max **do**
  for unnormalized Gaussians $G_i, G_j \in C$ **do**
    $[w_{i,j}, \mu_{i,j}, \Sigma_{i,j}]$ =Merge(G_i, G_j);
    Compute KL divergence upper bound:
    $B_{ij} = \frac{1}{2}[(w_i + w_j)\log|\Sigma_{i,j}| - w_i \log|\Sigma_i| - w_j \log|\Sigma_j|]$;
    $G_i = \text{Merge}(G_i, G_j)$, where $(i,j) = \arg\min B_{ij}$;
    C.remove(G_j);
  end for
end while
return C;
Subfunction: Merge
Input: G_i, G_j
$w_m = w_i + w_j$
$\mu_m = \frac{w_i}{w_m}\mu_i + \frac{w_j}{w_m}\mu_j$
$\Sigma_m = \frac{w_i}{w_m}\Sigma_i + \frac{w_j}{w_m}\Sigma_j + \frac{w_i w_j}{w_m}(\mu_i - \mu_j)(\mu_i - \mu_j)^T$
return w_m, μ_m, Σ_m

where, given two GMs $GM_h(s)$ and $GM_r(s)$,

$$
\begin{aligned}
&ISD[GM_h(s), GM_r(s)] \\
&\quad = \int_{\mathcal{S}(S)} (GM_h(s) - GM_r(s))^2 ds = J_{hh} - 2J_{hr} + J_{rr}, \\
&J_{hh} = \sum_{i=1}^{N_h} \sum_{j=1}^{N_h} w_i w_j \phi(\mu_i | \mu_j, \Sigma_i + \Sigma_j), \\
&J_{hr} = \sum_{i=1}^{N_h} \sum_{j=1}^{N_r} w_i w_j \phi(\mu_i | \mu_j, \Sigma_i + \Sigma_j), \\
&J_{rr} = \sum_{i=1}^{N_r} \sum_{j=1}^{N_r} w_i w_j \phi(\mu_i | \mu_j, \Sigma_i + \Sigma_j).
\end{aligned}
$$

This example indicates that both methods result in condensed GMs that have approximately the same ISD compared with the original GM, although the hybrid cluster-then-condense method is considerably faster.

3.6.2 Empirical Clustering Metric Comparisons

Theoretically, the cluster-then-merge approach is natural to consider, since any GM can be generally viewed a 'mixture of local submixtures'. From this standpoint, mixture components belonging to different local submixtures are unlikely to be directly merged in a pairwise global condensation algorithm, whereas those belonging to the same submixture are more likely to be merged. The global merging operation can then be broken up into several smaller parallel merging operations within each submixture. In our initial approach, the submixtures are identified using a simple fast k-means clustering heuristic on the component means. Additional work verifies the robustness of this method for general problem settings, and other techniques for identifying submixture groups could also be used (e.g. to also account for mixand covariances, etc.).

The k-means clustering heuristic employed in the example above utilized the Euclidean distance between mixand means. While this metric results in simple fast clustering, it also underutilizes the information available. Alternative techniques for clustering were therefore also evaluated; these take into account additional information, specifically mixand covariances, with the goal

of finding a method that performs with an improved level of accuracy without sacrificing too much of the speed achieved by the Euclidean distance between means. Five methods in total were considered for submixture formation: four alternative pdf distance measures and the original Euclidean distance heuristic. Each alternative method chosen has a closed form derivation for normalized Gaussian pdfs, and utilizes only the mixand mean and covariance. Weights are considered within the second part of the procedure when Runnalls' method is used to combine similar mixands.

The first alternative distance is the symmetric Kullback-Leibler divergence (KLD), which measures the difference in expectation between two distributions. The symmetric Kullback-Leibler divergence is defined for two normal distributions G_i and G_j as

$$D_{symKL} = \frac{KLD(G_i||G_j) + KLD(G_j||G_i)}{2}$$

Next, the Jensen-Shannon divergence is considered. The Jensen-Shannon divergence is a symmetric and smoothed version of KLD that uses an average of the two distributions G_i and G_j,

$$JSD(G_i||G_j) = \frac{1}{2}KLD(G_i||M) + \frac{1}{2}KLD(G_j||M)$$
$$\text{where} \quad M = \frac{1}{2}(G_i + G_j)$$

The 2-Wasserstein distance, sometimes referred to as the Earth Mover's Distance (EMD), is a measure of the minimum cost of turning one distribution into the other, factoring in both distance between distributions and the probability mass of each. The 2-Wasserstein distance is defined as

$$W_2(G_i, G_j)^2 = ||\mu_i - \mu_j||_2^2$$
$$+ Tr(\Sigma_i + \Sigma_j - 2(\Sigma_j^{1/2}\Sigma_i\Sigma_j^{1/2})^{1/2})$$

Finally the Bhattacharyya distance is considered, which measures overlap between two distributions and is also closely related to the Hellinger divergence. This takes into account both distance between means and similarity of covariances. The Bhattacharyya distance is defined as

$$D_B = \frac{1}{8}(\mu_1 - \mu_2)^T\Sigma^{-1}(\mu_1 - \mu_2) + \frac{1}{2}\log\left(\frac{|\Sigma|}{\sqrt{|\Sigma_1||\Sigma_2|}}\right)$$
$$\text{where} \quad \Sigma = \frac{\Sigma_1 + \Sigma_2}{2}$$

To more directly compare tests of different dimensions, starting sizes, and ending sizes, here we use the normalized version of the Integrated Squared Difference metric. The normalized ISD [109] constrains each measurement to a range $NISD \in [0, 1]$, and is derived from the ISD definition as

$$NISD[GM_h, GM_r] = \sqrt{\frac{ISD[GM_h, GM_r]}{(J_{hh} + J_{rr})}} \tag{3.34}$$

Test mixtures in $N = 1, 2$, and 4 dimensions were generated by sampling means from a uniform distribution from 0 to 10 on $\mathbb{R}^N$, sampling covariances from a Wishart distribution with N degrees of freedom and a matrix prior of identity scaled by a factor of 2, and sampling weights from a uniform distribution from 0 to 1. Each combination of dimensionality, clustering method, number of starting mixands, number of clusters, and final mixture size was repeated on ten different randomly generated mixtures. The time for clustering and condensation, and the accuracy of clustering and condensation, characterized by the normalized ISD between the starting and final mixtures, were recorded. Additionally, Runnalls' method without clustering was used as an state-of-the-art baseline for accuracy, and time and normalized ISD for Runnalls' were recorded. The time and normalized ISD results for each distance measure were then able to be compared to one another and to the time and normalized ISD results achieved using Runnalls' method. The results were obtained in Python on a 3.3 GHz Intel i7 processor running Ubuntu 16.04, with 32 GB of RAM.

Table 3.1 presents the results for time of clustering and condensation and the accuracy of each method as a percentage of the Runnalls time and accuracy, averaged across all parameters barring dimension. These results are also presented graphically in Fig. 3.5. In general, the alternative methods compared poorly to Euclidean distance in the accuracy vs. speed trade-off. In higher dimensions, the alternative methods tended to heavily favor circular or hyper-spherical clusters, leading to suboptimal clustering in mixtures with elongated high density regions. This combined with the additional overhead needed to compute the alternative metrics led to the Euclidean distance measure consistently providing the best balance of accuracy vs. speed, particularly in higher dimensions. Therefore, the Euclidean distance is used in the remainder of this work.

	Sym KLD	JSD	Euclid	EMD	Bhatt
	Norm. ISD ratio				
Dimension					
1	0.4527	0.6656	1.0666	1.0675	0.6067
2	1.7338	2.0567	1.9774	2.0408	1.9666
4	1.4089	2.1097	1.7130	5.0584	2.4256
	Time ratio				
1	2.3470	1.5198	0.1783	0.7512	0.9989
2	5.0418	2.0694	0.1725	0.6476	1.5415
4	14.6303	6.8890	0.1835	2.8365	4.2750

Table 3.1: Time and accuracy versus Runnalls' method

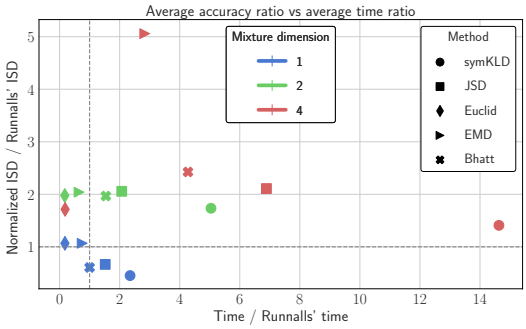


Figure 3.5: Time and Normalized ISD of clustering-based condensation compared to Runnalls' method without clustering.

3.7 VB-POMDP Results and Analysis

This section examines application of VB-POMDP to 3 simulated versions of the target search and localization problem (with $N = 2, 4$), as well as a 5-robot simultaneous localization and navigation problem ($N = 10$). VB-POMDP's performance on the target search application is compared to the performance of other state of the art policy approximation methods. The clustering-based GM condensation method is used for all scenarios and with all policy approximations.

3.7.1 Colinear Cop/Robber Results

Table 3.2 compares the resulting average final rewards achieved over a 100 step simulation for 100 simulation runs, using policy approximations for the 1D cop-robot search problem presented earlier in Sec.3.1. The second column shows the average final rewards the proposed VB-POMDP method (with the softmax likelihood model shown in Fig. 3.1b), while the first column shows the average final rewards obtained for the GM-POMDP policy approximation of [72] (using the GM observation models shown in Figs. 3.1c). Both methods used the hybrid GM clustering technique introduced in Section III.B. Results for a third greedy one-step implementation of the latter approximation are also shown in the third column. While a greedy policy approximation is generally expected to be suboptimal, it provides a realistic minimum implementation cost baseline result for use on a robotic platform, and also provides an indication of the problem's difficulty (i.e.

Co-linear Search Results

Method	Mean Reward	Standard Deviation
VB-POMDP	57	±54
GM-POMDP	59	±30
Greedy	19	±57

Table 3.2: Rewards achieved on basic co-linear target search problem (standard deviations over 100 simulation runs shown).

in this case, in a single dimension).

All methods were compared pair-wise using the Student's t-test for the difference of two means, with 100 samples each. Statistically, the VB-POMDP policy approximation average performance could not be differentiated from the baseline GM-POMDP policy, with $p > 0.05$. However, both policies achieved a significantly higher average accumulated reward than the comparison greedy approach, with $p < 0.05$. These results indicate that the VB-POMDP approximation performs as well as the GM-POMDP approximation on this problem. The VB approximations described earlier therefore do not lead to any significant compromises in optimality for this problem compared to the state of the art.

3.7.2 2D Random Walk Robber Results

Extending the colinear search problem, the cop robot now attempts to localize and intercept the robber in a bounded 2D space $S = \mathbb{R} \times \mathbb{R}$, where $s_{c,t} = [Cop_{x,t}, Cop_{y,t}]^T$ and $s_{r,t} = [Rob_{x,t}, Rob_{y,t}]^T$. The robber again executes a Gaussian random walk,

$$s_{r,t+1} \sim \mathcal{N}(s_{r,t}, I).$$

The cop's noisy actions are $A = \{East, West, North, South, Stay\}$; each has an expected displacement of 1 m in the corresponding direction. The cop receives semantic observations $\Omega = \{East, West, North, South, Near\}$, simulating a coarse proximity sensor that depends on the relative location between the cop and robber; the softmax likelihood model for this is shown in Fig.

3.6. Rewards are based on the cop's distance from the robber,

$$r(dist(Rob_t, Cop_t) \leq 1) = 5,$$

$$r(dist(Rob_t, Cop_t) > 1) = 0.$$

As such, policies are found for the combined difference state $s_t = s_{r,t} - s_{c,t} = [\Delta X_t, \Delta Y_t]^T = [Rob_{x,t} - Cop_{x,t}, Rob_{y,t} - Cop_{y,t}]^T$, so that $N = 2$. The corresponding continuous state reward function is modeled as a GM consisting of a single weighted Gaussian located at the point $[0 - \Delta(a)_x, 0 - \Delta(a)_y]$, where $\Delta(a)$ is the expected displacement of the cop resulting from a given action. This incentivizes the cop to drive s_t to $[0, 0]$. Of note, no negative reward is introduced as a time penalty. With a single source of positive reward and no negative rewards, the actual weight of the reward GM mixands is irrelevant, as any reward gradient is enough to encourage the cop to maximize reward by reaching the desired state quickly.

Both GM-POMDP and VB-POMDP solvers were given 8 hours to find policies, though in both cases approximations had converged within 4 hours. In addition to comparing these approximations to a simple greedy policy as a reference baseline 'online' approximation as before, an omniscient 'Perfect Knowledge' solver (i.e. which has access to perfect observations about the robber's location) was also assessed to provide an upper bound on optimal policy performance. This solver leads to a policy whereby the Cop's actions minimize its distance from the mode of its belief in the robber's location, where the belief is modeled by a Dirac delta function centered on the robber's true state.

All policy approximation methods were again compared pair-wise using the Student's t-test for the difference of two means, with 1000 samples each. Fig. 3.7 shows the accumlated results, in which the VB-POMDP method significantly outperforms both the GM-POMDP and greedy approximations with $p < 0.05$, while it is outperformed by the Perfect Knowledge policy with $p < 0.05$.

To examine the policies' sensitivities to the problem parameters, the simulations were re-

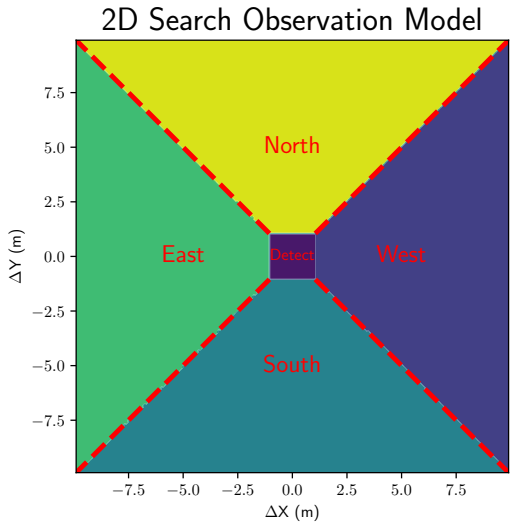


Figure 3.6: Softmax semantic observation model for 2D search problem, with $s = [\Delta X, \Delta Y] = [Rob_x - Cop_x, Rob_y - Cop_y]$.

peated with slower robber dynamics,

$$s_{r,t+1} \sim \mathcal{N}(s_{r,t}, 0.7 \cdot I)$$

The results of these tests are shown in Fig. 3.8. The relationship between Perfect Knowledge, VB-POMDP, and GM-POMDP remains unchanged. The Greedy policy now outpreforms GM-POMDP, yet still falls short of VB-POMDP, both with $p < 0.05$. These results, when combined with those of Fig. 3.7 indicate that problems with higher levels of noise or uncertainty derive greater benefit from more complex planning and control algorithms.

To examine the algorithms' responses to differing observation models, the tests were run again with a modified version of the 2D search observation model, shown in Fig. 3.9. This model contains two observations, the "Detect" observation which is identical to that from Fig. 3.6, and the "No Detect" observation which combines the 'North, South, East, and West' observations of Fig. 3.6. This leads to a multi-modal softmax (MMS) observation modal and highly non-Gaussian state beliefs with a generally lower certainty of the target's position. The measured statistic of these tests is the number of steps for the pursuer to catch the target for the first time. Each simulation was allowed 100 steps to reach this goal state before termination.

2D Search Results

Method	Mean Reward	Standard Deviation
Perfect Knowledge	118.4	±26.5
VB-POMDP	110.8	±25.8
GM-POMDP	101.2	±23.4
Greedy	81.2	±23.7

Table 3.3: Average final rewards for the 2D search problem (standard deviations over 1000 simulation runs).

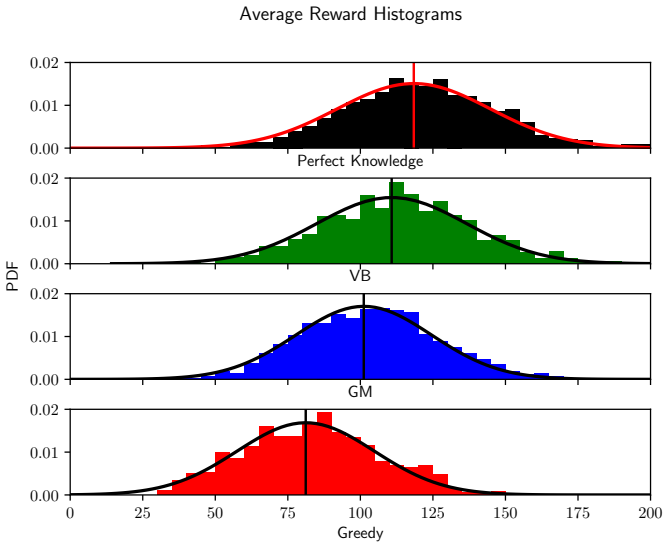


Figure 3.7: Total reward histograms for 2D search problem.

The results of these tests are shown in Table 3.4. A binomial statistical test was used to compare the percentage of captures for each method. From this metric, it is once again found that VB-POMDP outperforms both GM-POMDP and Greedy policies in pair-wise statistical comparisons with $p < 0.05$, while GM-POMDP outperforms Greedy with $p < 0.05$.

3.7.3 Discussion

The co-linear search scenario results indicate that for a simple problem VB-POMDP achieves near parity with GM-POMDP method, and that both methods surpass the greedy approach. This is expected, as both the GM and softmax observation models were constructed to approximate

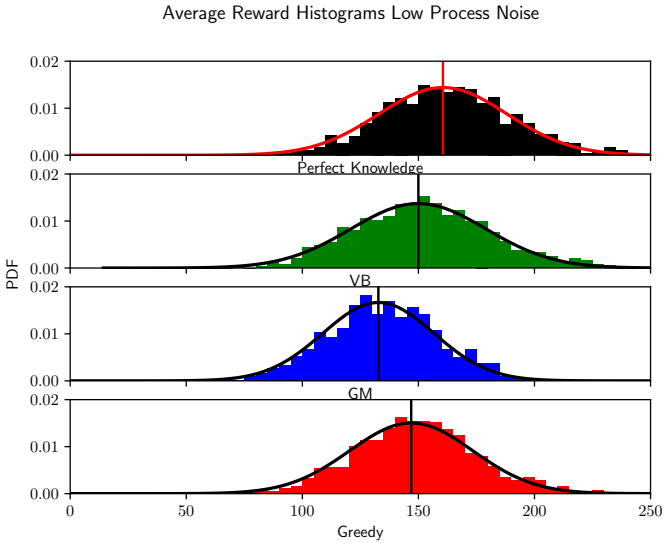


Figure 3.8: Total reward histograms with slower robber.

the same semantic model with all else held equal. Importantly, the VB approximations used by VB-POMDP do not seem to significantly impact the quality of the policy approximation.

Comparing the results from Sections 3.7.1 and 3.7.2 suggests that the VB-POMDP approximation outperforms GM-POMDP as problem complexity increases. A contributing factor to this disparity is that VB-POMDP can complete more backup steps within an allotted time than GM-POMDP for the 2D search problem (e.g. VB-POMDP completed 6 times more backups than GM-POMDP for this problem parameterization running on a 2.6 GHz processor running Linux with 16 GB RAM). This is largely due to the number of mixands generated by each method. In a single backup step, the GM-POMDP method produces alpha-functions of size $|\alpha_n| = \sum^{|\Omega|} |\alpha_{n-1}| M_o$ (where M_o is the number of unnormalized GM terms needed to define $p(o'|s')$ in eq. 3.11), whereas VB-POMDP produces alpha-functions of size $|\alpha_n| = \sum^{|\Omega|} |\alpha_{n-1}|$. The additional time needed for VB to converge is more than offset by the condensation time savings from having fewer mixands. An example of the results of these time savings is shown in Fig. 3.10. The VB-POMDP policy allows the Cop to act more strategically than the GM-POMDP policy when it loses contact, while avoiding the naive pursuit strategy of the Greedy method. This agrees with intuition: completing additional backups should allow the solver to better approximate the optimal policy.

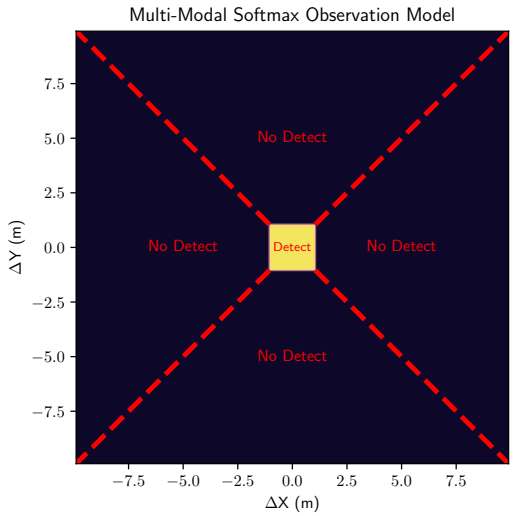


Figure 3.9: MMS semantic observation model for 2D search problem, with $s = [\Delta X, \Delta Y] = [Rob_x - Cop_x, Rob_y - Cop_y]$.

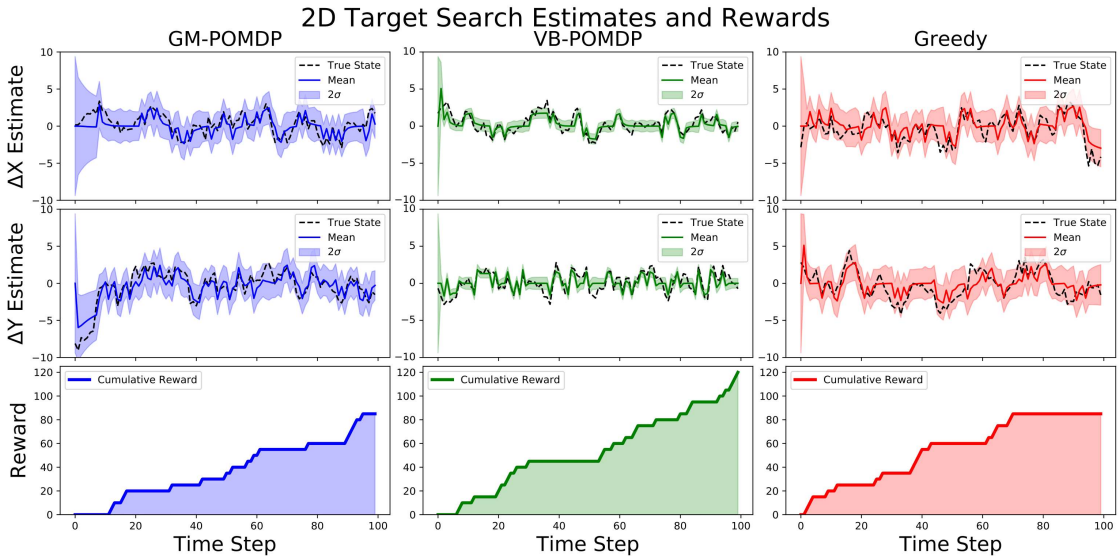


Figure 3.10: A comparison of robber state estimates and rewards for a typical 2D Target Search simulation using different policy approximations. VB-POMDP (green) maintains a slightly over-confident belief but avoids extended periods without reward, leading to a higher average reward than either GM-POMDP method (blue) or Greedy (red). All distances in meters, with each $\Delta T = 1$ sec time step representing a single discrete simulated dynamics and measurement update.

Another contributing factor to the quality of the VB-POMDP approximation is the amount of condensation required between backups. As shown in 3.5, condensation of larger mixtures leads to larger approximation errors, setting a practical limit on how closely a theoretically optimal policy

Multi-Modal 2D Search Results

Method	Percent Caught	Mean Steps to Catch	SD
VB-POMDP	65.3	48.4	±26.9
GM-POMDP	51.2	43.2	±27.3
Greedy	35.3	49.3	±29.4

Table 3.4: Capture statistics for MMS 2D Search

can be approximated with a finite number of GM components. Since VB-POMDP generates a smaller number of mixands during the backup step, such errors from condensation accumulate less often. This suggests VB-POMDP and GM-POMDP ought to produce the same results if each were given an infinite amount of time and an unbounded number of mixands for policy approximation. However, in practical resource limited situations for offline computation, VB-POMDP holds a distinct advantage.

Note that the VB approximation for state belief updates in eq. (3.32) can produce slightly over-confident posteriors, as seen in Fig. 3.10. As discussed in ref. [6], this can be mitigated by using a VB importance sampling (VBIS) approximate softmax update, which carries an additional Monte Carlo importance sampling step to compensate for optimistic covariances produced by the VB softmax update approximation. This would result in an additional speed vs. accuracy trade-off.

3.7.4 Comparison to Online Algorithms

Like GM-POMDP, VB-POMDP is an offline policy approximation algorithm, requiring the majority of computation to take place prior to deployment on a real platform. Alternative online policy approximation methods could also be used to solve CPOMDPs, but would result in different implementation tradeoffs for speed, online computing requirements, and belief space coverage. This section compares VB-POMDP to a state of the art online approach known as Partially Observable Monte Carlo Planning (POMCP) [89], which is a Monte Carlo Tree Search (MCTS) [32] based algorithm for online POMDP approximation. While this comparison of offline VB-POMDP to online POMCP is not strictly an 'apples to apples' comparison, it does provide some useful insight

to underscore the practicality of VB-POMDP (and offline solvers more generally) for problems like dynamic search and localization with semantic observations.

POMCP uses a generative model of state dynamics and observations to propagate a search tree of histories, choosing the path through the tree with the greatest expected reward. POMCP is of particular interest since it has successfully been applied to problems with large discrete state and observation spaces beyond what many non-sampling based offline algorithms can typically handle. Due to the fact that it only requires a 'black-box' generative model of the problem to function, POMCP has also been shown to function well in continuous state spaces [40]. POMCP also provides an online 'anytime algorithm', where computation can be cut short at some threshold and return the best answer found to that point. POMCP is considered here as a baseline state of the art 'general purpose' policy solver, though it can suffer from extremely suboptimal worst case behavior in certain kinds of problems with sparse rewards (as in search/localization), due to its reliance on the UCT algorithm [31].

The simulations here make use of the Julia POMCP implementation in the POMDPs.jl toolbox [36]. Simulations of the 2D Target Search Problem were run for 9 separate test cases using a 3.3 GHz processor, 32 GB of RAM, and a Julia language implementation on a system running Ubuntu 16.04. The cases were run with an exploration parameter of $c = 10$, and were allowed a planning depth of up to 100 time steps, mirroring the 100 steps allowed during the each run. The solver was allowed as many tree queries as could be completed within a given decision time. The cases differed only in the amount of time was allowed for an action to be chosen, varying from 0.05 secs up to 3 secs. These bounds were chosen to compare to the typical online decision time required for VB-POMDP or GM-POMDP running in a Python environment on the same machine, 0.05 secs, up to a maximum allowable wait time for a physical robot preforming a real-time task, 3 secs. Each case was run for 100 trials, with the mean final rewards shown in Fig. 3.11 (results were similar for more than 100 trials).

From Fig. 3.11, it is clear that increasing allowable decision time beyond 1 sec yields real but diminishing returns. Compared to offline GM-POMDP, POMCP reaches statistical similar-

ity in about 0.5 secs of decision time ($p > 0.05$). Compared to VB-POMDP, POMCP reaches statistical similarity around 3 secs of decision time ($p > 0.05$). While it is likely that POMCP would continue to show marginal improvements with additional decision time, it should be noted that these simulations were run with vastly more computing power than would be available to a typical small mobile robotic platform, and were run in isolation without siphoning off available processing for tasks such as control, vision, or communication. While Fig. 3.11 demonstrates that an online solver such as POMCP can achieve similar results to a full-width offline solver if given sufficient resources, the results imply that offline approximations like VB-POMDP can offer some implementation advantages for mobile robotic platforms.

3.7.5 LTI Dynamics Models Simulations

The 2D search problems considered thus far used kinematic Nearly Constant Position (NCP) random walk transition models for the robber, with $F = I$. The problem is extended so that the robber now uses a kinematic Nearly Constant Velocity (NCV) model, which is commonly used for target search and tracking. The NCV model requires 4 states to capture differences in cop/robber position and velocities. For a given action and an augmented state vector $s = [\Delta X, \Delta Y, V_{\Delta X}, V_{\Delta Y}]$, where $V_{\Delta X}$ and $V_{\Delta Y}$ are relative distance rates of change,

$$s_{t+1} = Fs_t + \Delta(a_t) = \begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} s_t + \Delta(a_t)$$

where ΔT is the physical time step. This state dynamics model requires the generalized Bellman backup equations for LTI dynamics introduced in Section 3.5.1. The softmax models used previously for semantic position observations are easily updated to accommodate the velocity states, namely by augmenting all softmax class weights and biases with additional rows of 0's for the new state dimensions.

LTI Simulation Results

	NCP Actual	NCV Actual
NCP Policy	110.0 ±29.3	97.5 ±26.2
NCV Policy	110.7 ±26.8	99.3 ±24.2

Table 3.5: Average Final Rewards for NCP and NCV Policies with Different Actual Target Models.

The target search simulations considered so far constrained the policy approximation's robber state dynamics model to be identical to the robber's actual dynamics. However, the true dynamics model will not always be exactly known in real applications. This constraint was therefore relaxed to examine VB-POMDP's sensitivity to model mismatches in this higher dimensional setting. Policies were approximated with VB-POMDP assuming either an NCP or NCV model; each policy type was then implemented in scenarios where the robber either actually used the NCP model or NCV model. Table 3.5 shows the results of 100 simulations for each scenario.

The VB-POMDP policy is able to adapt to which ever transition model is actually being used by the robber. While scenarios in which the policy and actual robber model matched performed slightly better on average, the mismatched model scenarios still achieved similar results. As seen in the example in Fig. 3.12, the consequences of model mismatch are more apparent when the simpler NCP model is assumed for more complex NCV actual robber dynamics. In the 'NCP Policy, NCV Actual' scenario, the approximate policy assumes zero mean robber velocity at all times. This repeatedly leads to incorrect beliefs, causing the policy to select suboptimal actions which lead to a slightly lower final reward. This issue is less pronounced in the 'NCV Policy, NCP Actual' simulations, where the belief converges to a correct estimate of zero mean velocity, and achieves similar similar rewards to the policy trained on NCP models. These results show that the approximations used by VB-POMDP perform well in a higher dimensional target search setting and still lead to reasonable behaviors even with slight model mismatches.

3.7.6 Multi-robot Localization/Goal-seeking Problem

The 'Cop and Robber' problems considered so far in $N = 2, 4$ continuous state dimensions still use a fairly limited set of actions and observations for a single decision-making agent. This subsection describes a considerably more challenging localization application to assess VB-POMDP's usefulness for approximating policies for CPOMDPs featuring higher dimensional continuous dynamical state spaces and more complex action/observation spaces. The problem consists of five independently controlled robotic agents who attempt to reach designated goals in a 2 dimensional plane. The combined localization and movement problem thus contains 10 continuous state dimensions. At each time step, only one robot is allowed an action. Each robot can take one of 4 actions to move (one at a time) in one of the 4 cardinal directions, resulting in 20 total possible actions. At each time step, a single robot is chosen to move and receive a semantic observation. One of the agents (Robot 1) can semantically observe its location with respect to a fixed landmark. Robots 2-5 can only observe their location with respect to the previously numbered agent (e.g. Robot 3 observes that it is North of Robot 2; Robot 4 observes it is East of Robot 3, etc.). After moving, each robot also receives a semantic observation indicating whether or not it has arrived at its goal (thus providing a source of negative information). All observations and beliefs are processed by a single centralized planner policy, which determines which robot to move at any given time. Fig. 3.13 shows the problem setup. Fig. 3.14 depicts an example initial 10-dimensional state belief for this problem, which is highly non-Gaussian and modeled with a GM pdf.

In addition to featuring non-Gaussian initial state beliefs, this problem is made more challenging by the random 'dropping' of observations (e.g. simulating the effect of a noisy communication channel between the robots and the centralized planner) and multi-modal/non-Gaussian state transition models. At any given time step, there is a uniform probability that one of the observations recorded by the robots will not be received by the planner. Movement actions are subject to multi-modal process noise, modeled by the GM process

$$p(s_{t+1}|s_t, a_t) = \frac{1}{3}\sum_{h=1}^{3} p_h(s_{t+1}|s_t, a_t), \tag{3.35}$$

where mixand $h = 1$ has a zero mean process term and mixands $h = 2, 3$ have non-zero mean process terms to model disturbances perpendicular to the intended direction of travel (e.g. due to wind, slippage, etc.). Using the softmax synthesis methods detailed in [5, 100], semantic observation models for each robot were separately constructed in two dimensions for Robot 1, and four dimensions for Robots 2-5, before being uniformly extended through the remaining 8/6 dimensions. This extension requires padding the weight vectors for each softmax class with 0's for each added dimension. Robot 1's observation model is identical to the one used earlier in the 2D search problem, but with absolute coordinates instead of relative ones. For Robots 2-5, the non-zero softmax weight terms are selected to embed a 4D probabilistic parallelepiped, such that for any given location of Robot i, the observation model for Robot $i + 1$ resembles Robot 1's absolute measurement model after a coordinate shift to a given location. These softmax models are also easily modified to derive the MMS models for each robot's goal-relative semantic observations (akin to Fig. 3.9).

The VB-POMDP policy approximation for this problem efficiently maneuvers each robot to its goal [4] . In contrast to a simple greedy policy which attempts to move each robot directly to its goal, the VB-POMDP policy pursues information gathering actions by moving robots across multiple semantic observation class boundaries in order to firmly localize positions (see motion traces in Fig. 3.13). This behavior is further illustrated in Fig. 3.15. The VB-POMDP policy also strategically positions well-localized robots so that downstream observers are better localized via relative observations. As expected, the VB-POMDP policy begins taking greedy 'go directly to goal' actions only after all robots are well-localized. VB-POMDP's ability to produce such sophisticated information gathering behaviors in high dimensional problems with complex uncertainties underscores its value as a scalable policy approximation for CPOMDPs with semantic observation models.

Note that while straightforward geometric reasoning can be used to easily design softmax observation models for this problem, synthesis of similar likelihood models via unnormalized GMs

[4] new policies must be solved for new sets of goals; the policy for results shown here required 36 hours to compute in the same computing environment used for the lower dimensional search problems

is especially cumbersome and impractical. At a minimum, thousands of mixands would be needed to define a GM likelihood that accurately models the boundaries of each semantic observation region in the 10-dimensional state space with sufficient resolution, while also obeying the 'sum to 1' constraint over all semantic class labels. Such enormous GM likelihoods would make offline Bellman backups for GM-POMDP extremely expensive to implement and Bayesian state pdf belief updates impractical for online execution (even on powerful modern computers), as extreme levels of condensation would be needed to maintain manageable GM pdf beliefs.

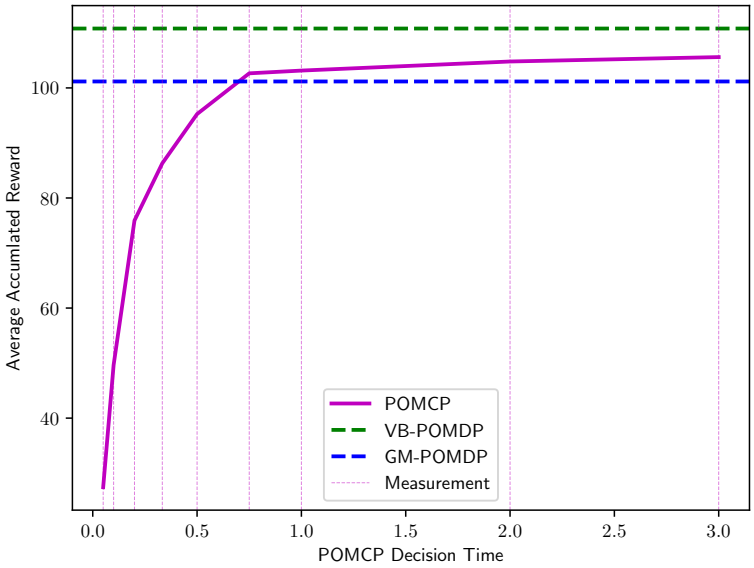


Figure 3.11: Average POMCP final rewards vs. planning time.

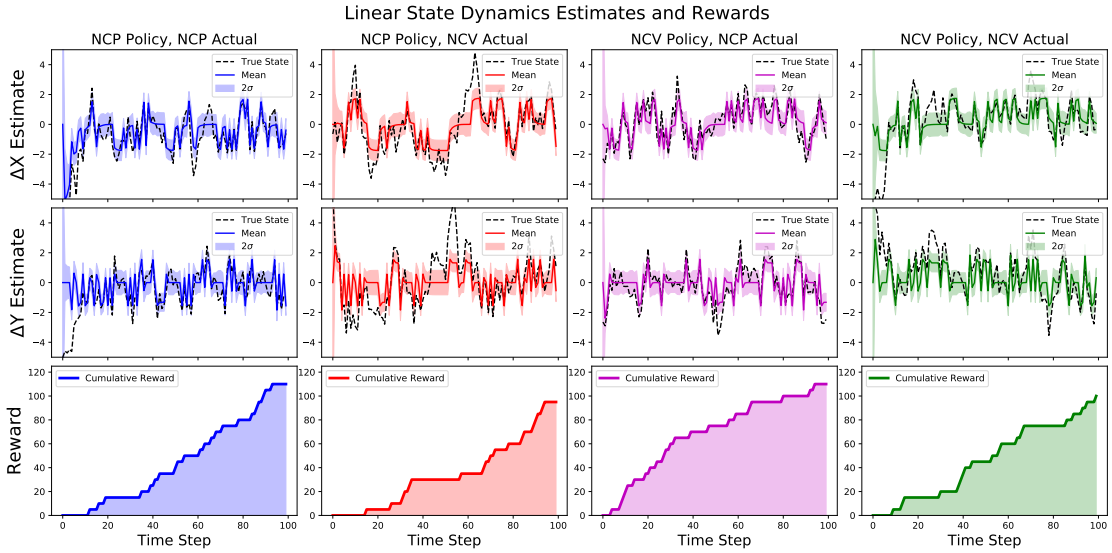


Figure 3.12: A comparison of state estimates and rewards for a typical run of the dynamic 2D target search problem under Nearly Constant Position (NCP) and Nearly Constant Velocity (NCV) models. All runs use VB-POMDP, which adapts in mismatched robber model cases to achieve nearly the same performance obtained by policies using the correct robber model. All distances in meters, with each time step $\Delta T = 1$ sec representing a single discrete simulated dynamics and measurement update.

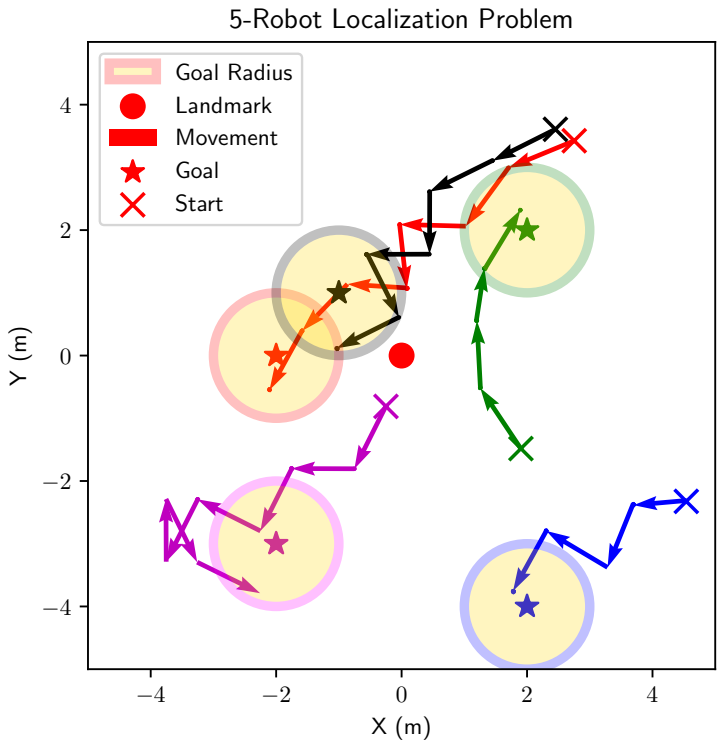


Figure 3.13: Set up and policy execution for 5-robot localization.

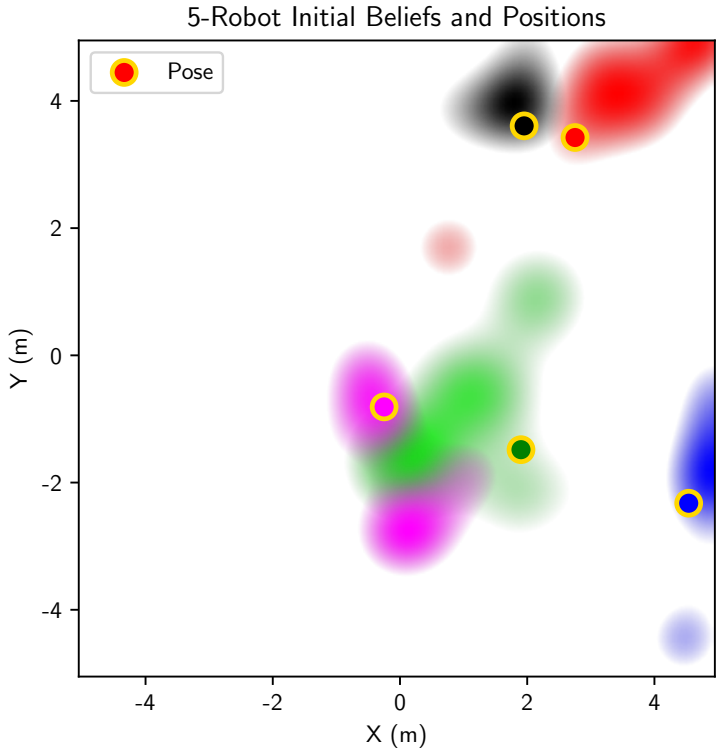


Figure 3.14: Initial marginal state pdfs (color-coded by robot).

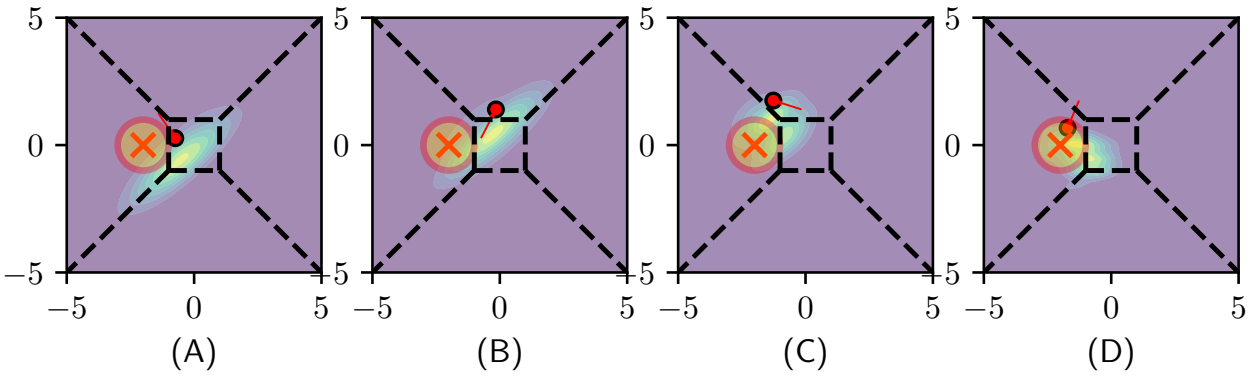


Figure 3.15: The VB-POMDP policy approximation for the 5-Robot problem displays non-myopic behavior to enable better robot localization before moving robots towards goals: (A), robot (red dot) has a broad uncertainty about its state, shown in the belief heatmap contour; (B) and (C): instead of attempting to move directly to the goal ('x', with 'reached' radius shown as circle), the robot moves (red solid trail) across the probabilistic class boundaries of its goal-relative semantic observation model (black dashes); (D): having localized itself sufficiently, the robot moves to its goal state.

Chapter 4

Collaborative Human-Autonomy Semantic Sensing through Structured POMDP Planning

Having established the VB-POMDP algorithm for semantic sensing, this chapter focuses on reintroducing the human as an active, query-able sensor for the robot to use. It also expands the problem domain to larger, more complex indoor environments and develops a framework allowing the preceding work to scale.

4.1 Cooperative Human-Robot Target Search

This chapter focuses on search and localization applications where the number and type of targets are known, and where mobile sensor platform dynamics, observation models, and environment maps are known. The motivating scenario is a dynamic target search problem, "Cops and Robbers" (CNR), which takes place in an indoor environment such as the one shown in Figure 4.1. An autonomous robotic agent, referred to here as the cop, is tracking a robber within the confines of an indoor environment, with the help of an off-site human collaborator. The cop's goal is to localize, track, and intercept the robber as quickly as possible. The environment is segmented into interconnected "rooms", each of which carries a known semantic label typical to a domestic home environment, such as "hallway" or "kitchen". Within each room, distinct objects are placed in fixed locations, each with its own semantic label. For instance, in the room "kitchen", one of the labeled objects might be the "refrigerator", while in the "study" there might be a "desk". In a real-world scenario, each room would likely contain a multitude of non-unique objects in many

possible configurations [94]. The robber moves randomly from room to room, independent from the movements of the cop and with no preference for a given room or sequence of rooms.

The human collaborator is able to imperfectly monitor the environment through the use of security cameras placed in the environment, and can also access the viewpoint of the cop through an on-board mounted camera. The human's role is to provide information relevant to the cop's task through two related methods, with the terminology for both adopted from previous work in [49]. The first is the "Robot-Pull" event, in which the cop requests information about the robber's position relative to a labeled object or room. For instance, the cop might ask "Is the robber in front of the chair?", to which the human can provide either a binary answer of "yes" or "no". The second method is through "Human-Push" events, in which the human can volunteer information which they deem useful. As an example, after checking the security camera mounted in the kitchen, the human might push the statement, "The robber is not in the kitchen". In both cases the human is assumed to be imperfect but well-intentioned, and is capable of passing along false information by mistake. Note that the human is unable to directly influence the cop's movements, and at no point are commands or directions issued. However, the semantic information they provide directly influences the cop's understanding of the targets position, and therefore will have some effect on actions. For instance, repeated assertions by the human that the robber is in the study would reasonably lead to the cop investigating the study. In most cases however, due to the view of security cameras not spanning the entire environment, the human will not be able to immediately locate the target and will be limited to negative observations.

4.1.1 Chapter Contributions and Summary

The first major contribution of this chapter is the specification of CPOMDPs for collaborative robotic information gathering and optimal planning where a human collaborator is formulated as a 'pointable' semantic sensor, which can be actively queried for information relevant to an autonomous agent's goal. The VB-POMDP [26] variant of the CPOMDP formulation is used to solve a target search problem with semantic sensor measurements. These measurements are described as softmax

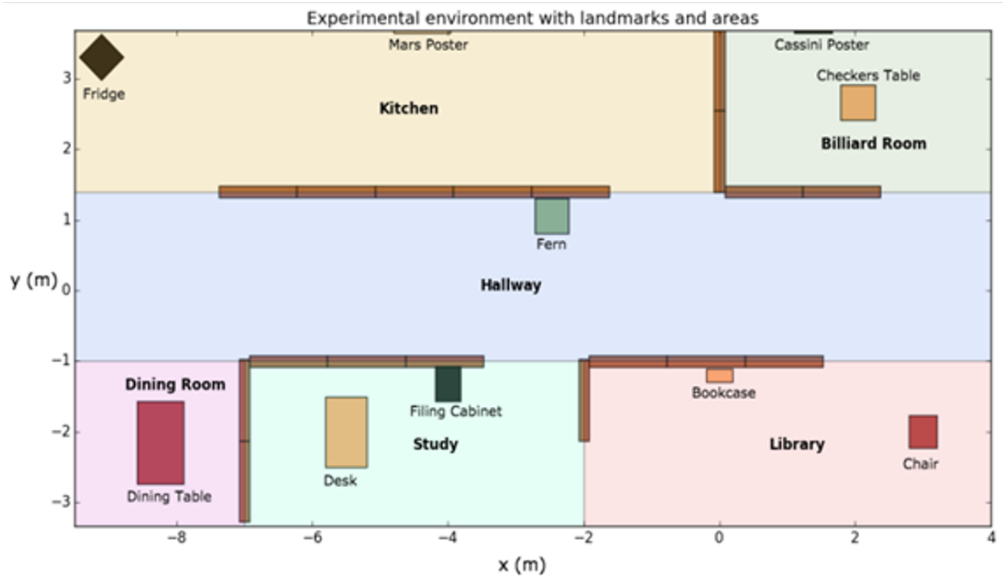


Figure 4.1: An example cops and robots room layout. The global space is fully partitioned into labeled rooms, each of which contains objects of known semantic labels and positions

functions as in [6], and are used to fuse information in both "Human-Push" and "Robot-Pull" events, as well as an on-board robotic measurements. This results in the computation of an optimal POMDP planning and querying policy, where the actions recommended by the policy consist of both physical movements and queries for the human, and observations consist of both robotic and human semantic information. This pre-computed policy is able to run in real-time on a physical robotic platform, and actively accounts for collaborative information gathering in a human-robot team.

The second major contribution of this chapter is a divide and conquer hierarchical POMDP-based querying strategy for large problem spaces. This approach exploits the natural segmentation of certain structured environments, such as a building, into multiple connected open subspaces, such as rooms. Each of these spaces can be treated as a separate CPOMDP, while a PBVI-based discrete POMDP solver can be used to connect policies for the lower-level CPOMDPs at a higher level. This results in a hierarchical POMDP structure, with the discrete solver directing which subspace level CPOMDP policy should be followed at each time step. Unlike previous hierarchical POMDP structures, which generally seek a single action at the "leaf" level of an action hierarchy, this approach combines a uniform set of action primitives from the lowest level POMDPs with a semantic labeling scheme applied to the higher level, resulting in composite actions unique to a particular decision path without requiring unqiue primitives for each subspace.

The final contribution of this chapter is a set of results from the implementation of such a framework to the grounding Cops and Robots problem. A discussion of these results, gathered in hardware using live humans, gives insight into the efficacy of the previous contributions, and explores robotic behavioral changes proceeding from the introduction of a human collaborator.

4.2 Formal Problem Definition

Formally, the generalized human-autonomy collaborative sensing and planning problem addressed in this chapter is stated as an infinite horizon POMDP, represented by a 7-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma\}$. States $s \in \mathcal{S}$ are assumed to be continuous in $\mathbb{R}^n$. The action space

$\mathcal{A}$ consists of a Cartesian product of the set of robotic actions $\mathcal{A}_m$, such as movements or other such decisions that purely affect the autonomy, and query actions $\mathcal{A}_q$, which denote a decision to request information from the human collaborator, such that

$$\mathcal{A} = \mathcal{A}_m \times \mathcal{A}_q, (a \in \mathcal{A}) = [a_m, a_q] \tag{4.1}$$

The probabilistic transition function $\mathcal{T}$, in this application, is specified as an n-dimensional Gaussian, with mean $s + \Delta a$, and variance Σ_a, where Δa is the n-dimensional vector of the expected resulting change occurring due to action a, and Σ_a is an $n \times n$ covariance matrix of transition noise resulting from action a. Executing action a results in a probabilistic transition from state $s \in \mathcal{S}$ to state $s' \in \mathcal{S}$.

$$\mathcal{T} = p(s'|s, a) = \phi(s'|s + \Delta a, \Sigma_a) \tag{4.2}$$

While this Chapter only considers transition functions of this unimodal Gaussian structure, they could also be constructed from Gaussian Mixtures as shown in Section 3.7.6 of this thesis, in order to capture multi-modal uncertainties. This would require no significant algorithmic or theoretical changes to the methods described below. Furthermore, non-Gaussian models could in theory be applied in so much as their product with either softmax functions or Gaussian mixtures can be approximated as a Gaussian Mixture after the manner of the VB-POMDP algorithm described in Chapter 3.

The observation space $\mathcal{O}$ is decomposed similarly to the action space in that it consists of a Cartesian product of observations resulting from the robot's onboard sensors $o_v \in \mathcal{O}_v$ and the human's responses to the robot's query actions $o_q \in \mathcal{O}_q$, such that:

$$\mathcal{O} = \mathcal{O}_v \times \mathcal{O}_q, (o \in \mathcal{O}) = [o_v, o_q] \tag{4.3}$$

The observation model Ω relates the probability of receiving a particular observation o to the state s through the use of an n-dimensional softmax function as per Chapter 3 and [6, 100].

$$\Omega = p(o|s, a) = \frac{e^{w_o^T s + b_o}}{\sum_{j=1}^{|\mathcal{O}|} e^{w_j^T s + b_j}} \tag{4.4}$$

The probability distribution over the current state s at time t, referred to in this work as the belief b, is represented as a Gaussian Mixture pdf consisting of M weighted mixands,

$$b(s) = p(s_t|a_{0:t}, o_{0:t}) = \sum_{m}^{M} w_m \phi(s|\mu_m, \Sigma_m) \tag{4.5}$$

$$1 = \sum_{m}^{M} w_m \tag{4.6}$$

4.3 General POMDP Solution

The general solution to the POMDP is a policy $\pi(b)$, which maps from a belief to an action $\pi(b) \rightarrow a$, such that the expected discounted reward function $\mathcal{R}(s, a)$ over time is maximized. Thus the value function V under a particular policy π and starting belief b_0 is

$$V^{\pi}(b_0) = \sum_{t=0}^{\infty} \gamma^t \mathbf{E}[\mathcal{R}(s_t, a_t)|b_0, \pi] \tag{4.7}$$

for a given time discount factor γ.

As shown in [93], the policy $\pi(b)$ can be represented as a set of piece-wise continuous functions, each of which correspond to a non-exclusive action. These piece-wise functions, when in a continuous state space as in [72], are known as α-functions, and are constructed as Gaussian Mixture models over $\mathcal{S}$. A given action may correspond to multiple α-functions, in a one-to-many fashion, while each function can be associated with only one action. These α-functions are collected in the set Γ, which is used is used to find the policy $\pi(b)$ for a given belief as the action attached to the α-function maximizing its continuous dot product with the belief,

$$\pi(b) = argmax_{\alpha \in \Gamma} < \alpha(s), b(s) > \tag{4.8}$$

Such a policy is obtained in the manner of the VB-POMDP algorithm [26], using dynamic programming to solve the Bellman backup equations with a Variational Bayesian approximation to fuse the softmax observation functions with Gaussian Mixture α-functions.

4.3.1 Application to Dynamic Target Search and Localization

The grounding CNR problem is now formulated in the CPOMDP setting. As stated above in the general formulation, it is an infinite horizon POMDP 7-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma\}$. The autonomous agent referred to as the "cop" has state $s_c \in S_c$ while the target "robber" has state $s_r \in S_r$ such that $S = S_c \times S_r \in \mathbb{R}^4$. The cop may initiate movement actions a_m and human query actions a_q such that the full discrete action space is $A = a_m \times a_q$. Movement actions a_m dictate changes from the cop's current state s_c to its next s'_c, modeled alongside the robber's motion through a conditional Gaussian transition model.

Query action a_q have no effect on the state, therefore the transition model is held independent of them.

The cop is rewarded only for being co-located with the robber, such that a robber in the same position as a cop is held to be captured. This reward is expressed in the reward model $\mathcal{R}$, such that for some distance threshold τ,

$$\mathcal{R} = \begin{cases} 100 & dist(s_c, s_r) \leq \tau \\ -1 & dist(s_c, s_r) > \tau \end{cases} \tag{4.9}$$

Note that the reward function in this case is not explicitly dependent on actions, but rather implicitly through the state transitions actions cause.

The cop receives observations from two sources, first being those from its on-board camera view o_v, which can take values $o_v \in \mathcal{O}_v = \{Detect, No\ Detect\}$. The viewable area of this cone is assumed to have angle θ projected to a forward flat leading edge of length L_v in the direction of movement. The results of human query actions a_q are modeled as additional action dependent observations $o_q \in \mathcal{O}_q = \{Yes, No\}$, such that $\mathcal{O} = \mathcal{O}_v \times \mathcal{O}_q$ with a set size $|O| = 4$. All observations are generated from corresponding softmax observation models Ω for some known fixed dictionary, and the time discount $\gamma \in [0, 1]$ is here set to 0.9. While a $\gamma = 1$ would correspond to a truly infinite horizon policy, most practical problems require some temporal discounting for policy convergence [47].

4.3.2 Single Policy Implementation

The implementation of the policy described above results in a single-policy, or monolithic, POMDP, wherein a single belief $b(s)$ is maintained which spans the state space $\mathcal{S}$ and a single policy maps said belief to an action. Obtaining such a policy for an active human sensing solution to a robotic search problem follows the four primary steps below, and is summarized in Algorithm 4:

(1) Identify and label salient semantic features in the state space. These could be objects, such as chairs and buildings, spatial areas such as rooms or neighborhoods, or categorical identifiers such as high, medium, and low on a continuous scale. Note that semantic features need not exist exclusively in Cartesian space. However, their ability to be interpreted by a human collaborator should be considered.

(2) For each semantic feature, identify and collect a set of semantic relational indicators with respect to the feature. These indicators can include global bearings such as North and East, local bearings such as "In front of" and "Behind", or binary existence indicators such as "Inside" and "Outside", as well as any other types of relation relevant to the problem at hand. Note, while not mathematically required, relational indicators for human collaboration problems should generally be intuitive or understandable to the human, as queries from the autonomous system will be drawn from this set. Including "Inside" as an indicator for the semantic feature "Chair" may not make sense in most problems, even if chair takes up a defined spatial area which a point could technically be inside of. The total collection of semantic labels and relational indicators make up the semantic dictionary for the problem.

(3) Using the softmax synthesis methods drawn from previous work [100], geometrically construct softmax functions around the spatial extent of each feature, labeling the resulting classes from the semantic dictionary. This could be in a one-to-one fashion, or a multi-

modal softmax representation in a one-to-many approach where each label combines multiple classes. The query action set A_q should consist of each member of the semantic dictionary, while the query observation function Ω should now contains the labeled softmax class weights and biases.

(4) Having combined A_q with a problem appropriate movement set A_m as in the previous section, apply the VB-POMDP algorithm from Algorithm 2 to the 7-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma\}$ with problem dependent choices for $\mathcal{T}, \mathcal{R}$, to obtain the policy π.

Algorithm 4 Active Human Collaboration POMDP Construction

1: **Input:** States $\mathcal{S}$, Transitions $\mathcal{T}$, Rewards $\mathcal{R}$, Discount γ
2: Hand select semantic feature set $\{f\}$
3: Assign semantic label set l_f to each feature
4: $A_q = \sum_f \{l\}_f$
5: $\Omega = p(o|s, a) =$ Softmax Synthesis$(f), \forall a \in A_q$
6: $\pi =$ VB-POMDP$(\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma\})$ #Algorithm 2
7: **return** π

Actions chosen during policy execution $\pi(b)$ will now be drawn from the set $A = A_m \times A_q$, which can then be disambiguated and used appropriately. The queries A_q and their resulting human observations have the effect of actively pointing a human sensor with respect to the chosen semantic dictionary, while the movement actions A_m effect the state and potential rewards more directly. It is imperative to note that while in this thesis actions A_m are referred to as movements, that does not constrain the frameworks discussed here to only physically embodied mobile robots. Rather, movement references a change in abstract state affected from an autonomous agent. Indeed, even the "human" part of the human query actions A_q is not strictly necessary. Any agent, physical or virtual, need only be capable of influencing the state through decisions and querying an external information source for probabilistically modeled semantic observations in order to implement this framework.

4.4 Hierarchical Continuous POMDPs

While the methods detailed in the previous section fully describe the problem at hand, problems with solving such a monolithic (single policy) POMDP arise in terms of scalability and modeling. This section puts forth hierarchical modifications to this formulation which allow effective approximations of optimal policies using humans as actively queryable sensors, as well as novel innovations for accommodating the human collaborator.

4.4.1 Formal Hierarchical Solution

In this section a general hierarchical framework is proposed for human-autonomy sensing and planning problems. The continuous state space $\mathcal{S}$ is fully partitioned into a set of non-overlapping lower level state subspaces $\mathcal{S}_l$, such that for each label (l), $\mathcal{S}_l$ is a proper subset of $\mathcal{S}$ which shares exactly zero states with other subspaces.

$$\forall l, \mathcal{S}_l \subset \mathcal{S} \tag{4.10}$$

$$\mathcal{S} = \cup_l \mathcal{S}_l \tag{4.11}$$

$$\cap_l \mathcal{S}_l = \emptyset \tag{4.12}$$

This partitioning occurs along discontinuous transition boundaries in $\mathcal{S}$, where the Gaussian transition model outlined in Section III.A fails to hold. Partitions need not be of equal size, nor are they necessarily restricted to any uniform regularity conditions such as shape or dimensionality. Each subspace S_l is then treated as a separate CPOMDP, and a policy π_l is found as in Section 4.3, with actions A_l and observations O_l. In general, partitions are predetermined by hand, and transitions between partitions should be representative of state space transitions in the unpartitioned space. For example, a neighborhood might "naturally" decompose into blocks, while transitions between blocks occur through a 4-connected or 8 connected grid. Alternatively, an indoor environment might be partitioned as a set of rooms, where the locations of doorways govern the transition between rooms.

A meta-space S_h is defined as the set of labels (l), and consists of a state space for a higher level discrete POMDP. The action space A_h of this POMDP corresponds to a Cartesian product of movements between partitions $A_{m,h}$ and human queries regarding each partition $A_{q,h}$,

$$A_h = A_{m,h} \times A_{q,h} \tag{4.13}$$

Similarly, observations O_h are given with respect to each partition as $O_{v,h}$ and queries regarding partitions $O_{q,h}$,

$$O_h = O_{v,h} \times O_{q,h} \tag{4.14}$$

The Gaussian Mixture (GM) belief $b(s)$ over the non-partitioned state space $\mathcal{S}$ is broken up into a set of conditional beliefs $b_l(s) = p(s|l)$. This is accomplished with a hard classification of each mixand to the partition containing its mean, creating the set $\{\omega_l\}$. Each mixand's mean in $\{\omega_l\}$ is constrained to stay within its assigned partition under dynamics updates, and any probability density originating from mixands outside the partition is ignored. ver if covariance of a mixands gets big, correct? Do mixands get to 'migrate' from one state region to another if updates or dynamics move them around, or do they get truncated to always stay inside a given region, or something else? The belief for the discrete meta-space S_h then becomes the sum of the weights for each element of $\{\omega_l\}$,

$$b(l \in S_h) = \sum w_m \mathbb{1}(w_m \in \{\omega_l\}) \tag{4.15}$$

$$\sum_l b(l \in S_h) = 1 \tag{4.16}$$

As the original belief $b(s)$ over $\mathcal{S}$ is required to be a proper pdf, this ensures that $b(l)$ is a proper pmf. The conditional beliefs for each partition can then be weighted by their respective probability $b(l)$, and the belief $b(s)$ over the full state space can be extracted from a sum over subspaces,

$$b_l(s) = p(s|l) = \frac{1}{b(l)} \sum_{m}^{M} w_m \phi(s|\mu_m, \Sigma_m) \mathbb{1}(w_m \in \{w_l\}) \tag{4.17}$$

$$b(s) = \sum_{l} b(l) b_l(s) = \sum_{l} p(l) p(s|l) \tag{4.18}$$

Of note, this leads to the slightly paradoxical assumption that a mixand mean transitioning towards a partitions boundary which would otherwise be traversable, such a doorway between rooms, will remain bound in its original partition. Rather than govern the flow of probability density between rooms in such a manner at the partition level, such flow is mediated at the meta state space level S_h, with probability mass being allocated among discrete states by a uniform change to each partitions contained mixand weights.

The observation models in the lower level CPOMDPs O_l are specified as softmax functions. These can be fused directly into the full space belief $b(s)$ by way of the Variational Bayes algorithm described in [6].

$$p(s|O_l) \approx \frac{p(s)p(O_l|s)}{p(O_l)} \tag{4.19}$$

Observations in the higher level discrete POMDP can be fused directly into $b(l)$ using a discrete Bayes Filter.

In terms of policy execution, action output $a_{m,h}$ from the discrete POMDP indicates which CPOMDP policy π_l to query. The action generated by $\pi_l(b_l(s))$ is then taken. Human queries are then generated from both $a_{q,h}$ and $a_{q,l}$. Responses to these queries, as well as robotic observations $o_{v,h}$ and $o_{v,l}$ are then fused back into the belief before requesting another action. This approach results in a set of CPOMDP policies, governed by a single discrete POMDP. Each policy can be solved independently and combined during runtime.

4.4.2 Application to Dynamic Target Search and Localization

Here the general hierachical POMDP formulation from the previous section is applied to the motivating CNR problem for target search in complex indoor search spaces, e.g. see Fig. 4.1. A

separate CPOMDP policy is found for each distinct room in a particular map, where obstacles are sparse enough not to necessitate the switching modes used in [24]. Each of these room level policies is then treated as an action selection by a discrete POMDP policy over the rooms. This leads to a novel hierarchical CPOMDP policy that can not only take fuse low-level semantic soft information about target locations in metric physical space (e.g. 'next to the chair'; 'not by the refrigerator'), but also exploit higher-level semantic data about target locations in abstract label spaces, i.e. room designations ('in the kitchen'; 'not in the dining room'). By accounting for the dependencies between these different types of high-level and low-level semantic data, we arrive at an intelligent hierarchical decision making policy that enables top-down motion planning (i.e. determine which areas to search, and then how to search them), as well as determination of the best set of high-level and low-level semantic queries for a human sensor that will ensure rapid capture of the robber. Transitions between rooms in S_h are governed by a discrete transition model, as shown in Figure 4.4.

4.4.3 Lower Level CPOMDP

The lower level CPOMDP for each room is specified in the same manner as the formal problem statement in the previous section, with state $S_l \in \mathbb{R}^4$ for room (l). The cop's state variables are changed deterministically with actions while the robber's are assumed to be drawn from a high variance Gaussian random walk,

$$p(s'_r) = \phi(s'_r | s_r, \Sigma_r) \tag{4.20}$$

The vector Δa is necessarily zero for the robber states s_r, as the cop's actions do not directly effect the robber's movements.

This Nearly Constant Position (NCP) model introduces an approximation when dealing with most real systems. While a drifting or unpowered target in a dynamic environment might adhere well to a Brownian motion scheme, intentional agents rarely do. Instead they tend to operate according to trajectories or patterns. The CPOMDP framework shown in [72] is equipped to

handle transition functions which can be modeled as conditional Gaussian distributions with their mean shifted by the actions Δa expected effect on the state s:

$$p(s'|s,a) = \phi(s'|s + \Delta a, \Sigma_a) \tag{4.21}$$

When only a limited number of state components are directly controllable, the others are forced to execute a Gaussian random walk with the given variance. This prevents the use of target trajectories in search problems such as the one described here. In order to incorporate target dynamics, it is desirable to have a transition function of the form,

$$p(s'|s,a) = \phi(s'|Fs + \Delta a, \Sigma_a) \tag{4.22}$$

where F is the state transition matrix which encapsulates changes in the state independent of actions. Bellman backups can be easily resolved with this alteration in the CPOMDP framework, thus permitting solutions to a broader class of continuous space planning problems, as shown in [26]. As the focus of this work is the hierarchical CPOMDP structure, and in an attempt to minimize the dimensionality, velocities are not included here.

The cop can choose from among 5 noisy movements $A_m = \{East, West, North, South, Stay\}$, and can ask questions about the robber's spatial relation to each object in the room such that $A_q = \{Objects\} \times \{Left, Right, Front, Behind\}$. Of note, while the question space could have been based in global compass coordinates, they are instead represented in local body coordinates. This is due to the fact that the cameras through which the human views the space, both the cop's on-board view and security cameras, contain no explicit global coordinate reference. While the provided map in the interface could allow the human operator to reason about global coordinates by knowing the camera placements, cognitive load is decreased by instead assigning orientations to each object and having the human refer to them each in their local frame. For instance, the chair object is set up with a 90 degree counterclockwise rotation, so the observation "Front" with respect to the chair indicates the area to its west. The full discrete action space is then $A = A_m \times A_q$. An example action might be "Move East and ask 'Is the robber in front of the fern?'".

In addition to human information resulting from its queries, the cop receives viewcone observations through its onboard camera, which is capable of visually detecting the robber at close range. Given that the cop's viewcone is fixed to the direction of travel along one of four directions, and has a leading edge of length L_v, the area of the box will correspond exactly to the area swept out by the leading edge of the viewcone during the preceeding action. This can be shown by integrating the length of the leading edge of the view cone in the direction of travel. So for any movement, the magnitude of the approximated area A_a swept out is:

$$|A_a| = \int_{s_c}^{s'_c} L_v ds_c = \int_0^{\Delta s_c} L_v ds_c = L_v \Delta s_c \tag{4.23}$$

Therefore any movement of m meters ($\Delta s_c = m$) behind a leading edge L_v meters long will produce an area of approximately $L_v \Delta s_c$ square meters which would have triggered the viewcone observation o_v during the time in which the movement was executed. From Eq. 4.23, it is clear that for different cop displacement or viewcone parameters the box approximation can be modified appropriately. The box model remains an approximation however, due to the fact that most of the area swept out by the viewcone will be in front of the cop, whereas we assume the cop to be centered in that area. Furthermore, the box approximation does not perfectly account for area swept out without contact with the leading edge of the viewcone, and sacrifices distant coverage in the direction of movement in favor of coverage near to the cop, as shown in Fig. 4.2. Thus while the area of the box represents a shifted area of similar size to that swept out by the viewcone, its size serves as a lower bound for that reached by the entirety of the viewcone. The approximation improves as the viewcone angle θ is increased while holding L_v constant, such that the area of the viewcone A_v becomes an infinitesimal area along L_v. Thus the approximation area magnitude $|A_a|$, and true

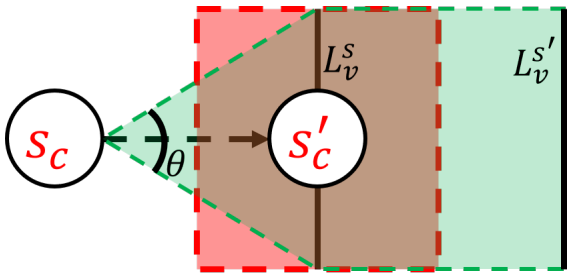


Figure 4.2: A comparison of the area swept out by the cop's viewcone in one timestep (green) vs. the area of the box approximation (red) for a viewcone with angle θ when moving from position s_c to s_c'

area coverage magnitude $|A_t|$ approach each other in the limit of the angle as:

$$|A_v| = \frac{1}{2} L_v \tan\left(\frac{180-\theta}{2}\right) \tag{4.24}$$

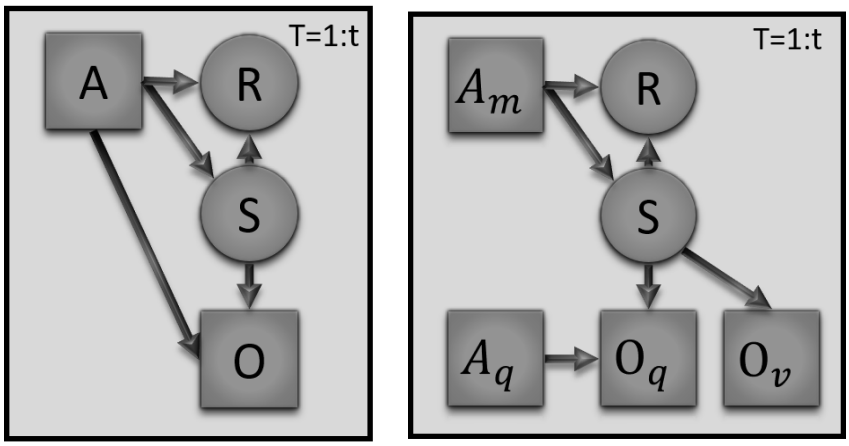
$$|A_t| = |A_v| + \int_{s_c}^{s_c'} L_v ds_c \tag{4.25}$$

$$\lim_{\theta \to 180} |A_v| = \lim_{\theta \to 180} \frac{1}{2} L_v \tan\left(\frac{180-\theta}{2}\right) = 0 \tag{4.26}$$

$$\lim_{\theta \to 180} |A_t| = |A_a| = L_v \Delta s_c \tag{4.27}$$

This limit, corresponding to a coverage area of a thin line immediately leading the robot, results in an approximation area which leads the true area, rather than the opposite which occurs with our hardware. Including orientation into the state vector would remove the need for the box approximation altogether, but further increase the dimensionality of the state. Thus to reduce the size of the state space for this implementation, the box approximation was used. Alternatively, the viewcone observations could carry a dependence on the movement action taken, transforming the likelihood model $p(o_v|s)$ to $p(o_v|s, a_m)$ where a_m acts as a switch between a number of likelihood models corresponding to the correct orientation for that action. While this modification increases the complexity of implementation somewhat, it is valid within the CPOMDP framework and will be explored in future work.

It is important to note that the action and observation spaces in the problem could be implemented without recognition of their ability to factor into distinct sets $\{A_m, A_q\}$ and $\{O_v, O_q\}$. However, this increases the difficulty of implementation when trying to account for the diverse



(a) Standard POMDP (b) Factored POMDP

Figure 4.3: POMDP graphical models with different action/observation factorizations.

results of the combined action/observation. In this case, the cop's movement actions A_m primarily effect the state without changing the observations, while the cop's 'question actions' A_q have no effect on the state at the current time, and fully dictate the meaning of the observations O_q. Similarly, the viewcone observations O_v are only state dependent and thus independent of either action, while O_q depends on both state and action. Factoring each space into it's constituent parts allows for simpler handling of these dependencies, and increases the explainability of the cop's actions and the changes in it's beliefs. The differences in the two approaches are summed up in Figure 4.3. Each room's lower level continuous POMDP policy is found using the VB-POMDP algorithm detailed in [26].

4.4.4 Higher Level Discrete POMDP

The higher level discrete POMDP is specified on a state vector consisting of all rooms. For example, the room configuration shown in Figure 4.1 corresponds to the state space:

$$S_h = \{\textit{Billiard Room, Hallway, Kitchen,} \\ \textit{Dining Room, Study, Library}\} \tag{4.28}$$

as shown in Figure 4.4. The state s_r represents the robber's current position, and the robber randomly transitions according to the particular connections between rooms in the map being used with probability $p(l'|l)$ independent of actions. The location of the cop is left out of the high level POMDP, instead being accounted for by a combination of the high level actions and low level state. The cop can choose movement actions $a_{m,h}$ corresponding to each room, which will deterministically move the cop to that room according to

$$p(s'_c = l|a_{m,h} = l) = 1 \tag{4.29}$$
$$p(s'_c \neq l|a_{m,h} = l) = 0$$

as well as questions actions $a_{q,h}$, which will ask the human if the robber is in a particular room. Thus for the higher level policy, both $A_{m,h} = S_h$ and $A_{q,h} = S_h$. If the $a_{m,h}$ indicates the room the cop currently occupies, the movement action of the lower POMDP policy is respected within that room as per Algorithm 5. In general, as a product of the PBVI [69] roots of the CPOMDP family of algorithms, similar beliefs will lead to similar actions. Therefore in practice movement between rooms tends to only occur after either a thorough search of a room or significant shift in belief, and the cop avoids bouncing back and forth between rooms without searching either. This behavior is confirmed in the experimental results below. As with lower level policies, the full action space is $A_l = A_{m,l} \times A_{q,l}$. An example action would be "Search the Library and ask 'Is the robber in the Kitchen?'".

In the higher level discrete POMDP, the cop is rewarded for choosing to move to the room the containing the robber, and penalized for choosing the wrong room according to

$$\mathcal{R}(s_h, a_{m,h}) = \begin{cases} 100 & a_{m,h} = s_h \\ -1 & a_{m,h} \neq s_h > \tau \end{cases} \tag{4.30}$$

The cop receives viewcone observations $O_{v,h}$ at each time step, similar to the lower level CPOMDP. Given that being in the same room as the robber does not guarantee a viewcone detection, likelihoods for $O_{v,h} = Detect$ are fairly low for any given time step even if the cop and robber are in the

Algorithm 5 Hierarchical CPOMDP Evaluation

Input: Discrete Policy π_h, Continuous Policy Set $\{\pi_l\}$
With Belief $b(s)$ and Cop State s_c
while $o_v \neq Detect$ **do**
  $l = a_m^h = \pi_h(b)$
  $a_m^l = \pi_l(b)$
  if $s_c \in l$ **then**
    $s_c' \sim p(s_c'|s_c, a_m^l)$
  else
    $s_c' \rightarrow l$
  end if
  $a_q = N\_Actions()$ #Algorithm 2
  $o_q =$ Human Responses
  $b = Belief\_Update(b, a_m^l, a_m^h, o_v, o_q)$
end while

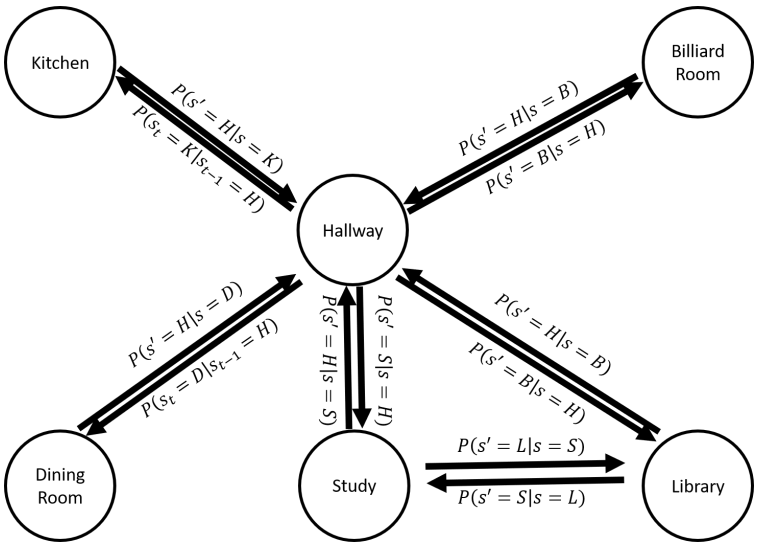


Figure 4.4: The states and transitions for the higher level discrete POMDP corresponding to the room layout in Fig. 4.1

same room. With respect to $O_{v,h}$ for the high level POMDP, rooms are treated as identical, such that each has an identical detection rate ν

$$p(o_{v,h} = Detect|a_{m,h} = l, s_h = l) = \nu, \forall l \tag{4.31}$$

This approach could be refined by considering the ratio of the area covered by the viewcone and the total area of the room, such that each rooms detection likelihood reflects its size, or by leveraging prior knowledge about likely robber movements and positions within each room.

It is important to note that during implementation of a policy, the observations $o_{v,h}$ and $o_{v,l}$ are only derived from the actual visual system, and are not double counted as separate observations from the high and low level systems. Viewcone observations are only modeled as above when finding a high level policy, to approximate the response of the low level viewcone. Also, as the policy is solved over the state space S_h, each additional spatial area considered adds only 1 additional state. This additional lower level policy does not contribute any complexity to the other room policies, allowing the hierarchical structure to scale well to larger numbers of spatial areas.

We use the Point-Based Value Iteration (PBVI) algorithm from [69] to find the policy π_h for the discrete layer.

4.4.5 Hierarchy and Question Lists

At each time step, the policy chooses an action consisting of a room to search and a room to query. If the cop is outside the search room, it is directed to go there. Otherwise, if the cop is already in the search room, the lower level CPOMDP policy is queried to provide a movement action. The query room is asked about in the form "Is the robber in (room)?", and the low level CPOMDP policy for that room gives an additional question about the robber's relation to an object in that room.

In this implementation the human sensor receives a question from the cop at every time step. Because the policy was trained to expect responses from the human sensor, steps where the human fails to answer are unknown events from the system's standpoint, i.e. they are not accounted for when solving for the policy. One method for handling these failures would be to include a "Null" observation with a uniform likelihood across states to represent a lack of human observation. Further steps could be taken by incorporating a form of human attention model into the state vector and an option to ask a "Null" question when the policy believes the human would not be able to answer. Each of these methods increases the problems complexity, either by enlarging the observation space for the first or by enlarging the state, action, and observation spaces for the second. Also, it should be noted that the policies for both the Higher and Lower level POMDPs map from **any** belief to an action. Both PBVI and VB-POMDP solve policies from a set of example beliefs that the system may encounter, and interpolate between them when an unseen belief is encountered during runtime. A belief resulting from an unmodeled human failure would in most cases not have been explicitly explored during policy solution. Yet it is likely sufficiently close to a belief that was such that a suitable action can be found. Therefore any effects of this discrepancy can be minimized, if not entirely negated, by solving over a sufficient set of example beliefs.

In most applications, the desired output of a POMDP policy is the action with the highest value for the current belief. This makes sense in most contexts as only one action can be taken at

a time. However, in our problem multiple questions could be displayed to the human at each time step, and so we want to ask the N most valuable questions. In both discrete and continuous policies using PBVI-type approximations each policy element, or α-element, contained in Γ corresponds to an action and encodes part of the approximate value function over beliefs. As each α-element is specified over the entire belief space, it can provide a value for its action at any belief, even were it does not provide the maximum value. Therefore, the α-elements with the top N values can be said to correspond to the top N actions, as shown in Figure 4.5. As multiple α-elements might correspond to the same action, this does not guarantee N unique actions. However, as all α-elements must be evaluated to choose the correct action for a belief, the top N unique actions can still be chosen. This also implies that choosing a list of actions requires only the minimal extra computation of a sorting function substituted for an argmax, as in Algorithm 6. This method of choosing the top N actions is ultimately a heuristic, equivalent to asking "Given the policy I have right now, if I ignored the existence of the top $N-1$ actions, what would the best action be". This disregards the fact that most PBVI-type algorithms prune away a large portion of α-elements which do not maximize the value of any particular belief, and are not actively collecting second-best or third-best actions. Therefore the only actions identified by this heuristic will be ones associated with α-elements which maximize the value of a different belief. While at the core of the PBVI approach rests the assumption that similar beliefs will generally have similar actions, the different beliefs maximized by a second or third best *alpha*-element are not guaranteed to be similar to the current belief.

Algorithm 6 Choose Top N Actions

1: **Function:** *N_Actions*
2: **Input:** Policy Γ, Belief $b(s)$, N
3: **for** $\forall \alpha \in \Gamma$: **do**
4: $\quad V(\alpha) = \int \alpha(s) b(s) ds$
5: **end for**
6: list = sort(V)
7: return list[0:N]

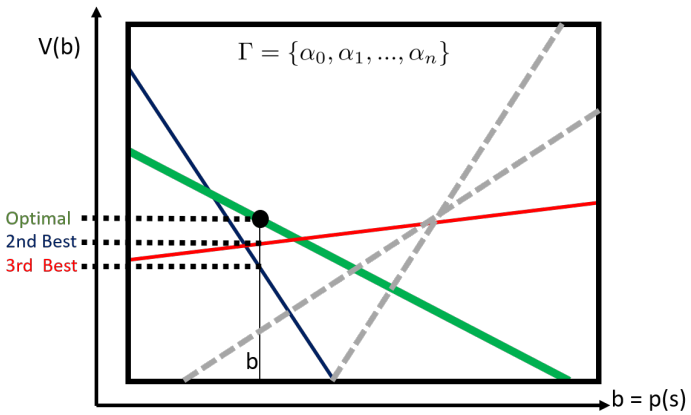


Figure 4.5: Application of the N_Actions() algorithm for belief b and $N = 3$. Policy elements α who have the greatest value at b are chosen, and their questions presented to the human collaborator

4.4.6 Hierarchical Policy Implementation

The implementation of the framework described above results in a hierarchical POMDP policy, composed of multiple monolithic continuous state policies governed by a single discrete space policy. Obtaining such a structure for an active human sensing solution to a robotic search problem follows the four primary steps below, and is summarized in Algorithm 7:

(1) Partition the state space. In the case of the CNR problem, this is done according to rooms, with each partitions boundaries matching precisely with the rectangular walls of each room. Transitions between rooms are mapped out according to doorways, wherein rooms without doorways have probability zero of transition. In the general case, depending on the appropriate target model, transition probabilities should either be distributed randomly between adjacent states or according to inferred target intent. Assign each partition a semantic label (l), creating a labeled continuous state space S_l, keeping in mind that these labels should be intelligible to a human collaborator. The set of labeled partitions becomes S_h, while the transition model becomes $\mathcal{T}_h$.

(2) Following the process laid out in Algorithm 4 and Section 4.3.2, create a semantic dictionary from labeled features within each partition. Construct the action set $A_{q,l}$ for each partition, containing relational indicators attached to each semantic feature, along with a query observation function Ω_l

(3) Solve each partition's associated POMDP using the Algorithm 2, and store each resulting policy π_l in a lower level policy set.

(4) Using the PBVI algorithm [69] and the higher level S_h and $\mathcal{T}_h$, solve for the high level discrete policy across partitions π_h. This policy, along with the lower level set π_l can then be implemented using Algorithm 5.

Algorithm 7 Hierarchical Human Collaborative POMDP Construction

1: Partition and label state space $\mathcal{S}$ into set of S_l
2: Construct semantic dictionary and models $A_{q,l}, \Omega_l, \forall l$ using Algorithm 4
3: $\pi_l = VB - POMDP(S_l) \forall l$
4: $\pi_h = PBVI(S_h)$

4.5 Results on the CNR Experimental Testbed

Hierarchical CPOMDPs were implemented and tested on the Cops and Robots (CNR) hardware platform at the University of Colorado at Boulder's Research and Engineering Center for Unmanned Vehicles. Cops and Robots is a physical simulation of a home environment, as described in Section 4.4.2. For the data presented here, a single human participant played the role of the deputy assigned to assist the cop robot. Semantic labels assigned to the rooms on a given map and objects within each room. These labels are known to both the cop and human, which allows for communication of information through a fixed codebook of possible observations. Individual robotic agents playing the part of the cop and robber were instantiated on Turtlebots running from an Odroid U3 microcontroller on iRobot Create platforms. The cop was equipped with an Xbox Kinect to both relay video to the human, and implemented an OpenCV Blob Detection algorithm to visually detect the robber. Given goal positions from the Hierarchical CPOMDP, the cop implemented low level navigation using the A* algorithm and an occupancy grid representation of object and wall locations. A VICON motion tracking system was used to both provide the cop's fully observable data during runtime and track the ground-truth robber location for post-run data analysis. All elements of the hardware system are networked using a Robot Operating System (ROS) [97] layer, which provides a node based publisher/subscriber interface inter-component communication.

In this work a maximum of three exclusively unique objects per room are considered, each of which has a fixed position and orientation as in Figure 4.1. In Figure 4.1, each room has either 1 or 2 objects, each of which carries 4 ego-centric relational indicators such as "In front of (Object)" or "To the left of (Object)". This results in a semantic dictionary, and corresponding query action set A_q, of size 4 or 8 depending on the room. With the movement action set defined in Section

4.4.3, where $|A_m| = 5$, this results in $|A| = 20 - 40$ per room. Given the arrangement of six rooms, with 10 objects total, framing the CNR problem as a monolithic POMDP via Section 4.3.2 would result in an action set of size $|A| = |A_m| \times |A_q| = 5 \times 50 = 250$. Given the one-to-many nature of action to alpha function assignments discussed in Section 4.3 this implies an optimal monolithic policy would likely be described by significantly more alpha functions than actions. Each of these alpha functions must be regularly condensed both when finding and executing a VB-POMDP policy under the logic of Section 3.6, which when combined with the additional processing for action selection during policy execution results in a significant computational burden during runtime. This limits the ability of such a policy to be used on compute limited platforms, such as those used in Cops and Robots. Furthermore, monolithic VB-POMDP policies explicitly assume a Gaussian transition functions as in Equation 4.2. While this is inevitably only an approximation in physical hardware (such as at the bounds of a state space), such models are more dramatically inaccurate when dealing with the discontinuous transition environments such as walls in CNR. Even if the policy could be constructed to prevent the cop from trying to transition through walls, the robber is assumed to be uncontrollable by the policy. This can easily result in policies solved accidentally assuming an intangible robber, the ghost of a robot perhaps, which further increase the unavoidable error between the policy model and the true hardware capabilities. Approaches such as Switching-Mode POMDPs [24] might alleviate this specific concern, but apply an additional complexity and computation layer on top of the already nearly intractable problem resulting from the large action set. Thus, a monolithic POMDP approach is impractical at best for this problem. However, the hierarchical techniques developed in Section 4.4 specifically allow large problems to be broken up into more manageable sizes, as well as allow discontinuous transition features to serve as subspace boundaries rather than mid-space obstacles.

The human interface, shown in Figure 4.6, visualizes the cop's belief about the robber's position as a heatmap, as well as the cop's position and viewcone detection range (See Section 4.4.3). The interface also displays a real-time feed from the cop's camera and various security cameras placed throughout the space. Each security camera is connected to a Raspberry Pi microcomputer,

Figure 4.6: Cops and Robots user interface: 'robot pull' queries are answered in the lower middle panel with 'Yes/No' buttons; voluntary 'human push' sensor inputs are provided with the structured text input on the lower right panel.

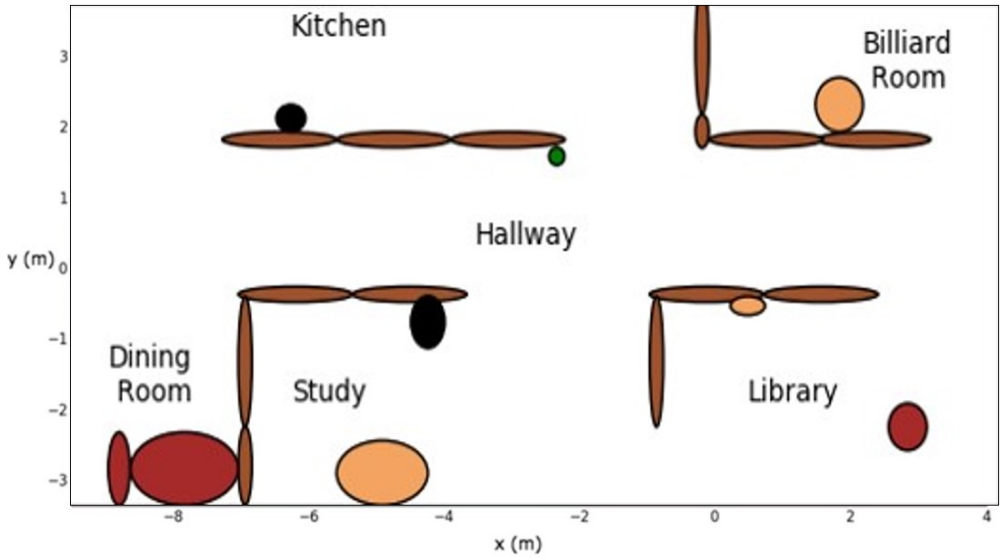
which relays imagery over ROS to the computer running the human interface. The security cameras each allow the human a fixed view of a room. The cop's camera facilitates observations in the cop's immediate area as well as a visual robber detection system. This system is implemented as an OpenCV blob detection algorithm [20], where a sufficient number of contiguous pixels in a given color range triggers a detection event. The robber robot is color coded bright red which is distinct from other colors in the environment, and the minimum threshold of the pixel count can be tuned to allow capture over a range of distances. While in this case a blob detection algorithm is used due to the simplicity of implementation and compatibility with available sensors, other proximity sensor modalities could be substituted without affecting planning or decision making, provided they can be modeled with similar types of likelihood models as described in Section 4.2.

The human plays the role of a sensor, voluntarily passing information to the cop through the semantic codebook embedded in the interface and answering binary 'yes/no' questions passed from the cop (e.g. 'Is the robber in the kitchen?'; 'Is robber in front of fern?'). The set of

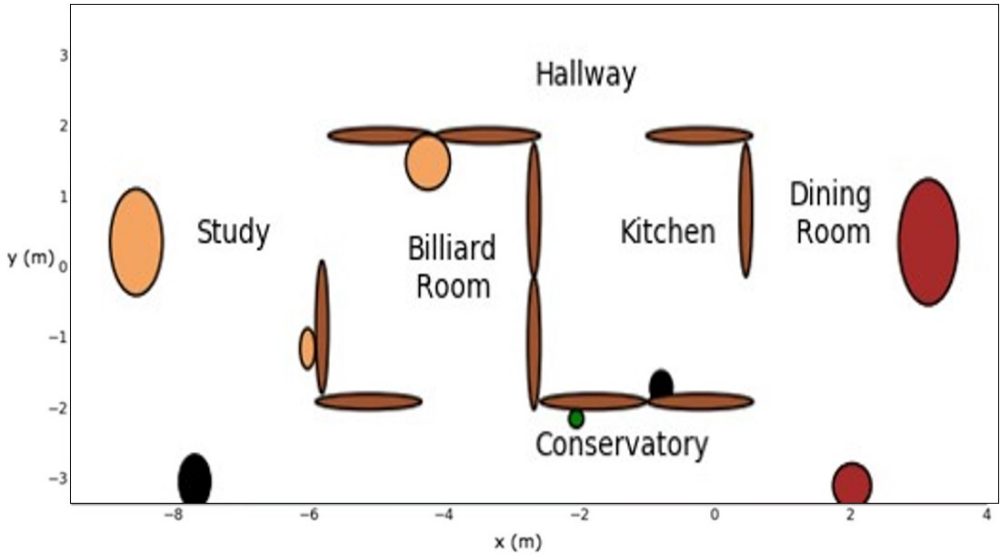
semantic statements constructable using the codebook represents a one-to-one correspondence with the classes of the cop's softmax function observation models. Therefore each observation which the human can volunteer has a corresponding form which can be used a robot query action. To further distinguish these forms of human statement, the following terminology from Section 4.1 and existing literature [49] is used. Robotic query actions in the following text are referred to as 'pull" or 'robot pull' actions, while volunteered human information is labeled as 'push' or 'human push' observations. Human statements generally followed the template form, "The robber (is/isn't) (relation) of (object)". For instance, the statement "The robber is in front of the dining table" initiates the same Bayesian belief update as an affirmative answer to the question "Is the robber in front of the dining table?". The human is also required to validate visual detections of the robber made by the cop, where a positive validation leads to successful capture of the robber and the end of the experiment run. Visual detection instances are suggested by the robot, eliminating the possibility of human error instigating false positives (where the human falsely indicated a successful capture). Also, in the data collected for this work no instances of false negatives, where the human rejected a successful capture, possibly resulting from the well established capability of the human visual cortex to identify well described and differentiated objects in still photos. Thus the data analysis to follow need not account for such instances, though future work leveraging a broader data set may need to.

The Hierarchical CPOMDP policy approximation method was tested on two CNR maps, each with a different rooms structure. The first map, shown in 4.7a, consisted primarily of a hallway running the length of the space, with rooms branching off on both sides. The second map, shown in 4.7b, had the rooms in a semi-bipartite arrangement, with two sets connected through a long hallway and conservatory on the margins. In data collection, the human participant was fully familiarized with the first map beforehand, while the second map was presented as a previously unknown environment.

Each map was tested under 4 input conditions. As a baseline, the Hierarchical CPOMDP policy was implemented without human input, with the cop relying only on its visual sensor to



(a) First/familiar map



(b) Second/unfamiliar map

Figure 4.7: Layouts for first (above) and second (below) maps.

gather information about the world. Second, the policy was implemented with a human who did not respond to the 'robot pull questions', but only provided 'human push' statements at their own discretion. Third, the policy received a human who only responded to 'robot pull' questions, and ignored 'human push'. Finally, the policy was implemented with a human who used both the 'robot pull' questions and 'human push statements' to give information. The resulting times required to

First Map: All Runtimes (seconds)

Room\Case	NonHuman	HumanPush	RobotPull	Both
Library	307	80	51	69
Study	191	132	42	61
Kitchen	123	87	75	40

Table 4.1: Times required for the cop to capture the robber in each scenario posed for the first map

catch the robber are summarized in Tables 4.1 and 4.2. The data shows that introducing human information, whether through 'push' or 'pull' data, shortens the time needed to capture the robber. Intriguingly, the case using both 'push' and 'pull' performs better than either singularly, implying that the robot is obtaining useful information the human did not volunteer through queries, while the human is able to push information the robot was not aware it needed, thus neatly complementing the strengths of each team member.

4.5.1 The Familiar Map

Across each input condition, tests were run with the robber's initial position in 3 different rooms: the Library, the Study, and the Kitchen. The cop's initial position was constant throughout all tests as the far right end of the hallway. The cop also held an identical initial belief for the robber state for each test, with belief dispersed equally between rooms. Each rooms belief was initialized with a single Gaussian, with mean located at the rooms centroid and covariance chosen to extend probability density throughout the room.

The results show that across starting positions, the "Only Robot Pull" and "Robot Pull and Human Push" input conditions tended to require less time to catch the robber than either the "Only Human Push" or"No Human" input conditions. This is expected as the policy was computed assuming the robot would be able to pull information from the human, and thus the resulting information is accounted for in the robot's planning, while the 'push' information is helpful yet unexpected. Furthermore, the "Only Human Push" condition improved on the times for the "No Human" input condition over all cases, demonstrating the utility of unexpected human

semantic sensor data.

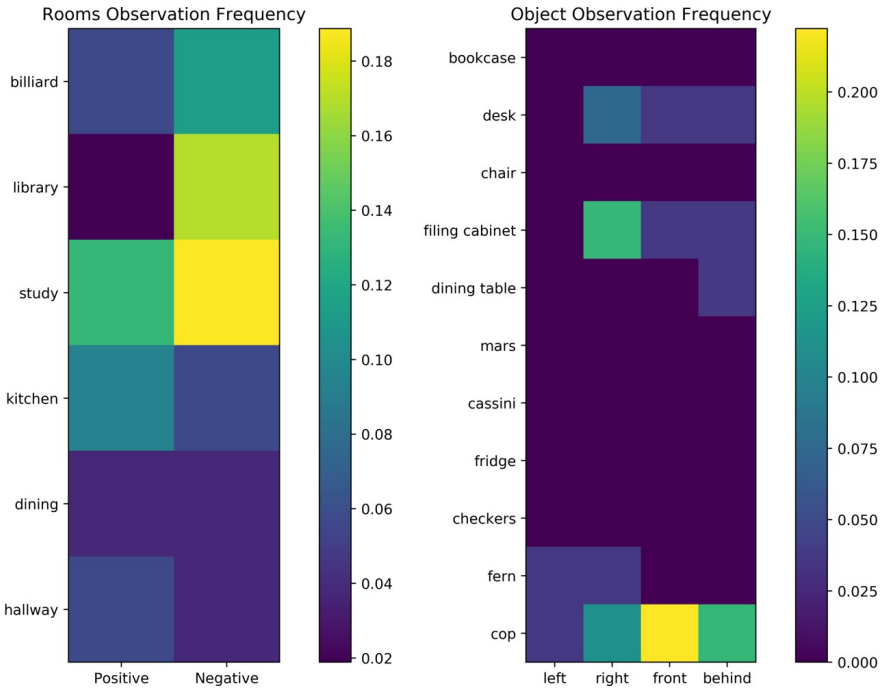


Figure 4.8: Heatmap of the relative frequency of each human observation in the first map.

Over all tests in the familiar map, 79 observations were given, averaging approximately 9 human inputs per test excluding the "No Human" condition. In terms of the propensity for the human deputy to provide negative information (e.g. "I know the Robber is not in the Study") vs. positive information (e.g. e.g. "I know the Robber is in the Study"), about 53% of all statements were positive relations. Limited to observations about rooms, the human only provided positive observations 40% of the time. When referencing objects in each room, 78% of observations were positive. The human referenced rooms about twice as often as they did objects, as shown in Figure 4.8. Furthermore, the sparse nature of the right hand side of Figure 4.8 indicates that many objects were rarely talked about by the human. While this peculiarity may subside given a larger dataset, it is possible that certain objects were not found as useful (explicitly or otherwise) by the human for their task. This implies the existence of a subset of salient features to which the full semantic dictionary could be reduced without substantially reducing the efficacy of the human-robot team. Ideally, this insight could be used to dynamically construct 'appropriate' dictionaries for the task at

hand, ensuring that planning efforts on the part of the autonomy are directed towards information the human is likely to engage with. This topic is explored in depth in Chapter 5.

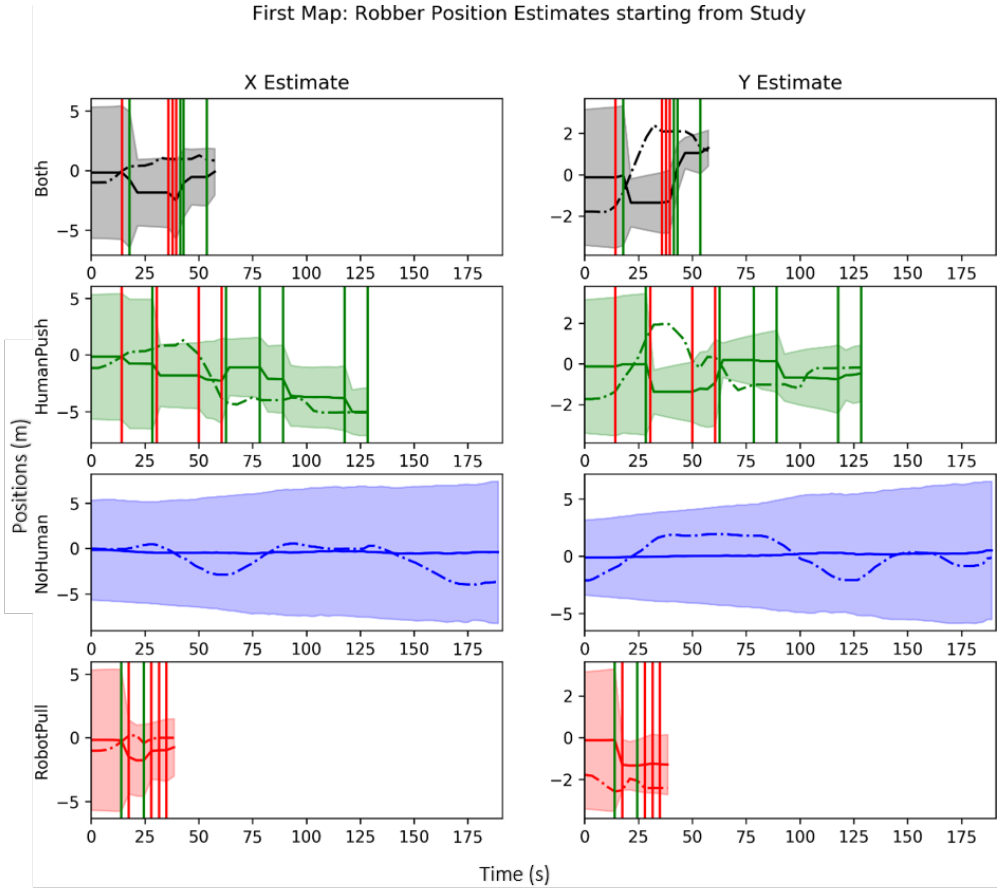


Figure 4.9: Summary of cop's beliefs for the first map. Gaussian mixture mean and 2-sigma bounds of the cop's belief pdf for robber state are plotted against robber's true position (dashed line). Vertical lines are color coded for positive (green) and negative (red) human statements.

The cop's beliefs are summarized in Figure 4.9 for the 4 test runs with the robber starting in the Study. For each input type, the mixture mean and 2-sigma bounds are plotted along with the robber's actual position. Belief compression, using the hybrid pre-clustering technique of Section 3.6, was uniformly applied across rooms to maintain a maximum of 10 mixands per room. The "No Human" input condition sees the belief expanding faster than the cop can gather visual sensor data, as the robber is out of view and moving with unknown direction and velocity. Human semantic observations, shown as vertical lines in the plot, can cause dramatic belief shifts. The robber's

Second Map: All Runtimes (seconds)

Room\Case	NonHuman	HumanPush	RobotPull	Both
Billiard	176	86	87	46
Study	214	183	99	35

Table 4.2: Times required for the cop to capture the robber in each scenario posed for the second map

position is can be seen to be generally well bounded by the cop's belief, which can correct for errors through additional human observations.

4.5.2 The Unfamiliar Map

For the second set of test scenarios, the human observer was familiar with the task and platform, but not with the map itself, shown in Figure 4.7b. The locations of the rooms and positions of the objects within remained unknown to the human until the start of testing, in order to explore whether the human's ability to communicate effectively with the robot was primarily an artifact of their ability to communicate about the current map. If the robot is similarly able to utilize both 'push' and 'pull' human information in an unknown (to the human, not the robot) environment, it implies that the advantage of human information is not diluted by the human's preconceived biases or experience, but rather a result of optimal use of the human sensor in a general sense. Furthermore, the transition structure of the higher level discrete POMDP in the unfamiliar map varies significantly when compared to the known first map, providing additional insight into the ability of hierarchical human-collaborative POMDPs to plan in a variety of structured environments. As in the first map, 4 input conditions were tested over multiple initial robber positions, in this case the Billiard Room and the Study. Across all tests, the cop's initial position was set in the Kitchen, and the belief was evenly distributed between rooms. The timing results from the test, summarized in Table 4.2, are generally comparable with those of the first map, taking an additional 11 seconds to catch the robber on average. The comparison between input conditions also remains consistent, with the unfamiliar map results even suggesting an additional advantage for the "Both"

condition over "Robot Pull Only".

For all tests in the second map there were a total of 66 observations, with an average of 11 human inputs per test excluding the "No Human" condition. In this case about 47% of all statements were positive relations, with 42% positives for room observations and 58% positive for objects. Rooms were referenced almost 3 times as much as objects, with frequencies for each statement shown in Figure 4.10. Interestingly, the human discussed a broader variety of objects in the unfamiliar map, implying a predilection for certain salient features in the first map arose partly from familiarity. However, for each semantic object used by the human, a single relational indicator tended to dominate the relative frequency of observations for that object. This fact might have a geometric interpretation, such that for certain objects the robber was far more often 'in front' due to their placement in the room, or another facet of the human's dynamic understanding of the map. In either case, the semantic dictionary used by the human ended up being much sparser than what was available to them, further indicating the opportunity to focus robotic planning around a smaller number of more salient features.

The cop's beliefs, summarized in Figure 4.11, are once again a reasonable estimate of the robber's position despite slightly more errors.

4.5.3 Discussion

The cop using the Hierarchical CPOMDP approach succeeded in all cases at catching the robber, and was demonstrably quicker in cases where it received and fused human information. Of particular note is the improvement of the "Robot Pull Only" input condition over the "Human Push Only" condition. This implies that information delivered at the policy's request was more valuable than that which the human decided to volunteer. As the policy is meant to approximate the optimal value function for the problem, this serves as evidence of its efficacy.

The system was also able to adapt to false information from the human sensor, as displayed in Figure 4.12. In Figure 4.12a, after the robber passed in front of the security camera in the Study while moving into the Kitchen, the human unintentionally gave a series of false observations, rapidly

shifting the belief from an uncertain but reasonable one to one that was decidedly inaccurate. Later in the same run, the human was able to combine visual information from both the Hallway camera and the cop's viewcone to indicate correctly that the robber had moved into the Dining Room, as shown in Figure 4.12b.

With human information, the policy was able to direct the cop more efficiently. As shown in Figure 4.13a, without any human sensor data the policy primarily directs the cop to patrol the Hallway, popping in and out of individual rooms along the way. This behavior is reasonable considering the Hallway's position as a hub room, where the cop could expect to eventually stumble upon the robber as it moves from room to room. This displays the robustness of the policy's action selection in the absence of expected information. However, when a human operator is able to provide information as in Figure 4.13b, the policy chooses a path through the Library, and ends up tracking the robber directly through the Study, and into the Hallway, finally cornering it in the Dining Room.

The observations given in each scenario show interesting differences between each map, as shown in Figs 4.8 and 4.10. In the first map, where the human was familiar with the map layout and object placement, the observations were sparse. While room observations were dispersed, the human tended to focus on a few key objects, where observations would have a well known and predicable effect on the belief. In the unfamiliar case, the operator gave observations about a broader range of objects, often using the objects in the cops camera feed for reference. In both maps, the object most referenced was the cop itself. In particular, the human operator used the observation "The Robber is in front of the Cop" more than any other across all tests, possibly in attempts to urge the cop forward when the robber was in view but distant.

In summary, the robot was able to actively use human information within a hierarchical POMDP framework to improve its own effectiveness. Such information led to distinct behavioral difference as in 4.13, and shorter capture times. The human, even as an imperfect sensor, was able to actively recognize mistakes and error correct. However, though the POMDP successfully advantage of the structure of the problem, it relied on perfect a prior knowledge of this structure.

The techniques used throughout this Chapter address models of uncertainty but fail in the face of models which are themselves incomplete or uncertain. Furthermore, Figures 4.8 and 4.10 show large parts of the semantic dictionary went unused, while others were heavily exploited by the human. Ideally, the dictionary would contain primarily useful entries, and respond to new information about the problem by expanding appropriately to allow communication about new semantic features. Such an approach is explored in the next Chapter.

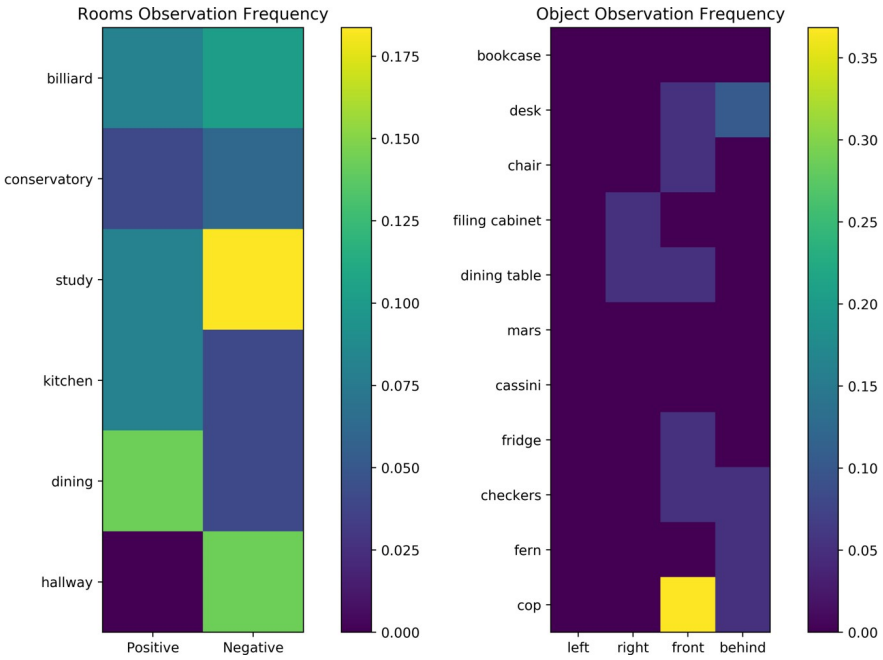


Figure 4.10: Heatmap of the relative frequency of each human observation in the second map.

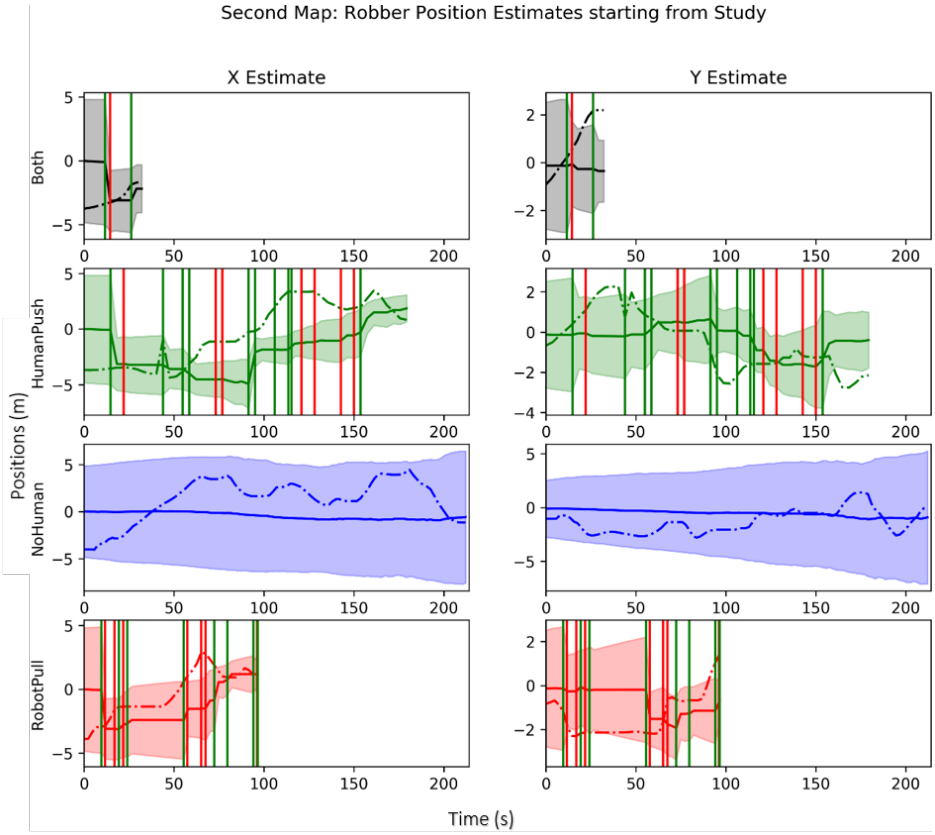
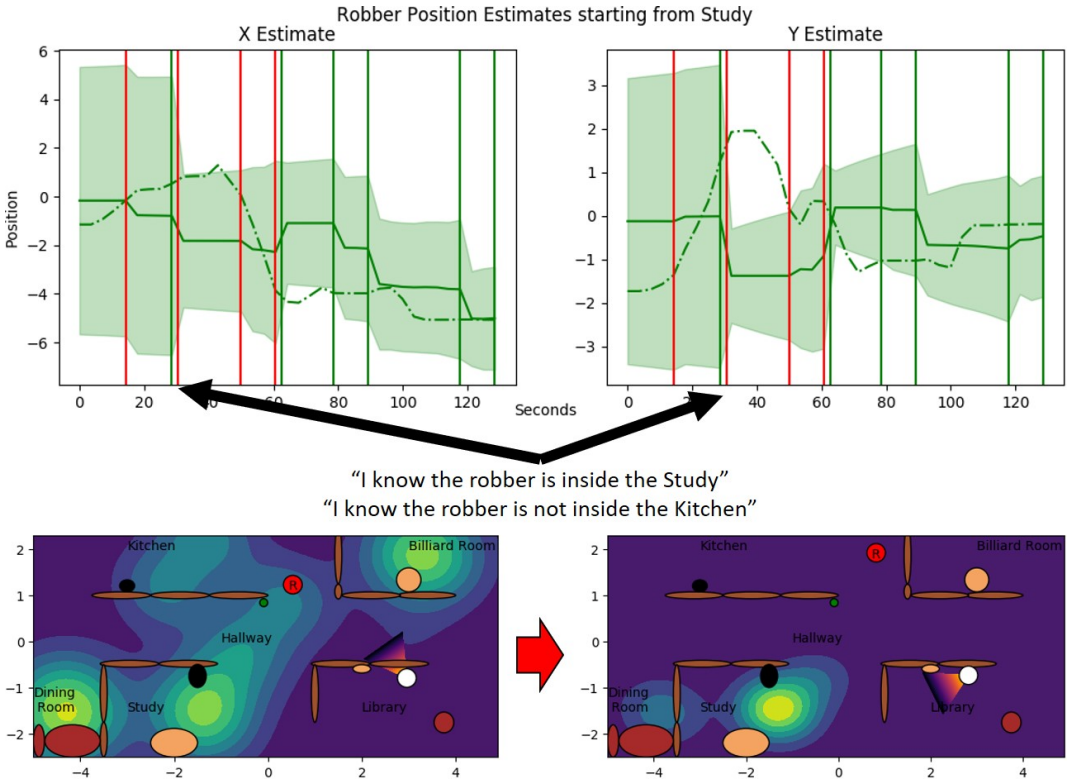
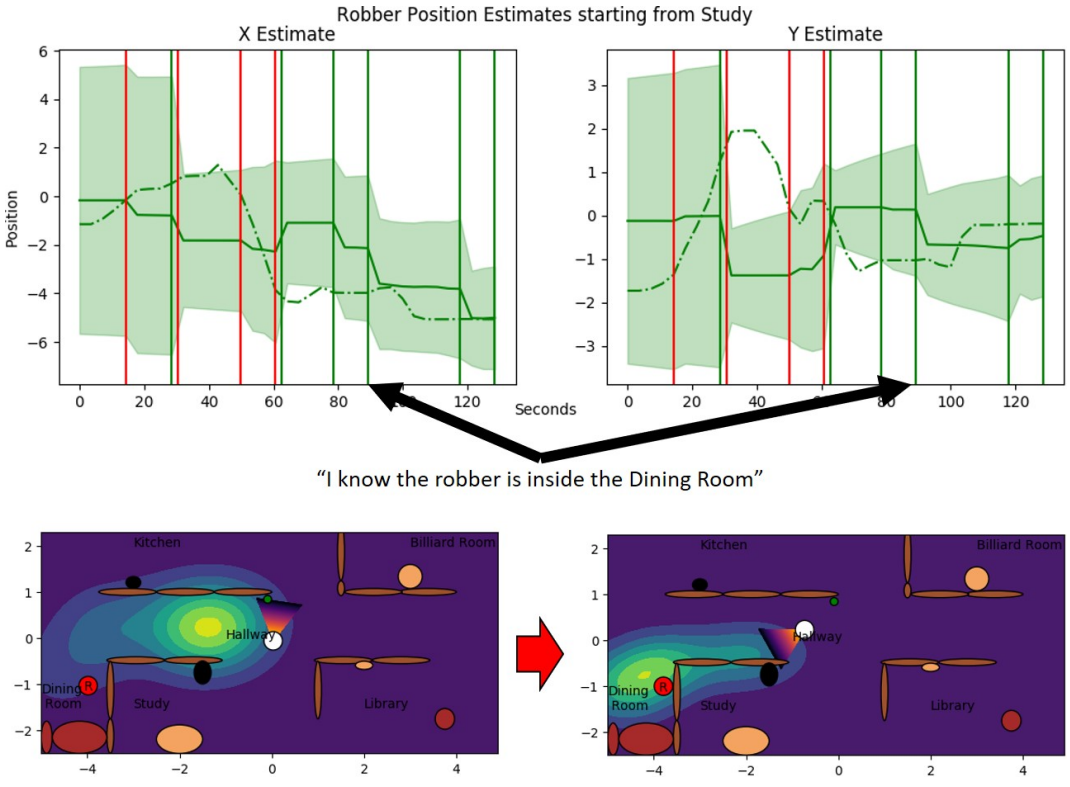


Figure 4.11: Summary of cop's beliefs for the second map. Mean and 2-sigma bounds of the cop's belief are plotted against robber's true position (dashed line). Vertical lines are color coded for positive (green) and negative (red) human statements. As expected, the unfamiliar environment leads to less accurate beliefs in the Human Push scenario.

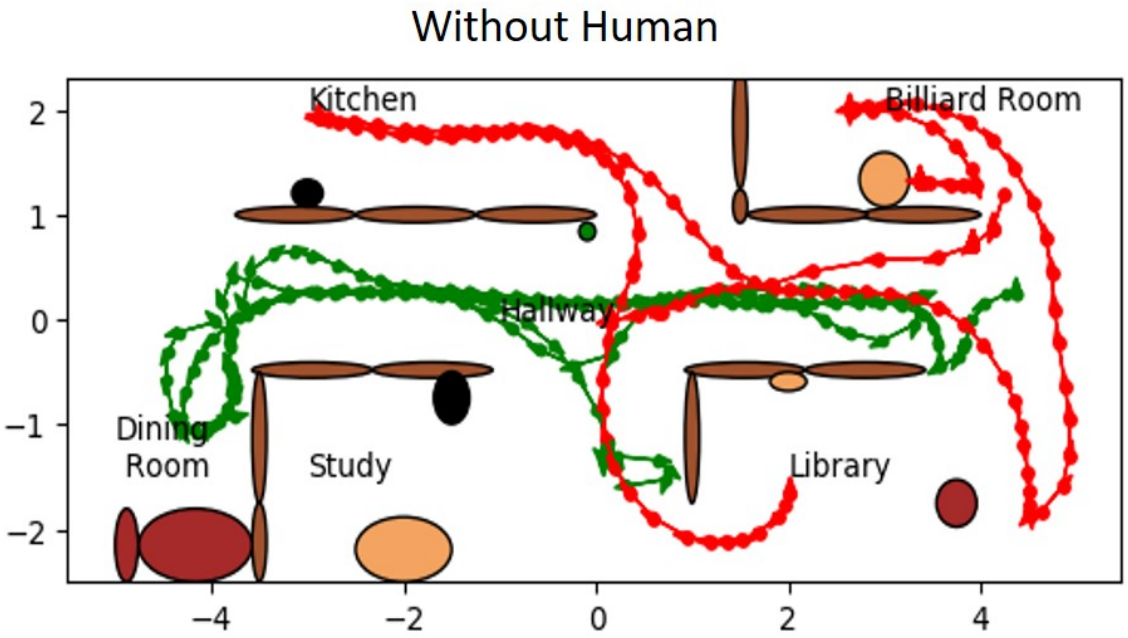


(a) Human gives inaccurate series of observations

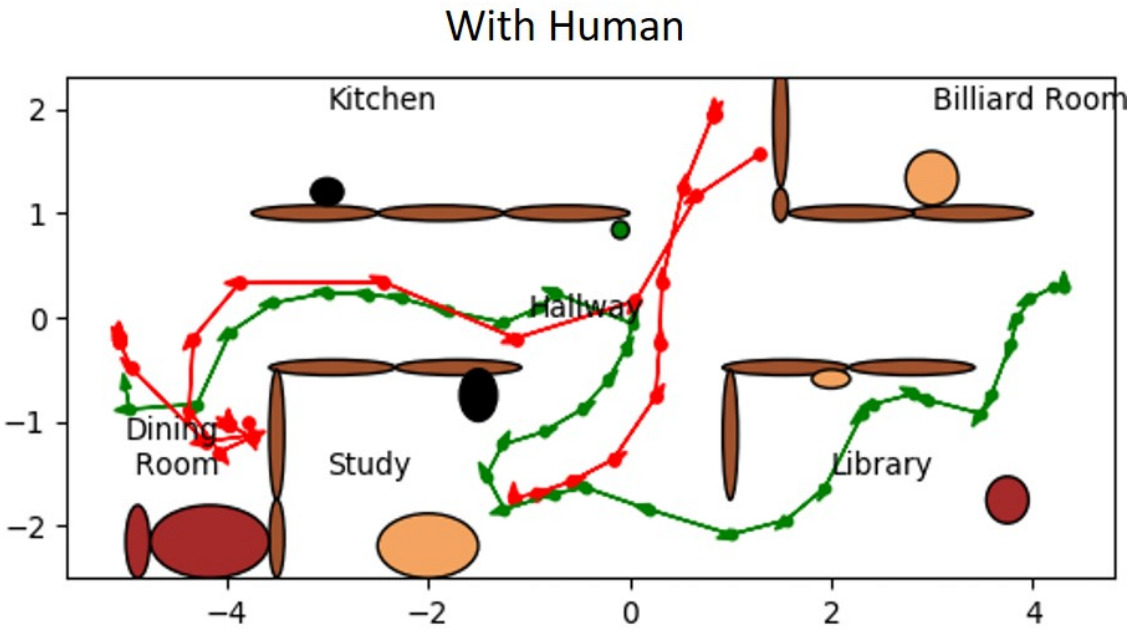


(b) Human shifts belief with information unavailable to the cop

Figure 4.12: Top: The human gives a series of mistaken observations. Bottom: The human gives a helpful statement.



(a) Example paths for "No Human" input condition



(b) Example paths for "Human Push Only" input condition

Figure 4.13: Cop (green) and robber (red) paths without vs. with human sensor input.

Chapter 5

Active Semantic Sensing and Planning for Human-Robot Collaboration in Uncertain Environments

Having built up scalable algorithms and a framework for human-robot cooperative planning in the previous chapters, here we begin to relax one of the key underlying assumptions: the static environment. This chapter introduces the idea of rapidly updating the robot's concept of its surroundings by augmenting a semantic dictionary along with the probabilistic world model for the task at hand. This is accomplished by using human sketches to identify salient features in the environment, and allowing the human to provide meta data such as labels and terrain features for the sketched area. These techniques for transferring human knowledge and perception of both target locations and environmental features in real-time allows the human-robot team to adapt to unknown or changing environments by actively modifying an online planning process to account for incoming information. Unlike previous chapters, which assumed static environments, the contributions of this chapter allow a broader class of problems (that of partially known environments in addition to partially known states) to be solved.

5.1 Human-Robot Search in Unfamiliar Environments

This chapter focuses on dynamic target search problems in which a robot must intercept a moving target as quickly as possible through an uncertain or partially mapped environment. Figure 5.2 shows two versions this problem for an unmanned ground vehicle (UGV) and an unmanned aircraft system (UAS), which will be used as examples. Major uncertainties arise in (but are not

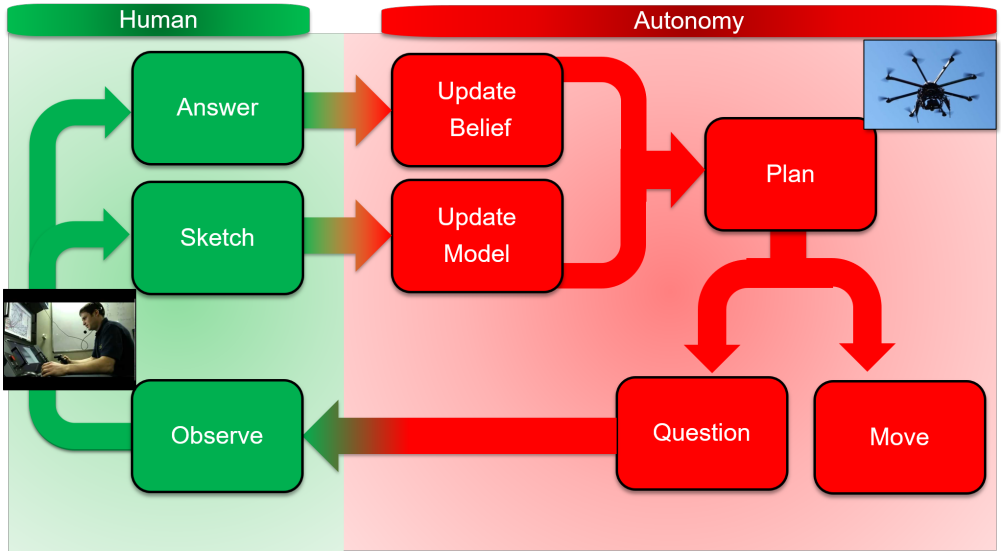


Figure 5.1: Concept of Operations for the Human-Robot Team.

limited to): vehicle motion and transition models (due to unknown terrain or wind fields that are difficult for robots to sense); target models and states (if targets display different dynamic behaviors); and models of language-based semantic 'human sensor' observations (e.g. Fig. 5.4), where 'soft data' can be expressed with variable consistency/accuracy [77, 84].

In each case, the environment includes map regions with differing probabilistic state transition models that affect robot and target motion dynamics. The target's state can also evolve via a stochastic jump-Markov kinematic motion model [12], where the unknown 'active model' (m) may switch randomly at any time within a finite model set and depends on the environment (but not necessarily on the robot's state).

$$m_{t+1} \sim p(m_{t+1}|m_t) \tag{5.1}$$

$$s_{t+1} \sim p(s_{t+1}|s_t, a_t, m_t) \tag{5.2}$$

The robot carries a visual sensor, which allows target proximity detection with a high probability and subject to false alarms. Such sensors can be specified in a variety of ways, such as the omni-directional proximity sensors used in Chapter 3 of this thesis, or the view-cone style of Chapter 4. For simplicity, it is assumed this sensor gives the robot no new environment information. In principle, this assumption could be relaxed with little to no algorithmic changes, while incurring

additional computational and modeling cost. In this work sensors are modeled as simple visual systems capable of running in real-time on mobile platforms, running basic detection methods for a known target, but more computationally intensive approaches such as stereo terrain mapping [98] could incorporate environmental information as well. Finally, the robot can communicate with a remote human, who can view the robot's telemetry data, sensing data, and map information with negligible time delay and generate soft data, but issues no commands to the robot. The robot plans its own movements to intercept the target using available models and observations, assuming an initial prior target state pdf. The robot carries a sensor capable of detecting the target within a fixed 'detection range', while the target is considered captured when positively detected within a smaller 'capture' range.

Figure 5.2: Left: 'Golf Course Problem': UGV navigates terrain to intercept ground target; Right: 'Road Network Problem': UAS tracks a ground target can deviate from road.

5.2 Formal Problem Statement

The problem described above is formally cast as a POMDP, specified by the 7-tuple $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma\}$. Let the continuous states of the mobile robot and target in some search environment be s_r and s_t, respectively. The joint state space is $[s_r, s_t]^T = s \in \mathcal{S} = \mathcal{R}^N$. The human sensor has full knowledge of s_r at all times as well the belief $b(s_t) = p(s_t|o_h^{1:k}, o_r^{1:k}, a^{1:k-1})$,

which is the updated Bayesian posterior pdf given all observations made by the robot $o_h^{1:k}$ and the human $o_h^{1:k}$ through time k. Both s_r and $b(s_t)$ are displayed over a terrain map, which dictates transition model $\mathcal{T} = p(s'|s,a)$ at each point space according to hazards and open areas. Action space $\mathcal{A}$ is a combination of movement action made by the robot which affect its state $\mathcal{A}_m$ and query actions which pull information from the human $\mathcal{A}_q$. The human can provide sketches $\mathcal{H}$, which are semantically labeled spatial areas corresponding to salient features of the space. These labels, along with problem relevant relational indicators, form the semantic dictionary from with the query actions A_q are drawn. The sketches themselves are defined geometrically, as a set of vertices forming a convex polytope in $\mathbb{R}^N$. Each sketch also corresponds to a softmax observation model like those used in Chapters 3 and 4, which is derived based on the polytopes vertices [100] and is the observation model $\Omega = p(o|s,a_q)$ for a given query action a_q. The observation set $\mathcal{O}$ is split into robotic detection observations $\mathcal{O}_r = \{NoDetection, Detection, Capture\}$ and human answers to action queries $\mathcal{O}_h = \{Yes, No, Null\}$, accounting for the possibility of the human not answering. Each sketch creates an enlarged query action space $\mathcal{A}'_q$ such that $\mathcal{A}_q \subset \mathcal{A}'_q$, where each new action is a possible query to the human regarding the relative location of s_t with respect to the new sketch. For a given sketch labeled (l), with a set of relevant relational indicators, $\mathcal{A}'_q = \mathcal{A}_q \cup [Relation \times l]$. Thus the robot is capable of asking more questions after a sketch due to the expansion of possible semantic queries. The robot initially holds some prior $\hat{p}(s'|s,a),\ \forall s$, which need not be reflective of the true transition model $p(s'|s,a)$, and which the human can update by attaching meta information to sketches indicating the relative ease or difficulty of traversal within a region. In this work, for a given sketch h_i, the human may introduce meta information in the form of terrain multipliers δ, such that $\hat{p}_{t+1}(s'|s,a) \neq \hat{p}_t(s'|s,a)$ and $E[\Delta s](a) = \delta_i$. Finally, the reward function $\mathcal{R}$ is stated as a piece-wise function incentivizing capture of the target and slightly

penalizing human queries such that for some distance threshold τ,

$$\mathcal{R} = \begin{cases} 100 & dist(s_t, s_r) \leq \tau \\ 0 & dist(s_t, s_r) > \tau, a_q = Null \\ -1 & dist(s_t, s_r) > \tau, a_q \neq Null \end{cases} \tag{5.3}$$

A POMDP policy π solved for this problem is one which maximizes sum of expected future discounted reward of actions $E[\sum_{t=0}^{\infty} \gamma^t R_t(s, a)]$.

5.3 Human Querying and Data Fusion

As in Chapter 4, the human can act as either a voluntary semantic sensor that voluntarily reports information without request whenever possible, or they can be actively queried by the robot to provide information on request. Unlike in Chap. 4, where the robot's policy technically assumes that the human sensor reports information at every timestep and respond to every active sensing request, it is assumed here that the human will respond to queries based on a static a priori known availability value ξ, such that, for all human observations o_h at all times,

$$p(o_h \neq \text{Null}|s) = \xi \tag{5.4}$$

This leads to an additional observation $o_h = \text{None} \in \Omega_h$. Similarly, while it could be assumed that the human would respond with perfect accuracy to every binary query, here the more reasonable assumption is made that the human has a static a priori known accuracy η, such that, for all human observations o_h at all times,

$$\hat{p}(o_h = j|s) = p(o_h = j|s) \cdot \eta \tag{5.5}$$

where probability is redistributed to the "inaccurate" observations,

$$\hat{p}(o_h = (k \neq j)|s) = \tag{5.6}$$
$$\frac{1}{|\Omega| - 1} p(o_h = j|s) \cdot \eta \cdot p(o_h = (k \neq j)|s)$$

The notation $\hat{p}(o|s)$ and $\hat{p}(s'|s,a)$ denote models used during online execution, in contrast to the true but unknown distributions $p(o|s)$ and $p(s'|s,a)$. This parameterization of accuracy ignores similarity between any two observations. While this simplifies implementation and allows comparative testing of accuracy levels, other models may yield more realistic results; e.g. in cases where the human is modeled via linear softmax model parameters such as Sections 3.1.1 and 4.2 [27, 25, 5]. For a given sketch $h_i \in \mathcal{H}$ with $|h_i|$ relational indicators,

$$p(o_h = j|s) = \frac{\exp(w_j^T s + b_j)}{\sum_c^{|h_i|} \exp(w_c^T s + b_c)} \tag{5.7}$$

where the softmax parameters w_c and b_c determine the steepness of probabilistic boundaries between different labels as a function of s, and thus can be used to determine the likelihood of mistaking semantic labels given s. In either parameterization, these human observations correspond to the "Observe" and "Answer" blocks of Fig. 5.1.

Here humans are treated as sensors which can be actively polled by the autonomy. This requires an explicit dependency on actions to be included in the observation model, as well as additional actions which trigger this dependency. Such actions take the form of queries to the human, with the resulting observation being dependent on which query was sent. Just as POMDP observations can be typically modeled as $o \sim p(o|s), o \in \Omega$, human responses to robotic queries $a \in \mathcal{A}_q$ can be modeled as

$$o_h \sim p(o_h|s, a_q) \tag{5.8}$$
$$a_q \in \mathcal{A}_q, o_h \in \Omega_h$$

These additional query actions can be introduced into $\mathcal{A}$ in one of two ways. Casting them as exclusive options, where either movement or a query can be pursued but not both, minimally expands the action space to the sum of queries and movements, $\mathcal{A} = \mathcal{A}_m + \mathcal{A}_q$. But this can be limiting in situations which benefit from rapid information gathering about models and states together. Instead, we cast the queries as inclusive actions, in which every time step permits any combination of movement and query, $\mathcal{A} = \mathcal{A}_m \times \mathcal{A}_q$. This can drastically increase the size of the

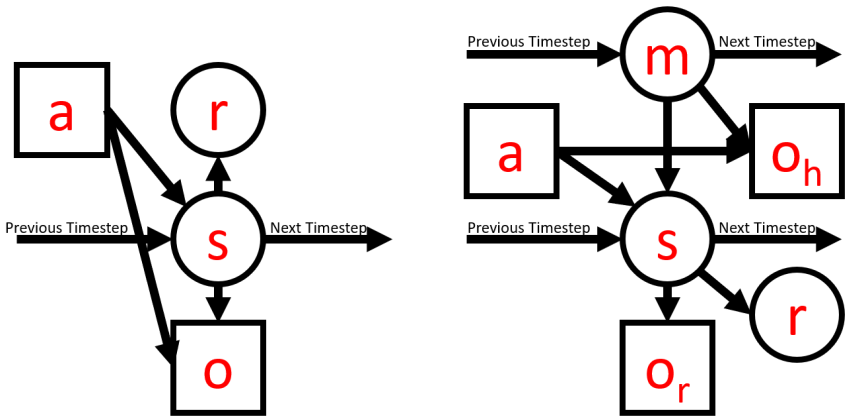


Figure 5.3: Left: Graphical model for Golf Course POMDP; Right: Graphical model for the Road Network POMDP.

action space, but allows rapid information gathering. Here, actions A_m correspond to the "Move" block of Figure 5.1 while A_q correspond to the "Question" block.

The robot's on-board sensor produces a single categorical observation per time step o_r , which must be combined with o_h. These observations can be fused into the belief using Bayesian data fusion [6]. We assume the robot's sensor observations are independent of the human observations given the state such that,

$$p(s_{t+1}|o_r, o_h) = \frac{p(s_t)p(o_h, o_r|s_t)}{p(o_r, o_h)} \propto p(s_t)p(o_r|s_t)p(o_h|s_t). \tag{5.9}$$

Fig. 5.3 summarizes the probabilistic dependencies for the POMDPs describing the search problems depicted in Fig. 5.2. While all observations are state dependent, some observations now depend on actions chosen by the policy. Also, query actions have no effect on the state, and are pure information gathering actions. In these new POMDPs, a combined action might be "Move to North, and Ask human if target South of Lake", which may return a combined observation of "Target is Far from me, and human says target is not South of Lake". So in this case $a = \{North, South/Lake\}$, and $o = \{Far, No\}$.

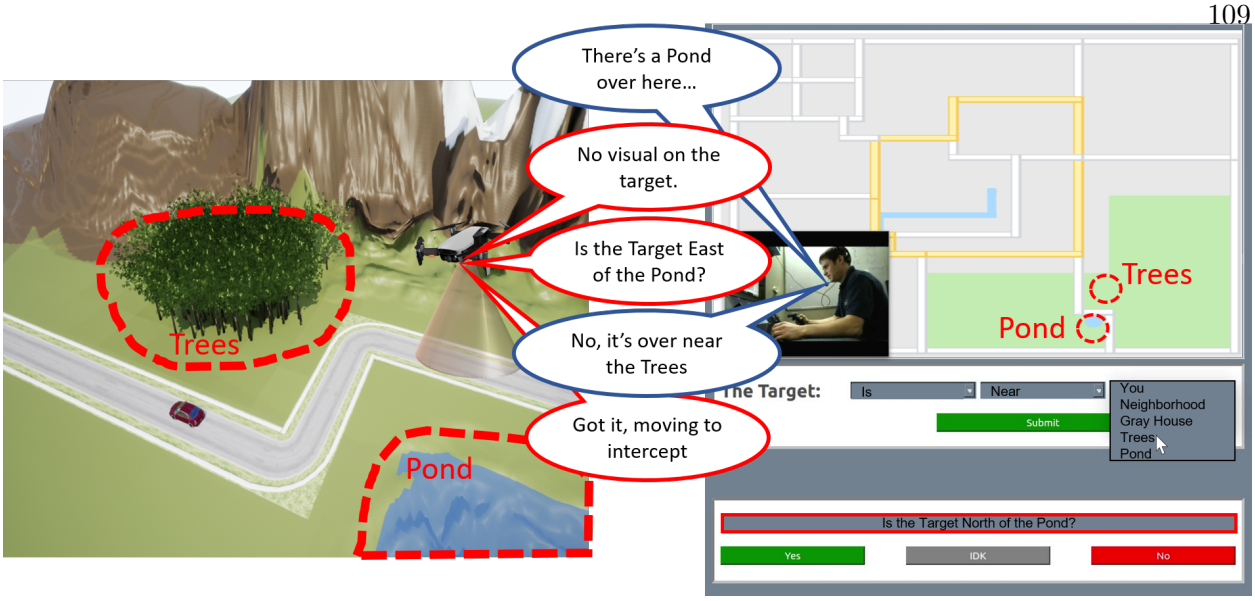


Figure 5.4: Aerial robot searching for ground target with human aiding via semantic labeling and referencing of environment.

5.4 Ad-hoc Semantic Modeling

As discussed previously, many existing approaches for augmenting autonomous robotic reasoning with human-robot semantic dialog only allow robots to reason about limited kinds of uncertainties, i.e. as long as task environments and conditions are known a priori, or do not change in unforeseen ways. This in turn limits the flexibility and utility of semantic communication for adapting to new or unknown situations.

This section of the thesis examines how robots operating in uncertain environments can use semantic communication with humans to solve the combined issues of model augmentation, multi-level information fusion, and replanning under uncertainty in an online manner. Such problems practically arise, for instance, with time-sensitive dynamic target search in areas with outdated/poor prior maps, which lead to uncertain robot and target motion models as well as uncertain human input models. A novel framework for integrated active semantic sensing and planning in human-robot teams is presented in this chapter that formally combines aspects of online POMDP planning, sketch and language-based semantic human-robot interfaces, and model-based Bayesian data fusion.

As shown in Fig. 5.4, this approach features three key technical innovations. Firstly, humans are modeled as 'ad hoc sensors' that push **multi-level** semantic data to robots via structured language, enabling robots to update task models and beliefs with information outside their nominal sensing range. Such multi-level data can correspond to information regarding updates to the POMDPs own transition, observation, or reward models, $\mathcal{T}, \mathcal{O}, \mathcal{R}$, or semantic information pertaining to previously abstracted or ignored state information such as modal behavior or intent. Such dynamic modeling was expressly forbidden by the techniques used in Chapters 3 and 4, which required a static, apriori known POMDP 7-tuple, but are now accessible via ad-hoc semantic sensor models. Secondly, humans can use real-time sketch interfaces to update semantic language dictionaries grounded in uncertain environments, thus dynamically extending the range and flexibility of their observations. Finally, robots actively query humans for specific semantic data at multiple task model levels to improve online performance, while also non-myopically planning to act with imperfect human sensors. These features effectively enable online 'reprogramming' of uncertain POMDPs together with human-robot sensor fusion to support online replanning in complex environments.

5.5 Language-based and Sketch-based Semantic Soft Data

While addressing scenarios of unknown/dynamic environments the human primarily acts as an auxiliary (and imperfect) semantic information source that can communicate with the robot at any time via either one of two interfaces. The first interface allows the human to compose linguistic statements that are parsed and interpreted as target state observations. As shown in Fig. 5.4, these are modeled in the structured form 'Target **is/is not Desc Ref**', where the **Desc** and **Ref** elements are taken from a defined semantic codebook, and the **is/is not** field toggles between positive and negative data [52]. In prior work [84, 6, 100, 61, 27], probabilistic likelihoods were developed for all possible statements in a given codebook to support recursive Bayesian fusion of linguistic human and robot sensor data. Since these works considered only **fully known** environments, all relevant semantic references could be enumerated in advance. However, the environments here are not fully known in advance, so the codebook is at best only partially defined at mission start.

This leads to the second interface: since new map information cannot be obtained from the robot's sensor to augment the codebook, the human may instead do this by providing labeled 2D free-form sketches, which each depict a spatially constrained region on a 2D map display (as in Fig. 5.4 with 'Pond' and 'Trees', or in Fig. 5.2 with 'Sand' and 'Fairway'). Building on [25], the codebook is automatically augmented with the labels of new landmarks/references, so that the corresponding spatial sketch data can also be used to generate suitable soft data likelihood models for the linguistic statement interface (see Fig. 5.4). However, unlike in [25], human sketches here may also provide direct information about probabilistic state transition models, to constrain how the robot and target may traverse certain map areas. While similar 2D sketch interfaces have also been developed for robotics applications [90, 14, 86, 4], their use in the context of multi-level data fusion for planning under uncertainty represents a novel contribution.

In this work, the model for the human sensor is assumed to be known in advance. That is, given a sketch h_i, the autonomy is capable of interpreting human responses to queries regarding it according to some probabilistic likelihood function, explored specifically in Section 5.7.1. Regarding the scope of human soft data in this work, such assumed models, including those for accuracy and availability, are further assumed to be static, and are not learned or adapted online. Furthermore, while the online planning approach in the following sections could in principle be applied to changing reward functions without modification, here only changes to the observation and transition functions are considered. Such changes are taken 'as is' by the automomy, rather than maintaining a belief over the uncertain model parameters as in previous work [78].

Finally, the example problems in this thesis assumed the semantic codebook to be empty at the start of the problem, and built up from scratch by the human using semantically labeled sketches. However, the techniques for online planning and sketch processing apply identically in problems for which the codebook is pre-available yet incomplete. In any case, building the codebook from scratch implies a slightly more difficult problem due to the potentially lacking information available at the start of the problem, and thus provides a more compelling case for human sketch input.

5.6 Optimal Planning and Sensing with Ad-hoc Semantics

The presence of model uncertainties, sensing errors, and process noise makes optimal planning quite challenging. As stated in previous sections, the POMDP family of decision-making algorithms is especially well suited to these types of combined uncertainty.

The primary challenge arising from casting a dynamic/unknown scenario as a POMDP lies in the ability of the human to modify the models $\mathcal{T}$ and $\mathcal{O}$ online in an unmodeled ad-hoc fashion via the sketch interface. These modifications can happen rapidly, and might change large swathes of each model with a single sketch. Bayes-Adaptive POMDPs [78] allow POMDPs to learn the parameters of $\mathcal{T}$ and $\mathcal{O}$, but require gradual changes to their parameters and static $\mathcal{T}$ and $\mathcal{O}$. In the scenarios under consideration, $\mathcal{T}$ and $\mathcal{O}$ change unpredictably with new sketches. This issue renders "full-width" offline point-based POMDP planners [69, 95] inapplicable, as they require models of how $\mathcal{T}, \mathcal{O}$ and Ω change.

Cast as a POMDP, robotic autonomy easily incorporates information gathering actions. Unlike the Value-of-Information [49] and Human Observation Providers POMDP [77] frameworks, which relied on static environments and fixed codebooks regarding state information, these problems require the ability to dynamically encode previously unknown environments and leverages multiple types of human input.

Online POMDP approaches [89], which eschew the process of pre-solving the policy prior to execution in favor of interleaving steps of policy execution and search, have successfully been used to address large observation spaces [99], continuous state spaces [40], and more recently, dynamic ad-hoc models [25]. These algorithms make use of a 'black-box' generative model, requiring only the 'current POMDP problem' at time of execution, making them good candidates for solving problems with dynamic model uncertainty. While they require significantly more computation during policy execution, online POMDP algorithms are particularly adaptable to changing their underlying models thanks to their reliance on rapidly building the simulated results of their actions. Some have even been used to address a changing understanding of the underlying POMDP, but have

been limited to considering transition model changes [57]. Thus, model changes can be incorporated quickly though simulation, without having to broadly modify an existing policy.

5.7 Online Planning and Model Revision

POMDP policy approximations typically assume a fixed problem during policy search and execution, meaning each element of $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma)$ remains fixed during policy search and execution. However, this limits their ability to find policies in poorly modeled but learnable environments. A human sensor can address this shortcoming: in addition to answering queries about target states, the human can help edit the POMDP by providing model-related information while requiring minimal expert knowledge. Assume that the human sensor provides updates to uncertain state transition models $\mathcal{T}$ with soft semantic data during execution. The spatial location of alterations to $\mathcal{T}$ are then labeled semantic references for new soft data options. These modifications to the observation and transition models imply the creation of a new POMDP to be solved, a distinct 7-tuple with a new Ω and $\mathcal{T}$.

5.7.1 Revising POMDP models through Sketching

A sketch, or drawing in the 2D plane, begins as a set of points $\{P\}$, whether with a pen, touchscreen, mouse, or other modality. The set $\{P\}$ is tagged by the human with a natural language label (l), and any applicable meta information δ as in Section 5.2. The communication of the label and meta information in this work is accomplished using fillable text boxes and radio button selection of a pre-selected set of meta data options, but could also be acquired entirely from natural language through word association methods such as Word2Vec [63] or other deep learning methods.

From the set $\{P\}$, the ordered verticies of a convex hull $v_i \in \{V\}$, where $\{V\} \subset \{P\}$ are obtained using the Quickhull algorithm. $\{V\}$ is progressively reduced using Algorithm 8 until it reaches a predefinied size by repeatedly removing the point contributing the least deflection angle to the line between its neighbors, calculated via the Law of Cosines for the vertex pair vectors

$\overrightarrow{v_{i-1}v_i}$,

$$\Theta(v_i) = \arccos\left[\frac{\overrightarrow{v_{i-1}v_i} \cdot \overrightarrow{v_iv_{i+1}}}{\|\overrightarrow{v_{i-1}v_i}\|\|\overrightarrow{v_iv_{i+1}}\|}\right] \tag{5.10}$$

in a procedure inspired by the Ramer-Douglas-Peucker algorithm.

Algorithm 8 Sequential Convex Hull Reduction

```
Function: REDUCE
Input: Convex Hull {V}, Target Number N
if {V} == N then
  return hull
end if
for v_i ∈ {V} do
  Θ(v_i) ← angle(v_{i-1}, v_i, v_{i+1})
end for
{V} ← {V} \ argmin_v Θ(v)
return REDUCE(hull,N)
```

This heuristic was chosen as a proxy for maximizing the area maintained by the reduced hull, but in practice other heuristics could also be used. The reduced set is then used as the basis for softmax model synthesis [100], in which M points define a final sketch h_i associated with a softmax function with $|h_i| = M + 1$ classes. In general, these are 1 interior classes and M exterior classes, each associated with a relational indicator. These classes are associated with relational indicators using the methodology described in Section 5.10.

A limitation of sequential hull reduction is the need to pre-define a set number of points at which to stop the progressive reduction. Ideally, the "natural" number of points needed to maximize some criteria could be chosen on a sketch by sketch basis in a geometric analogue to the Bayesian Information Criterion (BIC) score for model selection. Note that this approach for convex hull reduction differs from that used in Geometric SVM classifiers in that reduces the vertices comprising the boundaries of the hull itself, rather than reducing the hull area in a feature hyperplane.

An example sketch is shown in Fig. 5.5, where the initial input consists of 661 points, shown in black, making a roughly rectangular shape. The Quickhull algorithm is applied to find 21 points, shown in red, defining a convex hull on the set of points. These points are then used as input to

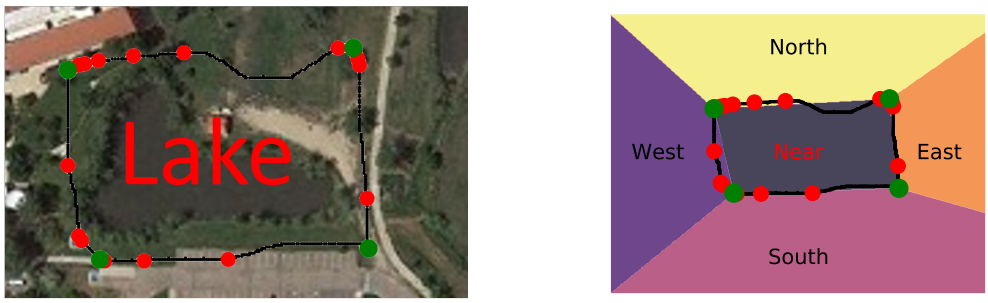


Figure 5.5: Left: Convex hull verticies (red), reduced to 4 points with sequential hull reduction (green). Right: Softmax function and labels resulting from reduced points.

Algorithm 8, which further reduces the number of points to the 4 points shown in green. From these 4 points, a softmax model consisting of 5 classes, a "Near" class and 4 relational indicators, is then synthesized. Semantic observations can now be constructed using the label (l) given by the human when they made the sketch in the form, $o =$ "The Target is *relation* of *label*." The area inside the sketch h_i is now modelled as subject to any modifying meta data δ_i provided by the human. In this work, where such information is provided, the δ acts only as an assumed transition function $\hat{p}(s'|s,a)$ modifier such that for a sketch made at time (t):

$$\hat{p}_t(s'|s,a)! = \hat{p}_{t+1}(s'|s,a) \tag{5.11}$$

$$E[\Delta s]_t(a) = X \tag{5.12}$$

$$E[\Delta s]_{t+1}(a) = \delta_i X, \forall s \in h_i \tag{5.13}$$

$$\tag{5.14}$$

where the $\in$ operation filters states located within the convex polygon defined by the vertices of h_i. This introduces a scaling factor, in which for any given state simulation in an online POMDP, the transition simulation must check if the state is contained in any previously drawn sketches. Alternatively, the transition modification can be stored in either a continuous function such as a unnormalized Gaussian Mixture, or a discrete grid as appropriate to limit computational expense. In the experiments in following sections that explore the use of human meta data to modify tran-

sition functions (specifically Section 5.11.1), the discrete grid approach is used, wherein a grid of nominal values $\alpha(s)$ is pre-initialized, then updated to store meta data:

$$\alpha(s) = \delta_i, \forall s \in h_i \tag{5.15}$$

Naturally in problems without specific transition function modification, Section 5.11.2 and 5.12, such a storage solution is not included. In this specific work, the human can communicate the appropriate δ value for a given sketch as one of a given set of semantic choices to indicate the effect of the terrain contained within the sketch on the movement speed of any ground vehicle passing through it. Each choice is assigned a δ value, corresponding to multipliers on a vehicles nominal speed. Implementation details are given in Section 5.11.1.

The query action set A_q and observation function Ω are modified alongside the transition function $\mathcal{T}$, introducing the newly minted sketch $h_i \in \mathcal{H}$ and its corresponding softmax function in the manner prescribed in Section 5.3. The observation set $\mathcal{O}$ remains the same in this work, but through the modified $\Omega = p(o \in \mathcal{O}|s, a)$ observations corresponding to the new actions in A_q can now occur. As per Section 5.3, the availability ξ and accuracy η parameters of the human collaborator are static across any model revisions, and their effect is folded into changes to Ω through Equations 5.2-5.4. In this way, both parameters are accounted for in planning through their effect on the observation liklihood.

In principle the human could provide nothing more than the sketched points $\{P\}$, with no meta information or label. The meta data is processed as available, and any lack of such data can simply be ignored by not updating the transition function. However, the robot will still need to indicate specific sketches to the human in queries, potentially requiring it to resort to referring to "Sketch 1" vs "Sketch 2". Semantic labeling of sketches ultimately leads to better understanding on the human's part of the robots queries and opens the door to the use of the semantic label to infer additional meta data using natural language processing techniques.

5.7.2 Online Planning with Revised Models

Having updated the POMDP with revised transition functions and observation likelihoods through a human sketch, a POMDP planner can be brought to bear to produce an optimal policy. However, this newly available information creates its own issues, as most POMDP solvers make two key assumptions on transition models. First is that the model is temporally static

$$p_{k+1}(s'|s,a) = p_k(s'|s,a), \forall k \tag{5.16}$$
$$\hat{p}_{k+1}(s'|s,a) = \hat{p}_k(s'|s,a).$$

where $\hat{p}_k(s'|s,a)$ is the internal transition function in use by the robot at time k, in contrast to the true underlying (unknown) model $p_k(s'|s,a)$. The second is that the model being used by the solver is identical to the model being used during the execution of the policy, $\hat{p}_k(s'|s,a) = p_k(s'|s,a)$.

The first assumption is particularly important for infinite horizon offline solvers, where Bellman-backup steps generate approximations of the value function agnostic of whichever timestep a reward may be achieved on. Finite horizon offline and online solvers can cope with this limitation [89, 92], but require that even non-stationary transition models be known for all times prior to generating a policy. The Adaptive Belief Tree (ABT) [57] online planner was developed to address changing environmental conditions with regard to the transition model, but carries no provision for the observation model alterations necessary in this work.

For $\mathcal{T}$, we assume the underlying model isn't changing, while the robot's understanding of it might, i.e.

$$p_{k+1}(s'|s,a) = p_k(s'|s,a) \tag{5.17}$$
$$\hat{p}_{k+1}(s'|s,a) \stackrel{?}{=} \hat{p}_k(s'|s,a)$$

However, we make no assumption on the nature of this understanding change nor the timing. For the observation model however, both the true and internal model change when given a new sketch. All of these model changes imply that the robot must solve a different but related POMDP after

each human sketch. With this in mind, we next consider viable approximations for solving such POMDPs.

The Partially Observable Monte Carlo Planning (POMCP) algorithm proposed in [89] is particularly promising due to its use of generative 'forward' models and online any-time characteristics. While the original implementation of POMCP uses an unweighted particle filter for belief updates, the authors of [36] note that, for problems with even moderately large $\mathcal{A}$ or $\mathcal{O}$, a bootstrap particle filter allows for more consistent belief updates without domain specific particle reinvigoration. This weighted particle filter approach coincidentally also provides a solution to the dynamic modeling problem. Each belief update is carried out using the most up-to-date model, while changes to the model only affect future timesteps. As model alterations require solving a different POMDP after each sketch, this approach allows each planning phase to be treated as the start of a brand new POMDP solution. Our procedure for carrying out POMCP planning while handling dynamic model updates from a human is detailed in Algorithm 9. Certain aspects of this procedure, such as our use of a discretized grid to store transition model modifiers (α) noted in the previous subsection or the semantic observations included with each sketch (described in more detail below), readily adapt to more general problems.

Algorithm 9 Planning with Human Model Updates

B_k = Particle_Set(Size = N)
$\alpha(s)$ = Discrete_Grid()
repeat
  $[a_m, a_q]$ = POMCP(B) (Ref. [89])
  $s \sim p_k(s'|s, a_m)$ (unknown state)
  $o_r \sim \bar{p}_k(o_r|s)$ (robot sensor observation)
  $o_h \sim \bar{p}_k(o_h|s, a_q)$ (human query answer)
  B_{k+1} = Bootstrap_Filter(B_k, a_m, a_q, o_r, o_h)
  if New Human Sketch ($\langle$) **then**
    $\delta = \langle.\delta$, (human assigned state modifier)
    l = $\langle.l$, (human assigned label)
    for $s \in \langle$ **do**
      $\alpha(s) = \delta$
    end for
    $\Omega_{k+1} = \Omega_k \cup (l \times [Near, E, W, N, S])$
  end if
until Scenario End

5.8 Predictive Tree Planning

Implementation of an online POMDP solver such as POMCP in a time-limited system raises an additional complication, regardless of human involvement. A typical offline POMDP solver computes a policy prior to execution and requires significantly less time to execute actions recommended thereby, as shown in Figure 5.6.

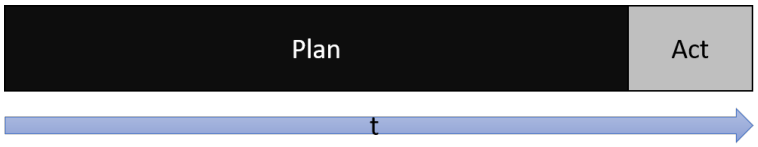


Figure 5.6: Typical Offline POMDP Time Allocation

Online planners like POMCP interweave planning and execution during runtime, leading to a time allocation structure like the one in Figure 5.7. This structure, when implemented directly on a robotic platform, leaves the robot immobile while planning and non-cognizant while acting. Planning for timestep $k+1$ cannot take place until the observation at timestep k has been fused into the belief, as POMCP operates on the current belief as the root node of its planning tree. If this challenge could be overcome, it would be ideal to operate according to the structure in Figure 5.8, where the planning for timestep $k+1$ takes place during the execution of action k. Additionally, given that POMCP is limited in this work to a maximum decision time, it would be ideal to allocate exactly as much decision time as action k takes to execute, as shown in Figure 5.9. An approach to achieving this structure is proposed here, known as Predictive Tree Planning.

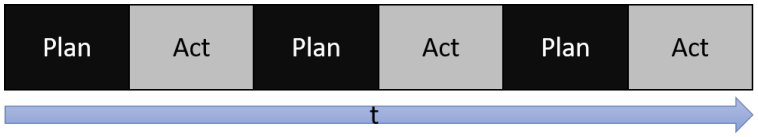


Figure 5.7: Typical Online POMDP Time Allocation

If such planning can take place only after receiving an observation, the tree structure of POMCP can be leveraged to pre-plan by predicting which observation will be received. For a

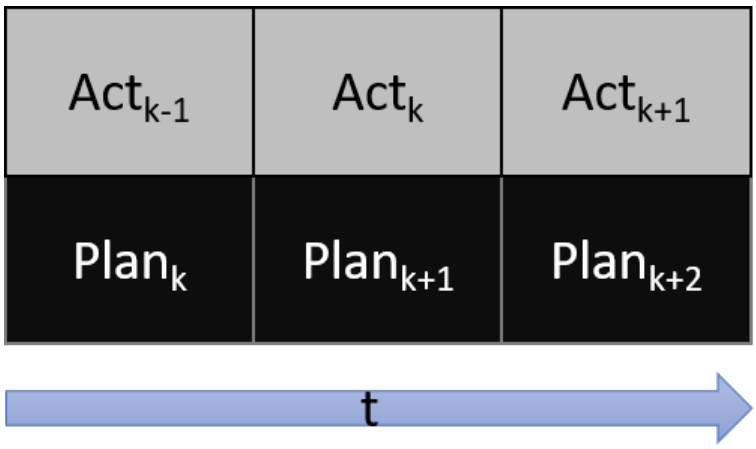


Figure 5.8: Simultaneous Planning and Execution Time Allocation

given action $a_k \in A_k$ and belief b_k, the observation model $p(o|s)$ is used to direct tree samples to nodes with history $b_k a_k o$. The weighted proportion of samples in nodes $[b_k a_k o^1, b_k a_k o_2, ..., b_k a_k o_{|\Omega|}]$ represents the distribution $p(o_k|b_k, a_k)$,

$$p(o_k = o|b_k, a_k) = \sum_{s_k \in b_k} w_s p(o_k = 1|s_k, a_k) \tag{5.18}$$

which is the observation probability distribution for the current belief and action. In order to spend $\Delta t(a_k)$ time planning during the execution of a_k, $|\Omega|$ plans can be pursued, each allocated $\Delta t(a_k)p(o_k = o|b_k, a_k)$ time. That is to say, after carrying out a belief update assuming a future observation, we generate a POMCP policy using time in proportion to the probability of receiving said observation. Upon receiving the true observation at the end of the action execution phase, the autonomous system can immediately begin executing the action from to the policy corresponding to that observation. That is, having generated N policies $[\pi(o_{k+1} = 0), \pi(o_{k+1} = 1), ..., \pi(o_{k+1} = N)$, and received the observation $o_{k+1} = 1$, the policy $\pi(o_{k+1} = 1)$ can be used to find action a_{k+1} without delay.

It must be noted that although this approach works well for all of the example application problems considered here (as there are only two observations the drone can receive during a movement action) it faces significant scalability issues. Creating and choosing between significantly larger numbers of predictive planning trees reduces the accuracy of the policy approximation

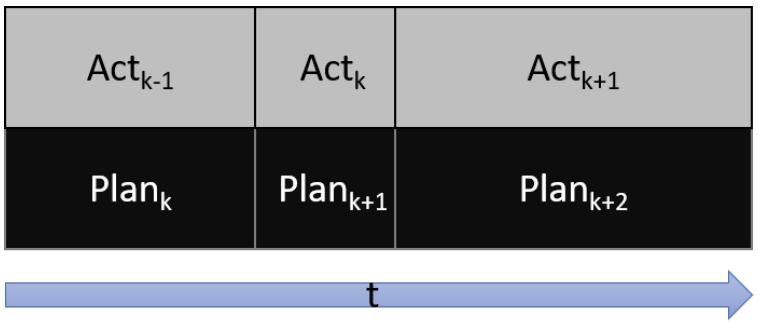


Figure 5.9: Irregular Action Duration Time Allocation

sharply, especially if many observations hold similar likelihoods. This can be addressed in future work through a likelihood threshold to filter out rare observations, combined with a heuristic to determine actions when such rare observations are seen. Alternatively, observations could be collected into meta-observations when they lead to similar beliefs, echoing the discretization of observations spaces seen in [72].

5.9 Multi-Level Active Information Gathering

To accommodate stochastic switching dynamics, it is generally convenient to include mode states m, specified as discrete integers $m = \{1, ..., N_m\}$, as shown in Fig. 5.3 (right) for the Road Network problem, which follow hidden Markov models to dictate alterations to the target state transition models,

$$\bar{s} = [s, m]^T, m' \sim p(m'|m)\ s' \sim p(s'|s, a, m'). \tag{5.19}$$

where $p(m'|m)$ is the transition function between modes, and $p(s'|s, a, m')$ is the modified state transition function accounting for the current mode. This additional discrete state variable can be easily handled by POMCP and other Monte Carlo tree search approximations, which rely on generative black box simulators and support edits to $\mathcal{T}$ and $\mathcal{O}$. This is similar to the offline approximations based on switching-mode POMDPs [24], which can also accommodate hybrid switching dynamics but require stationary models. Such hybrid model extensions also open the door to active semantic queries for specific human observations o_h pertaining to m. This can greatly enhance the

Bayesian belief for m, as such switching mode states are not typically observed directly by robot sensors, which indirectly improves the ability of the system to accurately estimate state s.

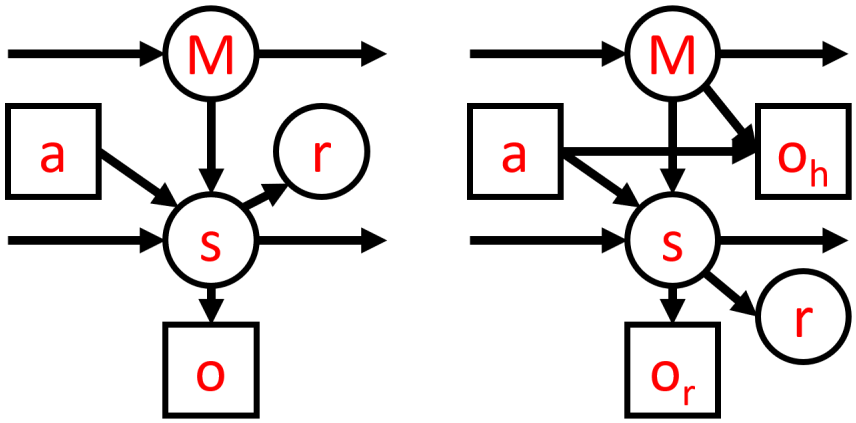


Figure 5.10: Left: Probabilistic Graphical Model of a Jump Markov State Space Model, shown at a single arbitrary time slice, Right: The same model, with the inclusion of semantic queries to human sensor regarding the discrete mode m

With this modification to the generative model, the probability of different state transition models being in effect can be explicitly represented via particle approximation as

$$P(m = x) = \frac{1}{N} \sum_{n=0}^{N} \mathbf{1}(m_n = x), \tag{5.20}$$

where $\mathbf{1}(m_n = x)$ is an indicator function applied to identify if the mode of particle n is a given $x \in \{1, .., N_m\}$. This allows queries to be constructed in the same way as those presented in Chapter 4, but referring instead the mode segment of the state vector $\bar{s}$. For instance, Section 5.11.2 considers a problem where the modes governing the transition probabilities represent whether or not a target is on a road. Thus, actions can be taken such as "Ask the human if the target has gone off-road", which can improve the robot's estimated belief of the target's speed. In this way, the model changes brought about by queries are anticipated by the solver, and don't imply the need to solve a different POMDP. With the mode treated an additional state variable as in Equation 5.19, the belief can be updated using Equation 5.9. The bootstrap particle filter used in Algorithm 9 approximates this belief update through use of weighted particles.

The mode represents a fully modelled dynamic element of the problem, and as such changes

to it do not constitute unmodeled changes to the POMDP in the same way that sketches do. Rather than relying on the human to provide sketched updates to the transition model $\mathcal{T}$, the robot can directly query the human about the mode to update its understanding of the targets current dynamics. Thus, sketching and modal queries provide distinct but complimentary methods for receiving dynamics information, through flexible unmodeled sketches or more restrictive modal belief updates.

The introduction of a discrete mode creates a hybrid continuous/discrete state space with partially constrained dynamics, which can complicate some continuous space offline POMDP algorithms [72, 26]. One could potentially address this using a Hierarchical Gaussian Mixture belief representation as in Chapter 4, with each mixand being tagged as belonging to a sub-mixture for a mode in the same way Chapter 4 tags mixands as belonging to different rooms. In order to make use of a POMCP type algorithm for planning, these mixtures would then have to be sampled at each timestep. The use of particle filters in concert with POMCP here instead allows such states to be handled gracefully through a single generative model.

5.10 Sketch Generation and Preparation

This section discusses the pipeline by which sketches are processed and augmented to serve as observation likelihoods for POMDPs. First, a method of automatic sketch generation is discussed which can be used to simulated human input under a range of tunable parameters. Then a Monte Carlo based automatic labeling method is detailed which can be used to semantically label softmax classes resulting from sketches, whether drawn by a human or simulated. Next a methodology for augmenting bearing based softmax models with range information is introduced, before all elements are packaged together into a single Sketching Pipleine. This pipeline is capable of end-to-end sketch generation and use for simulation based experiments, as well as improving the usefulness human drawn sketches with no additional human input.

5.10.1 Parameterized Sketch Generation

A distinct drawback of human-robot interaction research is the time associated with collecting human subject test data. For this work in particular, the tightly integrated planning and sensing at the core of information gathering problems further complicates the collection of human data, as the POMDP must be run in real-time while simultaneously receiving real-time input from a human. Also, while observations from a given softmax model can be drawn probabilistically to simulate a human response, the motivating problem requires the real-time generation of new softmax models from sketches. Thus gathering large sample sizes becomes extremely time intensive. However, such sample sizes are highly desirable for the purpose of investigating the value of online POMDPs in human-robot teams.

This section describes the Parameterized Sketching Emuluator Utilizing vertex Downsampling (PSEUD), Developed for the purpose of investigating the response and efficacy online human-interacting POMDPs. PSEUD allows the generation of convex sketched polygons according to a learnable set of parameters, effectively drawing a sketch from a distribution of identically parameterized sketches. In this way, the effect of a human's sketch input on the POMDP can be quantified for a broad variety of possible sketching styles and idiosyncrasies, even those likely to represent rare edge cases in the actual human population.

A 2D PSEUD sketch is parameterized as the 5-tuple $\{\mathcal{X}, r, \sigma, \lambda, \psi\}$. The centroid $\mathcal{X} = \{x, y\}$ and characteristic size r are left as features dependent on a given landmark around which the sketch is being made. Specifically, in this work, both $\mathcal{X}$ and r are defined in the 2D plane corresponding to the target state s_t, while more generally they are defined whichever state space plane the sketch is being drawn on. The chosen landmark might be a choice from a pre-selected set, or generated autonomously using computer vision object recognition approaches. In either case, r serves as the Gaussian mean distance away from the centroid, with some standard deviation σ, such that for a given vertex v in a hypothetical sketch object h to be simulated the distance of the vertex from the

centroid is drawn from the distribution:

$$|\vec{\mathcal{X}}v| \sim \mathcal{N}(r, \sigma) \tag{5.21}$$

Where $|\vec{\mathcal{X}}v|$ denotes the magnitude of the vector from the centroid to $\mathcal{X}$ to the vertex v. The number of vertices N_v is distributed according to a shifted Poisson Distribution, with mean λ. The distribution is shifted uniformly upward by three, with support on $\mathcal{R} \in [3, \infty]$, ensuring that sketches are at a minimum triangles.

$$N_v \sim 3 + Pois(\lambda) \tag{5.22}$$

For a given N_v, the requirement that the angles of a convex polygon sum to 2π radians reveals a nominal angular distance between each point of $\hat{\Delta\theta} = \frac{2\pi}{N_v}$ radians. To capture the irregularity of a sketched polygon, the angular noise parameter ψ is used, such that $\hat{\Delta\theta}$ and ψ specify a Gaussian distribution on angular distance. For a vertex v, the angular distance to the next vertex is drawn as

$$\Delta\theta \sim \mathcal{N}(\hat{\Delta\theta}, \psi) \tag{5.23}$$

The process for drawing vertices from the distribution specified by the 5-tuple $\{\mathcal{X}, r, \sigma, \lambda, \psi\}$ is summarized in Algorithm 10.

Using PSEUD, sketches can be drawn from the distribution of sketches carrying the same parameterization, allowing high fidelity testing of each parameters effect on POMDP performance. These sketches, represented by their vertices can then be directly converted into softmax observation models using the techniques of [100], as described in 5.7.1. For example, both of the sketches shown in Figure 5.11 are drawn from a single parameterizied distribution, and yield similarly structured softmax models.

5.10.2 Monte Carlo Auto-labeling

Once a softmax observation model has been generated, whether by a real human or through a simulator such as PSEUD, the individual softmax classes can be mapped onto semantic labels

Algorithm 10 Parameterized Sketching Emulator Utilizing vertex Downsampling

Function: $PSEUD$
Input: Centroid $\mathcal{X}$, Radius r, Sd σ, Poisson Mean λ, Angular Noise ϕ
Vertex Set $\{V\} = \emptyset$
Magnitude List $mags = []$
Angle List $angles = []$
$N_v \sim 3 + Pois(\lambda)$
for $i \in N_v$ **do**
    $mags_i \sim \mathcal{N}(r, \sigma)$
    $angles_i \sim \mathcal{N}(\hat{\Delta\theta}, \psi)$
end for
angles = Normalize($angles$,2π)
$\theta = angles_0$ #Starting angle
for $i \in N_v$ **do**
    $V_i = \mathcal{X} + [mags_i cos(\theta), mags_i sin(\theta)]$
    $\Delta\theta = angles_i$
    $\theta += \Delta\theta$
end for
return $\{V\}$

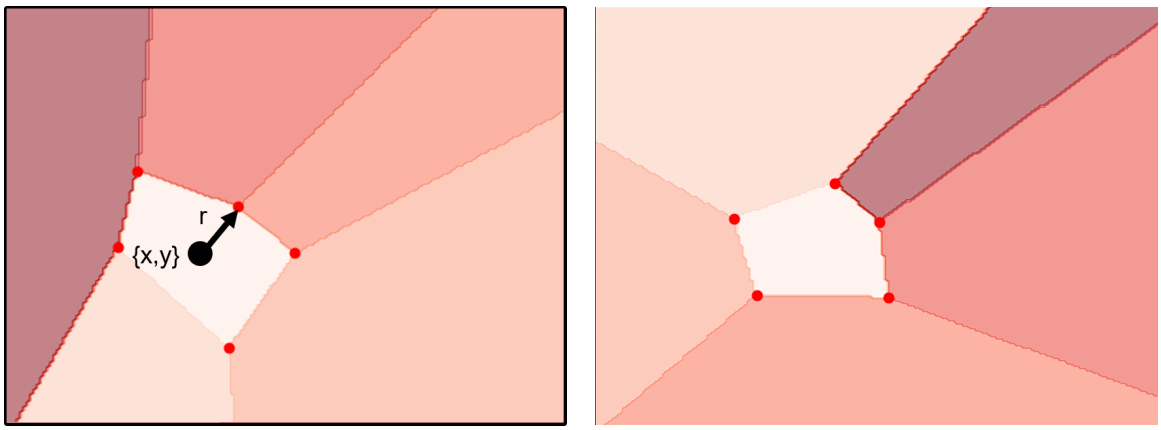


Figure 5.11: Two draws from the PSEUD Algorithm for a given centroid {x,y} and radius (r)

for further human interaction. In the Cops and Robots experiment of Chapter 4, this mapping was largely defined by hand offline to be 1-to-1, with a particular class label index always corresponding to 'South' for instance. This was enabled by assuming that the 2D spatial extent of all known semantic reference objects could be represented by simple Cartesian-aligned rectangles, so that softmax parameters for semantic classes describing 4 canonical bearings relative to the reference object needed to be specified. The use of free-form ad-hoc sketches leaves little room for such restrictions. For instance, in either of the sketches in Figure 5.11, it is unclear which of the classes

would receive the exclusive use of the label 'North', and indeed it seems unwieldy to attempt such a hard classification. This suggests that a soft classification approach could be used, disassociating the semantic labels from tight correspondence with individual classes.

To start this process, a new set of conditional probabilities are defined as the probability that a state drawn from softmax class (c) would be associated with the relational label (l), $p(label = l|class = c)$, and the probability that a given human observation o_h arising from a query action a_q corresponding to semantic label l was referring to the current state being associated with softmax class c, $p(class = c|a_q = l, o_h = Yes)$, or more succinctly $p(class = c|label = l)$. That is, given that the human indicated the state was associated with relational label (l), with what probability the true state would have been drawn from softmax class (c) The generative process used to build the POMCP planning tree will rely on $p(label = l|class = c)$, $p(l|c)$ for short, while updating the target belief distribution requires $p(class = c|label = l)$, $p(c|l)$. Both conditional distributions can be derived from the joint probability distribution $p(c, l)$. In order to find this distribution, first a canonical semantic bearing model L with overlapping 90 degree increments is assumed. Each semantic label, corresponding to the 8-point set of cardinal directions, covers an angular distance of 90 degrees, and overlaps by 45 degrees with the labels on either side. For instance, a point at $\frac{\pi}{4}$ radians or 45 degrees counterclockwise above the horizontal in this canonical model can be accurately labeled 'North', 'NorthEast', or 'East', while a point at 60 degrees counterclockwise above the horizontal can be labeled only 'North' or 'NorthEast'. Such a model can be represented as an overlapping piece-wise function on the angle θ made by state (s) with the horizontal, generating labels (l),

$$L(s) = \begin{cases} NorthEast & 0 \leq \theta \leq 90 \\ North & 45 \leq \theta \leq 135 \\ NorthWest & 90 \leq \theta \leq 180 \\ West & 135 \leq \theta \leq 225 \\ ... \end{cases} \tag{5.24}$$

Note, the overlapping nature of this function allows for the return of a set of labels, rather than a single label. This further suggests the possibility of a softmax representation, which will be explored in future work. Here the probabilistic representation $p(l \in L(s)|s)$ is used to describe the current deterministic function to preserve generality.

In order to find the joint probability $p(c, l)$, it is necessary to calculate the integral of the product of the class and label probability over the state space.

$$p(c, l) = \int_{s \in S} p(c|s)p(l \in L(s)|s)ds \tag{5.25}$$

While the label term $p(l \in L(s)|s)$ can only take values 0 or 1, as it describes a deterministic piece-wise function, the integral over the softmax function $p(c|s)$ is analytically intractable [6]. Therefore, a Monte Carlo approach is used to approximate the integral, selecting J states $s \in \mathcal{S}$ to carry out the summation:

$$p(c, l) = \frac{1}{J} \sum_{j=1}^{J} p(c|s_j)p(l \in L(s_j)|s_j) \tag{5.26}$$

From this joint distribution, both the conditional probabilities $p(c|l)$ and $p(l|c)$ can be easily obtained. The former is used within the Bayesian belief update to find $p(s|l)$,

$$p(s|l) = \sum_{c} p(s|c)p(c|l) \tag{5.27}$$

while the later is used to generate semantic labels from belief states during the planning phase through $p(l|s)$,

$$p(l|s) = \sum_{c} p(l|c)p(c|s) \tag{5.28}$$

Again, in this instance $p(c|l)$ corresponds to $p(class = c|a_q = l, o_h = Yes)$, such it serves as a probabilistic mapping from human information to a set of mathematical objects after a robotic query.

While this approximation approaches the true joint distribution with infinite samples, it is necessary in practice to choose a finite number of points. While these would typically be randomly scattered throughout the state space to avoid sampling bias, the number of points required to achieve an accurate approximation could be rather high. To limit computational expense, while preserving the approximation's accuracy, this work leverages the determinism and radial symmetry of the piece-wise canonical bearing model L by using a set of points uniformly distributed on θ at a constant radius. This method is illustrated in Figure 5.12, where the softmax model on the left is evaluated using the ring of Monte Carlo points on the right.

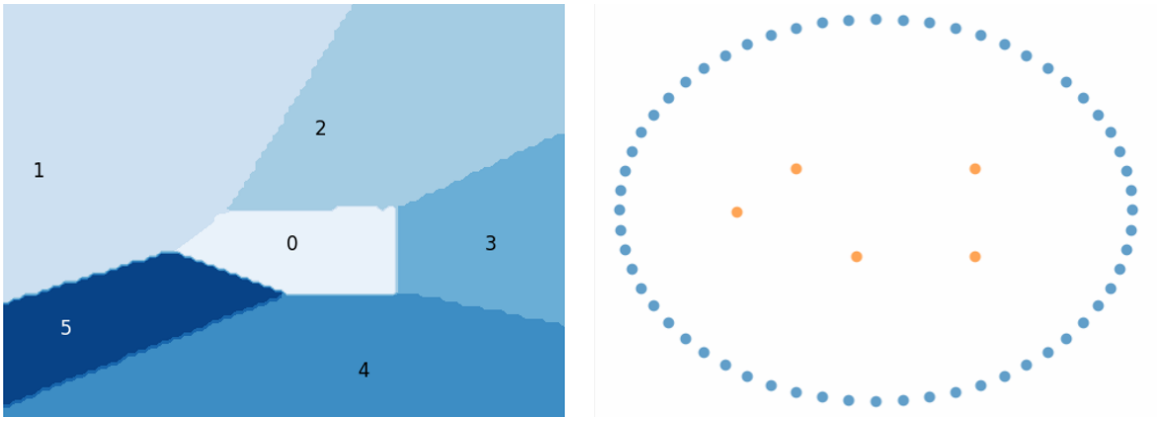


Figure 5.12: An irregular softmax model with numbered class labels (left)
The Monte Carlo approximation points used to approximate semantic class labels (right)

This approximation yields the conditional probability distributions shown in Figure 5.13. These distributions, substantially similar to those found using a dense random sampling method

with 10000 points, were found using only a sampled ring of 360 points. Furthermore, they correspond to intuition regarding the semantic labels. From model in Figure 5.12, $p(class|label)$ in Figure 5.13 indicates that an observation 'South' of the object in question could only reasonably be associated with class 4, while $p(label|class)$ indicates that states sampled from class 4 would likely be labeled either 'SouthWest', 'South', or 'SouthEast'.

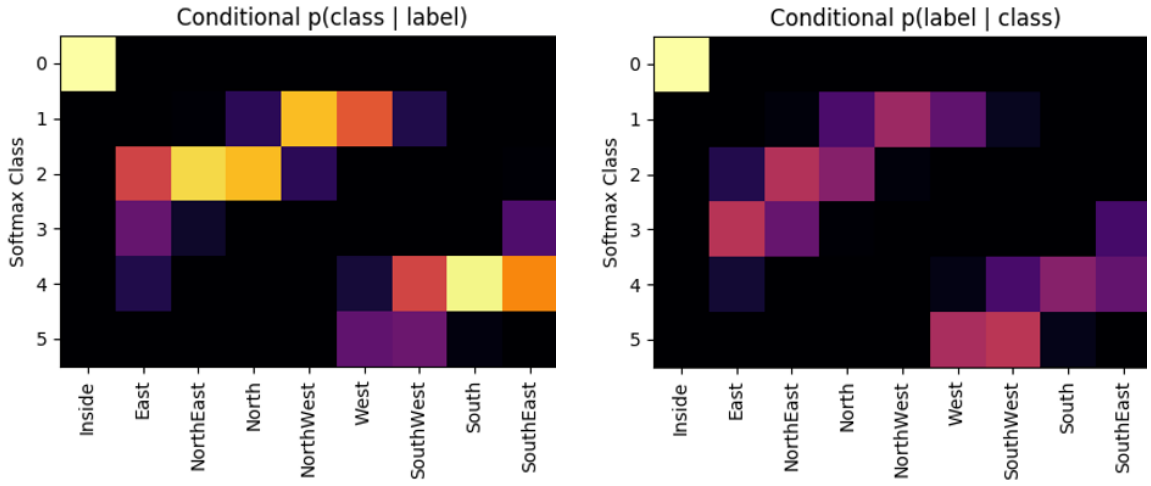


Figure 5.13: Conditional probability tables derived from Monte Carlo Auto-labeling

5.10.3 Composite Range Models

While the Monte Carlo Auto-labeling technique described in the previous section establishes a probabilistic correspondence between softmax classes and the type of semantic labels a human would use, the interior class based method of building softmax models used in [26, 27, 25] is limited in its ability to express range information. The target can be described as "Inside" the sketch if it is literally within the bounds of the sketched convex polygon, or as "Not Inside", in which case it can be anywhere else in the state space. In many applications, including the motivating problem for this chapter, it is desirable to express when the target is in the general vicinity of the sketch, when it is "Near" or "Near NorthEast" to a particular landmark. Previous work [6] constructed such range-bearing softmax models using hand crafted classes for specific range-bearing combinations ("Near Northeast"), and used an multimodal softmax (MMS) model summing together a set of

classes to express range only measurements ("Near").

In this section, an alternative method is proposed for constructing range labels which is more applicable to ad-hoc sketch-based softmax models. Here, the range softmax model is constructed based on the structure of the sketch before being combined with the sketched bearing model in a product softmax model [100]. Specifically, this section and the following text make use of a single label "Near" range model, rather than the multiple option range models of previous work.

The "Near" class is first assumed to hold an identical, yet inflated, shape to the original interior softmax class, which has conditional probability $p(Inside|s)$. That is to say, the vertices of the near model are an affine transform of the original vertices such that the area they encompass is a scalar multiple h of the previous area. The interior class of these inflated points can be applied as a product softmax model as shown in 5.14, under the assumption that range and bearing are independent, such that

$$p(Near, Bearing|s) = p(Near|s)p(Bearing|s) \tag{5.29}$$

Thus, a point can be evaluated against both models separately to determine the probability it was drawn from the composite joint range-bearing class label. This work only considers the use of the "Near" range observation, with observations outside the range models interior class carrying no explicit range information. In such a framework, the negative observation "Not Near" serves as a proxy for the implicit observation "Far". In principle, rather than being restricted to binary range information, additional affine transforms of larger size can be applied to indicate any number of semantic ranges. However, such models would necessarily overlap and fully contain the smaller "Near" interior class, such that for some hypothetical "Near/Far" range model it must be assumed that any observation "Far" is actually the compound observation "Far and Not Near":

$$p(Far,\ Near|s) = p(Far|s)(1 - p(Near|s)) \tag{5.30}$$

In this way, Equation 5.27 can be generalized, with the assumption of range-bearing independence intact, to:

$$p(Range, Bearing|s) = p(Range|s)p(Bearing|s) \tag{5.31}$$

A cleaner approach to be explored in future work would be the use of softmax models specified directly in range space, such that "Near" and "Far" could correspond to 2 different classes within the same function rather than 2 separate functions. However, this approach would neglect the shape information from the sketch preserved in the affine transform method, a trade-off to be further explored. The example problems in the following sections are restricted to the "Near" only range models.

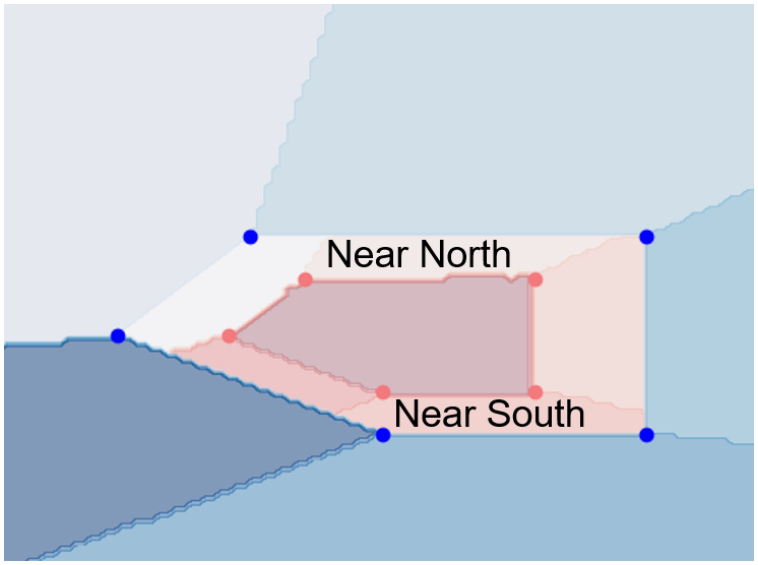


Figure 5.14: Composite Range and Bearing Softmax Model

5.10.4 Observation Model Generation Pipeline

The techniques developed for PSEUD, Monte-Carlo Auto-Labeling, and Near Class Composite Modeling can combined into a single end-to-end pipeline capable of generating useful ad-hoc observation models which mimic those a human might produce, as shown in Figure 5.15. When a real human is introduced, Monte Carlo Auto-labeling and Near Class Composite Modeling still

remain applicable.

Figure 5.15: Sketch Generation and Preparation Pipeline

5.11 Concept Simulation Results

This section presents the results of two simulated scenarios depicted in Fig. 5.2 in which the robot attempts to localize and intercept a moving ground target. In the first 'Golf Course Problem', an unmanned ground vehicle (UGV) pursues the target in an unmapped area with unknown terrain features for which the state transition dynamics are unknown a priori. In the second scenario, an unmanned aerial system (UAS) pursues a ground based target in a large road network; the target moves according to a two-layer HMM as shown in Fig. 5.3 (right), where m indicates whether the target is road-bound.

5.11.1 Golf Course Problem: Search in Unknown Environment

The golf course shown in Fig. 5.2 is the underlying environment. The course contains numerous regions which either slow or speed movement through them by ground vehicles. We model these differences through the use of a modifier variable α applied to the nominal transition model. The UGV is initialized with a uniform value of α across the state space. The UGV's human teammate, for cases wherein human assistance is provided, is able to modify the UGV's understood value of α through sketches. Here we assume the human's provided modifiers are perfectly accurate even if their observations about their sketches aren't necessarily. Expected transitions for the UGV $E[s']$, with $\Delta(a)$ being the expected move resulting from an action, are given as $E[s'] = s + \alpha\Delta(a)$,

$\alpha = [Water : 0.05, Sand : 0.5, Fairway : 1.5, Green : 2, otherwise : 1]$.

$$p(s'|s,a) = \phi(s'|s + \alpha\Delta(a), \Sigma_a) \tag{5.32}$$

Where the noise associated with any action Σ_a is uniform across all actions and terrain modifiers, and is set to $\Sigma_a = 1$ for this example. The UGV has a nominal speed of 5 m/s in any of the 4 cardinal directions, while the target nominally moves at 2 m/s using a nearly constant velocity (NCV) kinematic state model [12]. The reward model is given as a piecewise function where having the target within $\tau = 7.5m$ results in a high reward, and asking the human a question results in a slight cost.

$$\mathcal{R} = \begin{cases} 100 & dist(s_t, s_r) \leq \tau \\ 0 & dist(s_t, s_r) > \tau, a_q = Null \\ -1 & dist(s_t, s_r) > \tau, a_q \neq Null \end{cases} \tag{5.33}$$

Observations made by the robot's on-board proximity sensor have three possible values, either indicating that the target is "Far", "Near", or "Captured", all with true positive rates of 98%. The robot's field of view, and thus the "Near" observation, extend out to $2\tau = 15m$ while distances beyond are tagged "Far". Capture is declared at a threshold distance $\tau = 7.5m$.

$$\mathcal{O} = \{Captured, Near, Far\} \tag{5.34}$$

$$\Omega(Captured) = p(o = Captured|s) = .98, \forall s|dist(s_t, s_r) \leq \tau \tag{5.35}$$

$$\Omega(Near) = p(o = Near|s) = .98, \forall s|\tau \leq dist(s_t, s_r) \leq 2\tau \tag{5.36}$$

$$\Omega(Far) = p(o = Far|s) = .98, \forall s|2\tau \leq dist(s_t, s_r) \tag{5.37}$$

In the "Human" test case, a non-simulated human was used to provide a set of labeled sketches prior to testing. Sketches were drawn from a list of 15 pre-drawn options, focusing on areas of interest such as water, sand, and greens. For each of the 100 simulated runs, the human provided a sketch at the beginning of the run, and added an additional sketch every 20 time steps

Golf Course Problem Results

Method	Mean TTC	Standard Deviation
Non-human	34.16	±23.97
Non-human Informed	31.22	±22.04
Human	25.46	±15.78

Table 5.1: Golf Problem Comparison of simulation TTC

up until the end of the run at 100 time steps. Each time step corresponded to 2 seconds, which was the planning time allocated to the POMCP solver. Using the observation model described in Section III-a, the simulated humans were assumed to have an accuracy and responsiveness of $\eta = \xi$ =95%. The queries made by the robot were constructed in binary fashion similarly to those in [27, 25]: questions a_q took the form of queries about the target position relative to cardinal directions for each sketched map landmark, e.g. "Is the target east of Fairway 1", with potential replies $o_h \in [Yes, No, None]$, where $None$ occurs with probability $1 - \xi$, and the correct Yes or No reply occurs with probability η.

The UGV gains a reward of 100 for capturing the target, which requires being within the capture radius τ, and incurs a small cost of -1 for asking questions of the human. This reward function encourages fast target capture. Since POMDP policies maximize the cumulative reward utility resulting from $R(s, a)$ and thus seek to achieve a real-world goal encoded by the resulting utility, we report the results of our simulations in terms of our goal metric, Time to Capture (TTC), which is the first timestep the UGV is within 7.5 meters of the target.

Three deployment cases are evaluated to determine the effect of introducing human sketches and observations. In the first "Non-human" case, the UGV is given a uniform α modifier over the map. The "Human" case gives the UGV a human, capable of providing both sketches and observations in response to requests from the UGV. The final case, "Non-human Informed", gives the UGV the true α values across the state space. The TTC results are shown in Table 5.1, with the Human case showing significantly less time to capture than either the Non-human and Non-human Informed cases at $p < 0.05$ via Student's t-tests.

Road Network Problem Results

Method	Mean TTC	Standard Deviation
Non-human	39.43	±23.74
Human	30.45	±23.28

Table 5.2: Road Network Comparison of TTC

In order to examine the effects of differing levels of human accuracy η and responsiveness ξ, each variable was tested with select values ranging from 30% to 99%. The number of questions asked as a percentage of all actions, as well as the TTC metric for each test, are shown in Figure 5.17. η and ξ here are known and static for the robot. Additional research, possibly extending the HOP-POMDP framework [77] or [66] could allow the robot to learn these online.

5.11.2 Road Network Problem: Search with Switching Dynamics

In this scenario, the target moves through the search area in one of two modes, $m_0 =$ *on-road* and $m_1 =$ *off-road*. An on-road target has a chance to transition to off-road, and will follow an NCV model until it encounters another road. Thus the target dynamics in the continuous state space s are conditionally constrained with respect to the mode m. When available, the human is given a binary query asking "Is the target on the road", which yields response $o_h \in [Yes, No, None]$ in accordance with η and ξ. The reward, robot observation models, and human responsiveness and accuracy statistics are otherwise identical to the 'Golf Course' scenario presented above. Transitions are modeled with a uniform $\alpha = 1$ across the space.

The road-network scenario was tested in cases with and without a human collaborator answering questions pertaining to the target mode. As in the 'Golf Course' problem, the true simulated human's accuracy and responsivity are $\eta = \xi = 95\%$. The TTC results are displayed in Table 5.2. The Human case significantly outperforms the Non-human case ($p < 0.05$) via a Student's t-test.

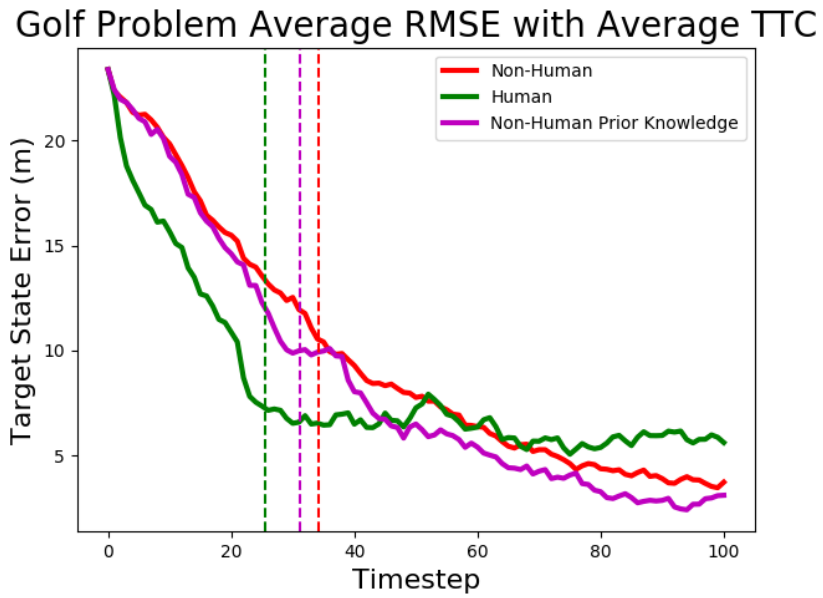


Figure 5.16: Average Root Mean Squared Error for Golf Course problem (vertical lines: average TTC per method).

5.11.3 Discussion of Results

In the Golf Course problem, the human-robot team was able to capture the target faster on average than the robot alone, ($p < 0.05$). A major factor in their success was the ability of the simulated human to rapidly improve the robot's estimate of the target's location, as shown in Figure 5.16. A better estimate leads to more effective querying and planning, with in turn produce better estimates and faster capture times.

When testing differing η and ξ values in Fig. 5.17, the most accurate and responsive humans were asked the fewest questions and achieved the best average TTC. Similarly, the simulated humans with lower η and ξ received more questions. The robots knowledge of η and ξ indicates it adjusts the frequency of questions to obtain a similar quality of target state estimate regardless of human. Furthermore, these tests support our claim that a human-robot team generally outperforms a lone robot, despite introducing increased complexity. In addition to simply providing more information than the robot otherwise has access to, the human can be further interacted with in a purposeful and active manner. As the robot takes movement actions to not only localize the target but

also intercept it, also actively makes queries to the human in support of the same goal. In 89% of these tests, the TTC was lower than that achieved in the Non-human case from Table 5.1, and the combinations of η and ξ which did not surpass the 'Non-human' case still carried similar performance. This suggests the 'plug-and-play' nature of our framework allows it to make the best of a poorly suited human as well as take advantage of a highly useful one, and in the worst case see comparable performance with a more complex problem.

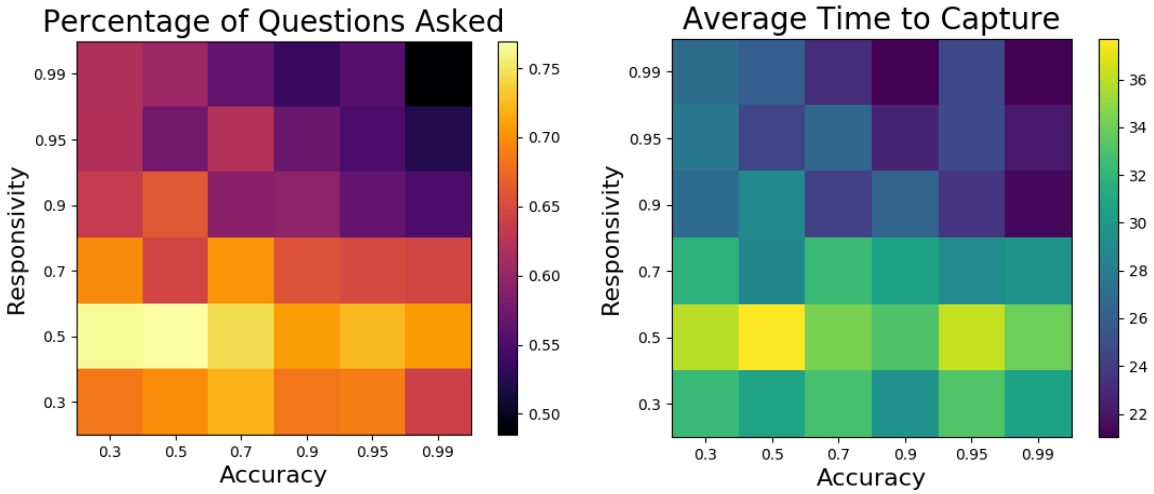


Figure 5.17: Given humans with various levels of accuracy and responsivity, Left: The number of questions asked by the robot as a percentage of all actions; Right: The average TTC

In the Road Network problem, the human was able to positively impact the robot's search even without the ability to provide semantic data about the target's continuous state. As indicated in Table 5.2, the option to query a human about the current mode of the target, i.e. whether or not it was bound to the road-based dynamics model, resulted in significantly lower TTC, ($p < 0.05$). This is demonstrated by Fig. 5.18, where the robot's estimate of the target's mode is clearly superior when the human is available. The robot is able to periodically constrain its estimate of m by querying the human, and maintain a better understanding of the target's future movements. Thus, the use of a human to measure an indirect variable such as the target's mode of travel allows the robot to more effectively carry out its task.

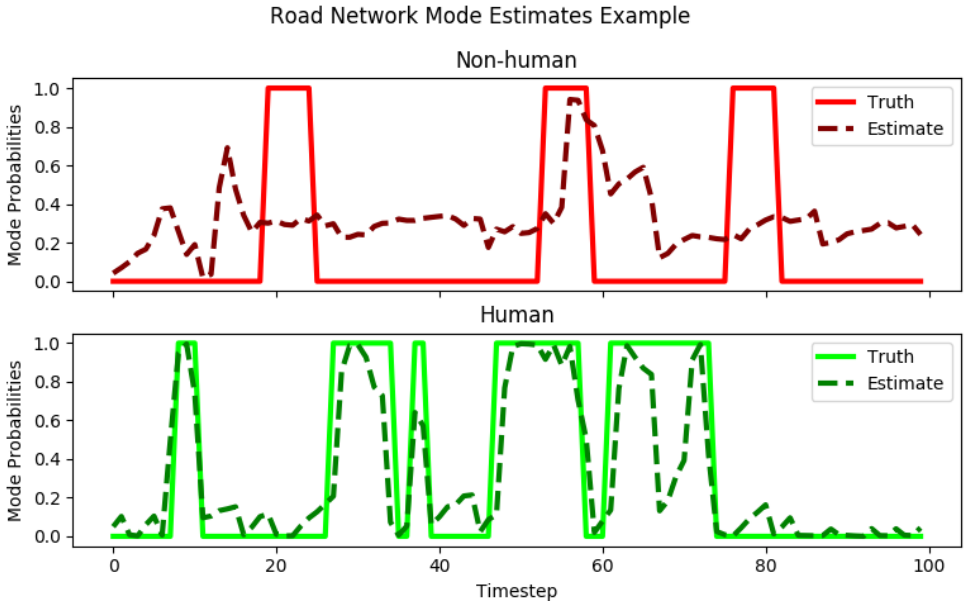


Figure 5.18: Example of robot's ability to estimate whether target is bound to road network, with and without human input.

5.12 Simulated HARPS Results

The Human-Assisted Robotic Planning and Sensing (HARPS) environment was developed in Unreal Engine 4 [37] to provide a physics based simulation on which to test POMDP tracking problems with human input. A drone, controlled through ROS [97] using the Microsoft Airsim API [87] attempts to localize, track, and intercept a ground based target within an outdoor environment, while interacting with the human through the sketch based interface shown in Figure 5.19. This PYQT5 based interface inherits robot pull and human push functionality from the CNR interface in Chapter 4, but populates the semantic dictionary within from user sketches drawn directly on the map. Also similar to CNR, the human can view the space either through a selection of security cameras or through the robot's on-board camera.

In order to assess the ability an online POMDP to work together with a human to complete this task, a simulated human with a variety of parameters generating sketches and observations using the methods of Section 5.10 was first tested on a slightly simplified version of the HARPS environment built in Python 3.5. This allowed for large batch testing to examine the POMDP's

Figure 5.19: HARPS Human Sketch Interface

effectiveness and potential weak points prior to live human subject testing.

In all simulated run, the drone was given $t_{max} = 600s$ maximum flight time, after which the run was considered a failure. Rather than pose actions as simple directional objects ("North" and "South") as in previous experiments, HARPS used a graph-based POMDP navigation approach, which designated intersections, or nodes, of the local road network as actions. In principle, this is no different than designating 4 cardinal direction displacements as actions, and thus requires minimal changes to the planner. Specifically, the POMDP was allowed to choose any neighboring node, or neighbors of neighbors, from its previous action:

$$A_m^t = \cup n_{t+1} \in Neighbors(Neighbors(n^t)) \tag{5.38}$$

The use of such an action model opens the door to variable time actions, and the use of Predictive Tree Planning techniques discussed in Section 5.8. The robot was allowed to move at an average speed of $15\frac{m}{s}$, with $\Sigma_a^r = 1$ for all actions, while the target followed the switching Markov dynamics model in the road network problem of Section 5.12.2, using an average speed of $20\frac{m}{s}, \Sigma_a^t = 5$ while on the road and $5\frac{m}{s}, \Sigma_a^t = 1$ while off-road.

Query actions A_q were handled identically to those in 5.11.1, with the exception that instead

of being selected from a pre-drawn set all sketches were generated in real-time using the PSUED algorithm.

Rather than the circular detection assumption of Sections 5.11.1 and 5.12.2, HARPS uses a conical model with a 30 degree viewcone v_c for the robot similar to that used in Chapter 4, with observation set:

$$\mathcal{O} = \{None, Detected, Captured\} \tag{5.39}$$

and observation likelihood model with a 98% accuracy:

$$\Omega(Captured) = p(o = Captured|s) = .98, \forall s|dist(s_t, s_r) \leq \tau \& s \in v_c \tag{5.40}$$

$$\Omega(Detected) = p(o = Detected|s) = .98, \forall s|\tau \leq dist(s_t, s_r) \leq 2\tau \& s \in v_c \tag{5.41}$$

$$\Omega(None) = p(o = None|s) = .98, \forall s|2\tau \leq dist(s_t, s_r) \& s \in v_c \tag{5.42}$$

As HARPS takes place on a larger scale environment, with state space $\mathcal{S}$ comprising a 1000m x 1000m area, the capture distance threshold τ is set to 75m. However, the restriction of detection and capture to the area under observation by viewcone v_c limits the ability of the robot to detect targets it is not directly facing as a result of movement actions A_m.

The reward function was identical to that used in Section 5.11.1 Golf Course problem, in that the robot was given a large positive reward of 100 for the capture of the target within $\tau = 75m$, and a small penalty of -1 for asking the human a question.

All statistical testing in the simplified HARPS environment was conducted using binomial significance tests on the ratio of successful captures by a given method when compared to a control.

5.12.1 Human vs Nonhuman

Figure 5.20 compares the ability of the drone to intercept the target with and without human assistance. Each case was tested with N=250 independent trials, with significance being determined via Binomial Test. It is particularly important to note that the POMDP is able to locate and intercept the target in the majority of cases even without human intervention. With or without

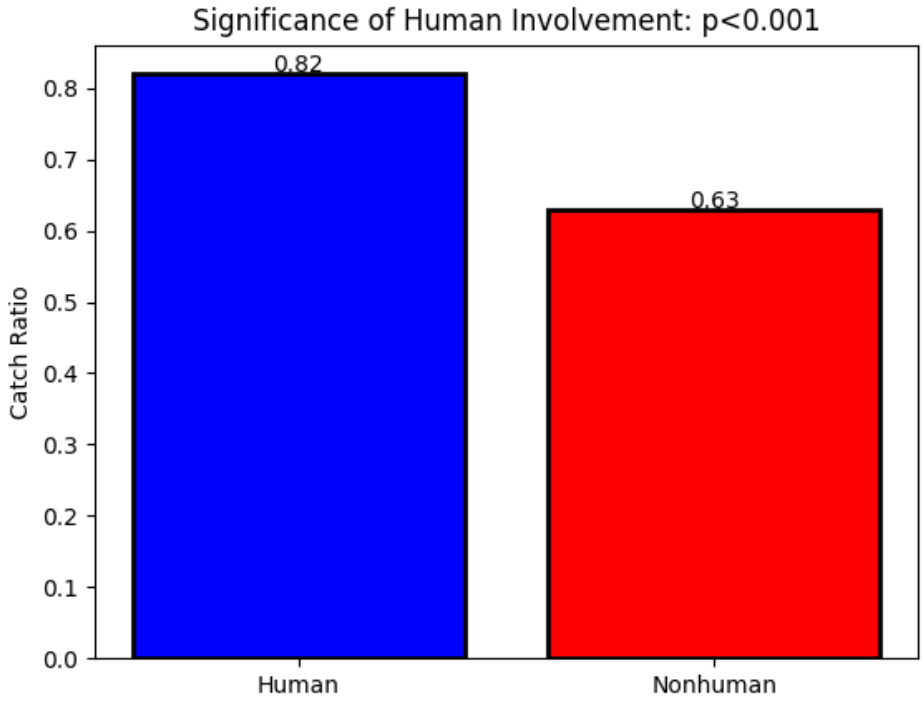


Figure 5.20: Effect of Human Input in Simulated HARPS Environment

the human's information, the online POMDP is still approximating an optimal policy for the information it has available. Far from minimizing the improvement shown in the "Human" case, the significant ($p < 0.05$) discrepancy with the "Nonhuman" case emphasizes that even an otherwise effective algorithm can still benefit greatly from the introduction of additional human information.

Qualitatively, the introduction of human information resulted in similar behavioral changes as in Chapter 4. For example, in Figure 5.21, without human input, either sketches or semantic observations, the robot engages in a broad ranging patrol behavior, prioritizing well connected nodes in the graph and eventually capturing the target as it ventured through a high traffic area. However, the human case of Figure 5.21 shows a more directed pursuit behavior, wherein the robot moves quickly towards the target area indicated by the human before firmly localizing and capturing the target. Note, this example does not imply all human cases were so direct, nor all non-human cases so wandering. Rather, it represents general behavior of each approach.

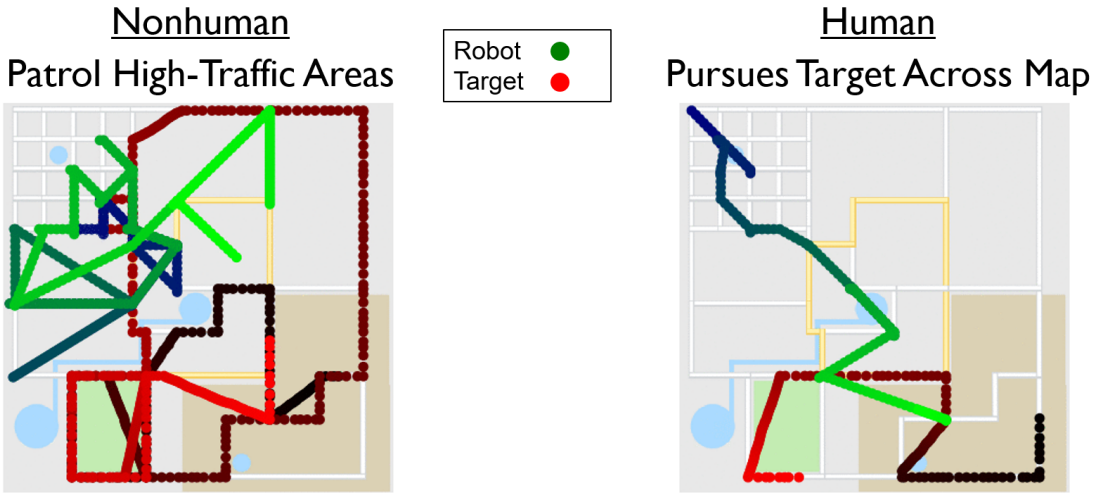


Figure 5.21: Representative example traces of the robot (green) and target (red) showcasing Left: Patrol Behavior of Non-Human HARPS Case, and Right: Pursuit Behavior with Human Information

5.12.2 Predictive Tree Search vs Blind Dyanmics Updates

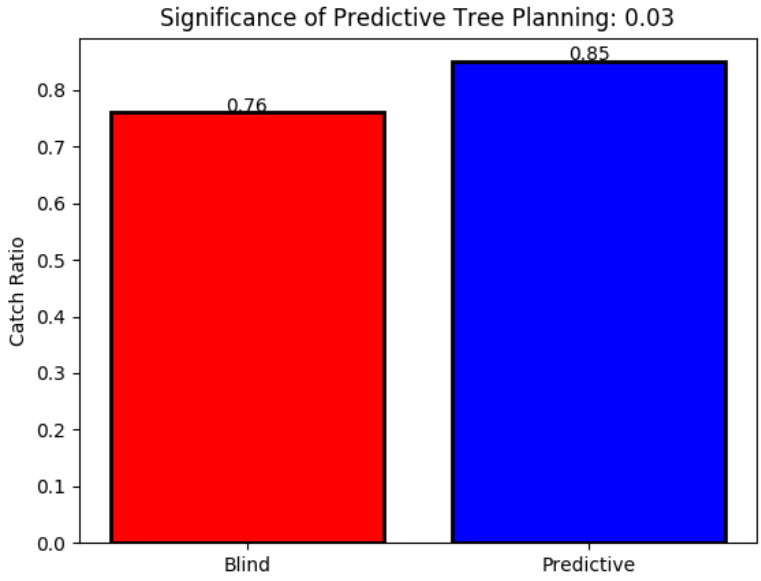


Figure 5.22: Effect of Predictive Tree Search Input in Simulated HARPS Environment

In Figure 5.22, the Predictive Tree Search method is tested against a naive "blind" real-time

planner. Each case was tested with N=100 independent trails, with significance being determined via Binomial Test. In the blind case, planning is also carried out during the execution of the previous action, but using a belief predicated only on the expected dynamics update. This is opposed to the Predictive method which probabilistically allocates planning time among multiple possible observations to produce predicted measurement updates in addition to the dynamics update. Of note, the blindness of the blind method only applies to a single step, as the full measurement update still occurs after the end of the current action, so it is at most one step behind. Here the Predictive Tree Search approach produces significantly ($p < 0.05$) more captures by drones operating with a battery dictated 10 minute search restriction. This may be attributable to the fact that the drone moves at a slower speed than the target on the road, and therefore timely arrival of a single positive measurement can have an out-sized effect on the ability of the drone to capture the target before it escapes the local area.

5.12.3 Human Accuracy

After establishing the general effect of a human collaborator on the effectiveness of the POMDP, specific attributes likely to vary between live subjects were examined. First, human accuracy was tested, using values in the set: [.3,.5,.7,.9,.95]. That is, for a value of .7, the human would respond to questions correctly 70% of the time, and incorrectly 30% of the time. These same values were tested as applied to the robot's model of the human, allowing the difference between the true and assumed sensor dynamics to be examined. The full results of these tests, carried out for N=50 independent trails for each combination of assumed and actual accuracy, are show in Figure 5.23. In particular, the results with regard to matched assumed and actual accuracy are displayed in Figure 5.24 alongside the Binomial significance test results for each pair-wise comparison. Similar to work shown above for the Golf Course problem, a human operating at 50% accuracy acts as a random observation generator, and therefore does not produce significantly more captures than the earlier "Nonhuman" case. Also similar to the Golf Course problem, the 30% case allows the human observations to be taken in negative form, still allowing useful information to be passed

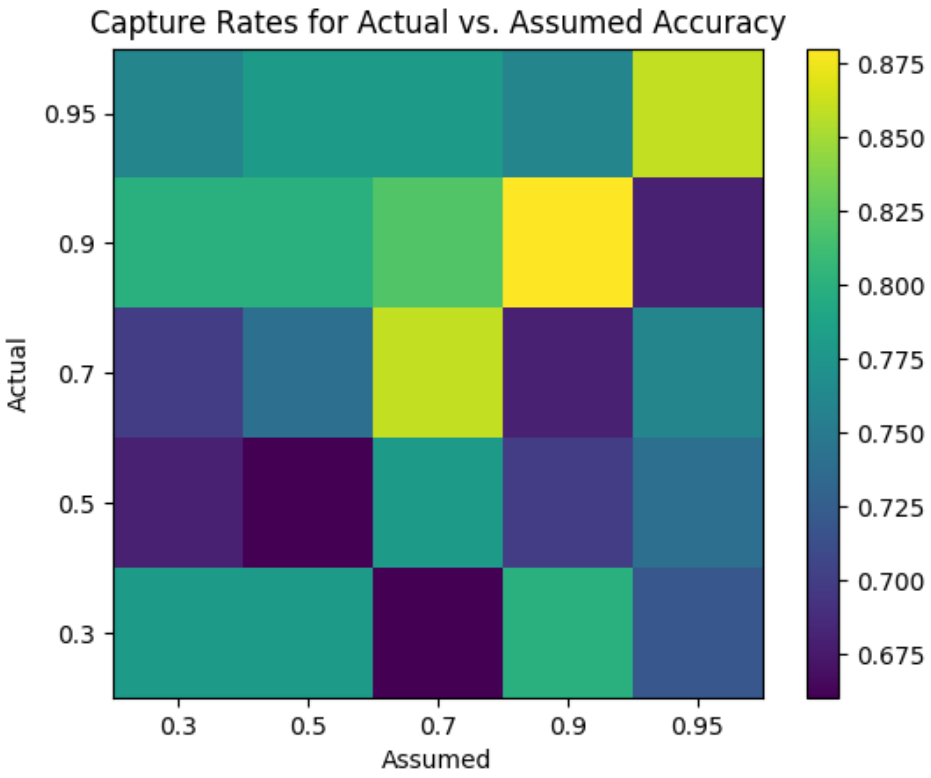


Figure 5.23: Effect of True Human Accuracy vs Modeled Accuracy in Simulated HARPS Environment.

along.

The average effects of both assumed and actual accuracy are displayed in Figure 5.25. The average true accuracy shows similar results to the match case, with the averaging over assumptions compressing the range of results, while the assumed accuracy displays no clearly significant effect. Naturally there can be limits to such data in marginalized form, as examining the combined effects in Figure 5.23 seems to indicate a slight advantage to a more pessimistic view of the human. That is, the upper-left triangle of results in which the human is more accurate than assumed performs slightly better on average than the lower-right.

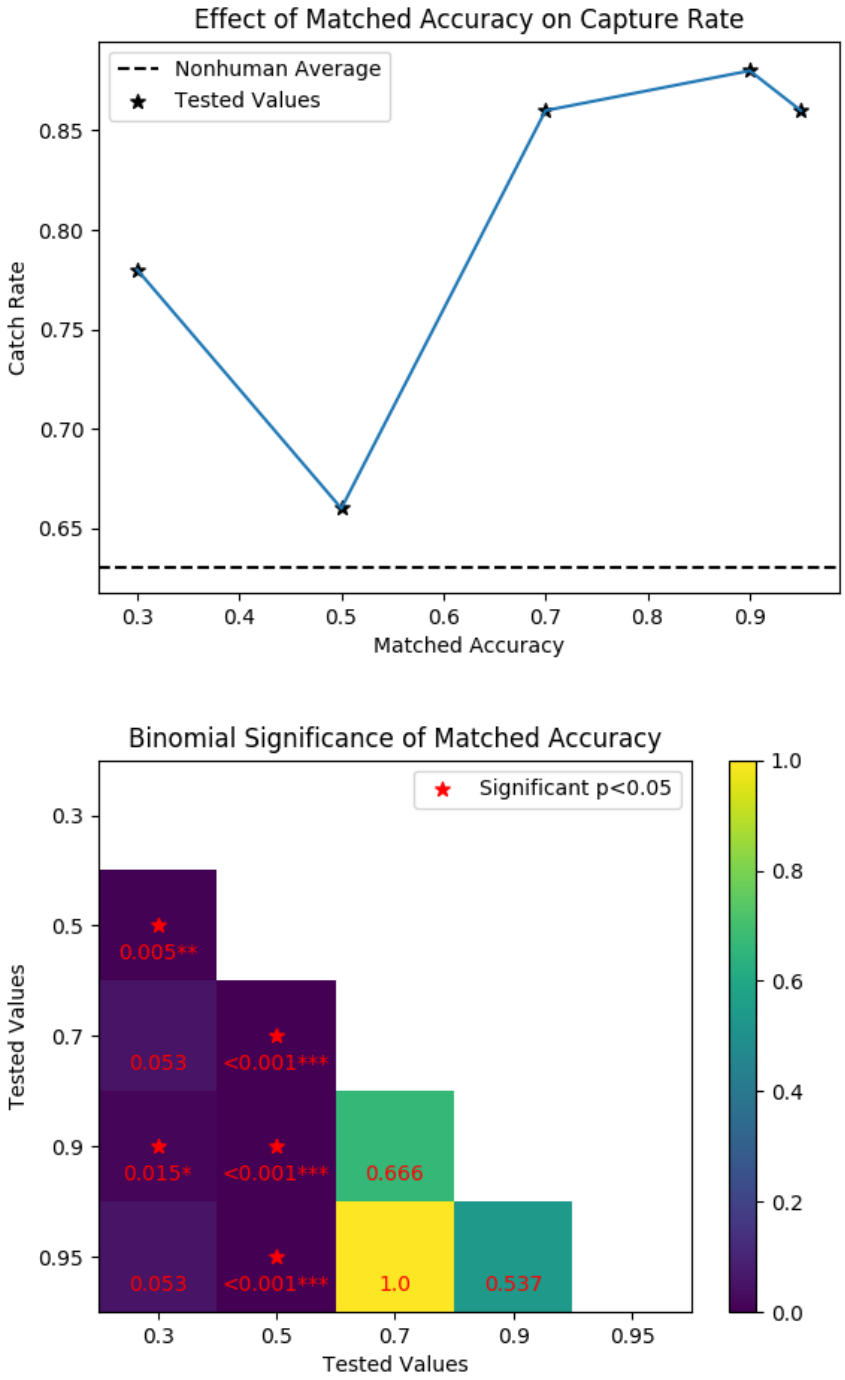


Figure 5.24: Effect of Matched Human Accuracy in Simulated HARPS Environment

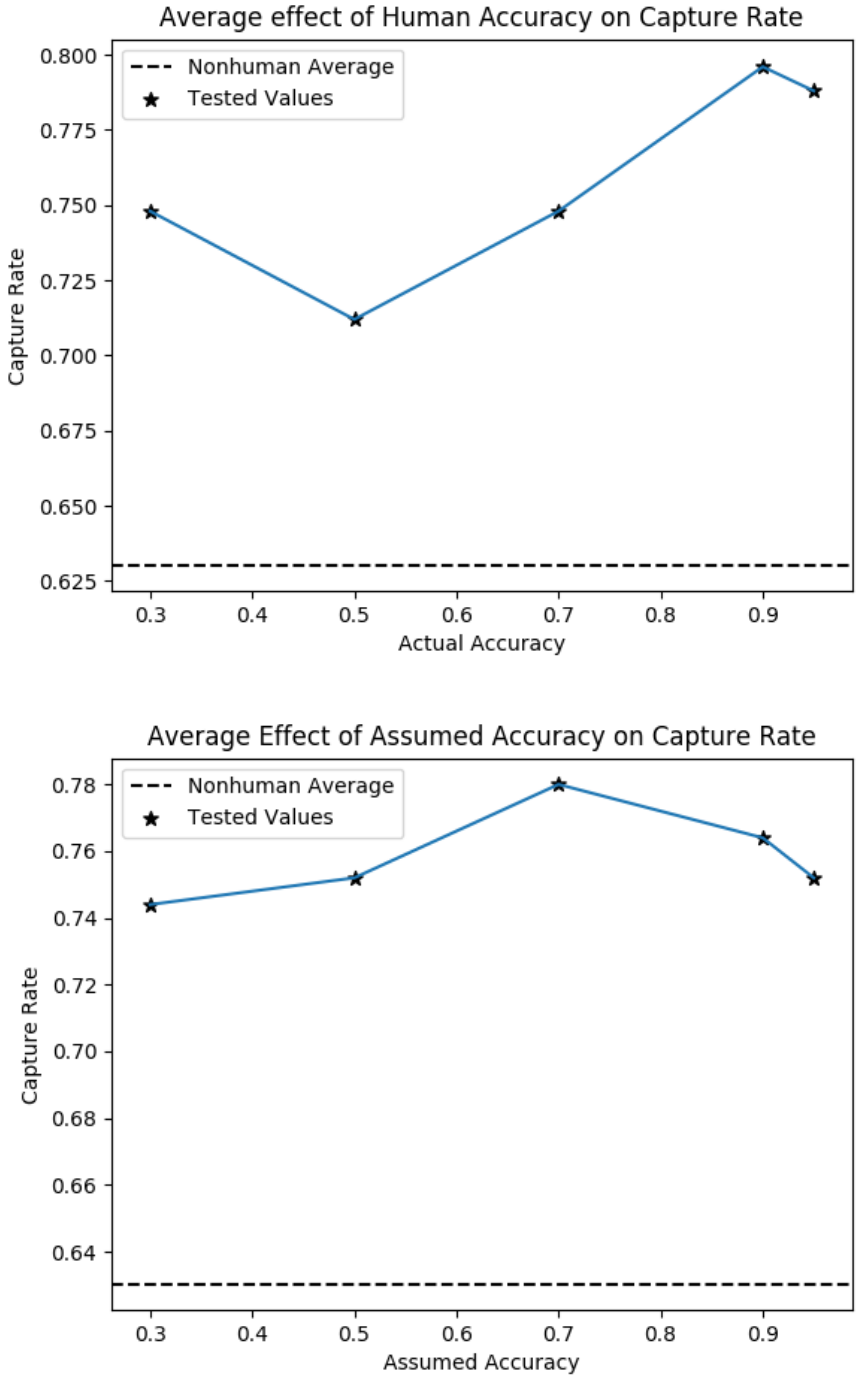


Figure 5.25: Average effect of Actual Human Accuracy (left) and Assumed Human Accuracy (right)

5.12.4 Human Availability

Similar to accuracy, human availability was also tested across the same range of assumed and true values, with the results aggregated in Figure 5.26. Given the same size of N=50, the

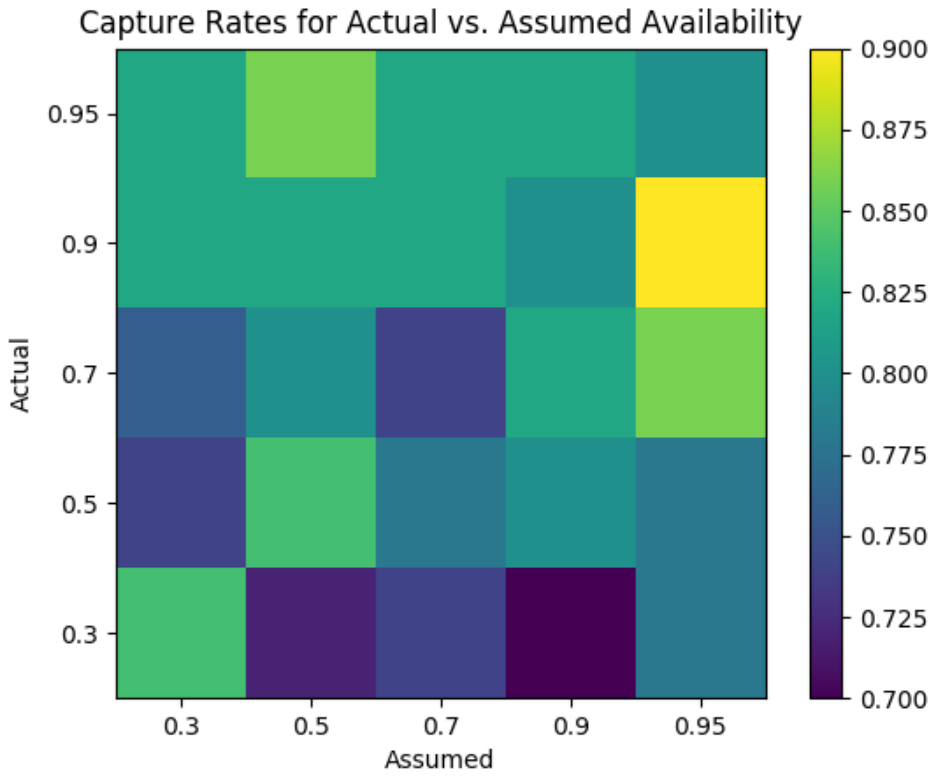


Figure 5.26: Effect of True Human Availability vs Modeled Availability in Simulated HARPS Environment

matched true to assumed results in Figure 5.27 do not indicate a strong effect. However, examining the marginal result distributions in Figure 5.28, there is a clear positive trend line for true human accuracy, while the effect of the robot's assumption of the human's accuracy is minimal.

5.12.5 Sketch Rate

Finally, the rate at which a human inserts additional sketch entries into the robot's semantic dictionary was examined. Simulated humans drew sketches at a constant rate drawn from: [15,30,60,120] seconds. For completeness, a trial was run in which the human was present but drew no sketches and introduced no new semantic information. As expected, this case of an infinitely long average sketching rate performed identically to the Nonhuman baseline. As shown in Figure 5.29, there exists an optimal rate at which sketches provide a sufficiently diverse semantic dictio-

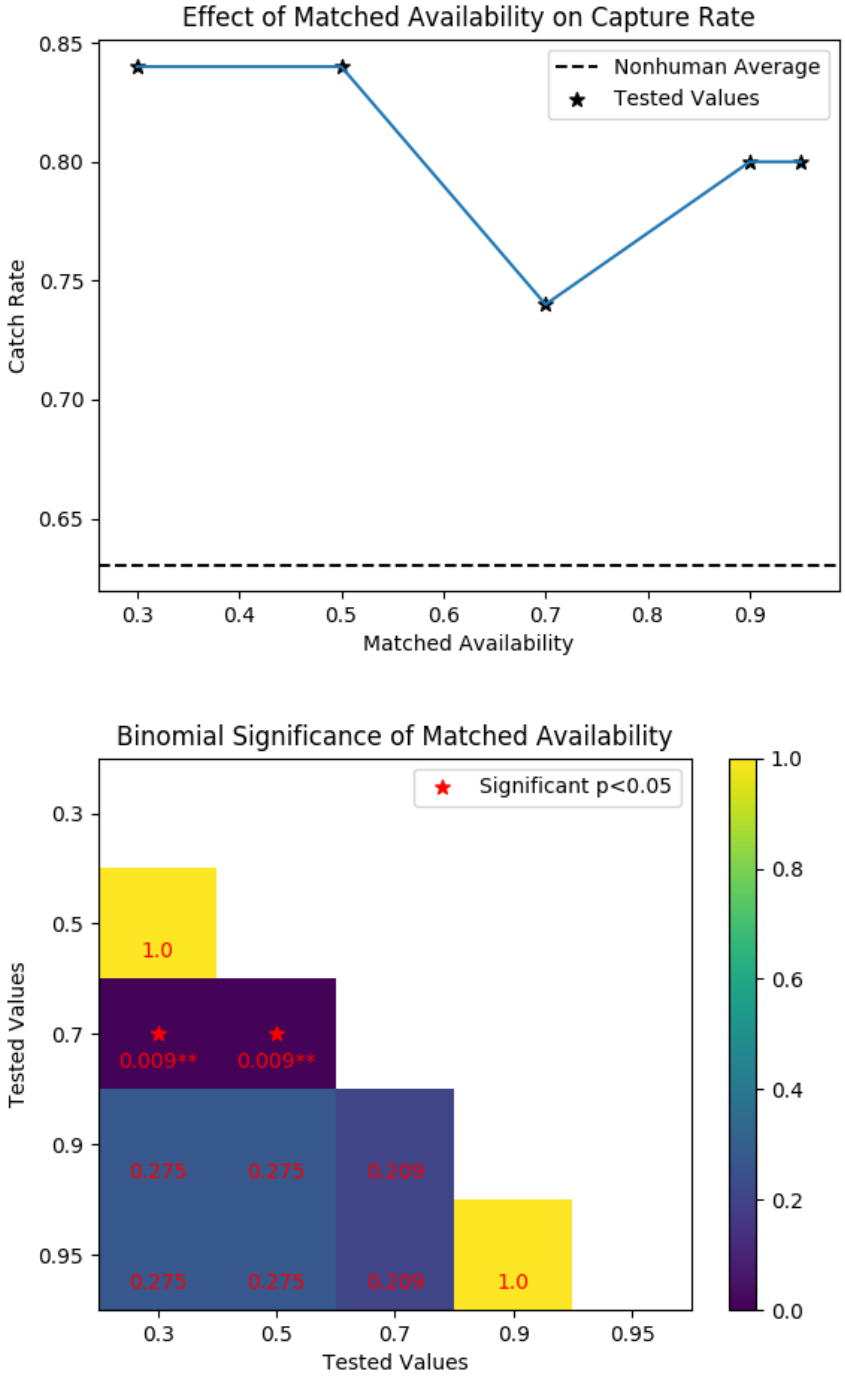


Figure 5.27: Effect of Matched Human Availability in Simulated HARPS Environment

nary without over-complexifying the planning problem with additional actions to consider. This optimum, which peaked around one sketch per minute in this scenario, is likely highly problem

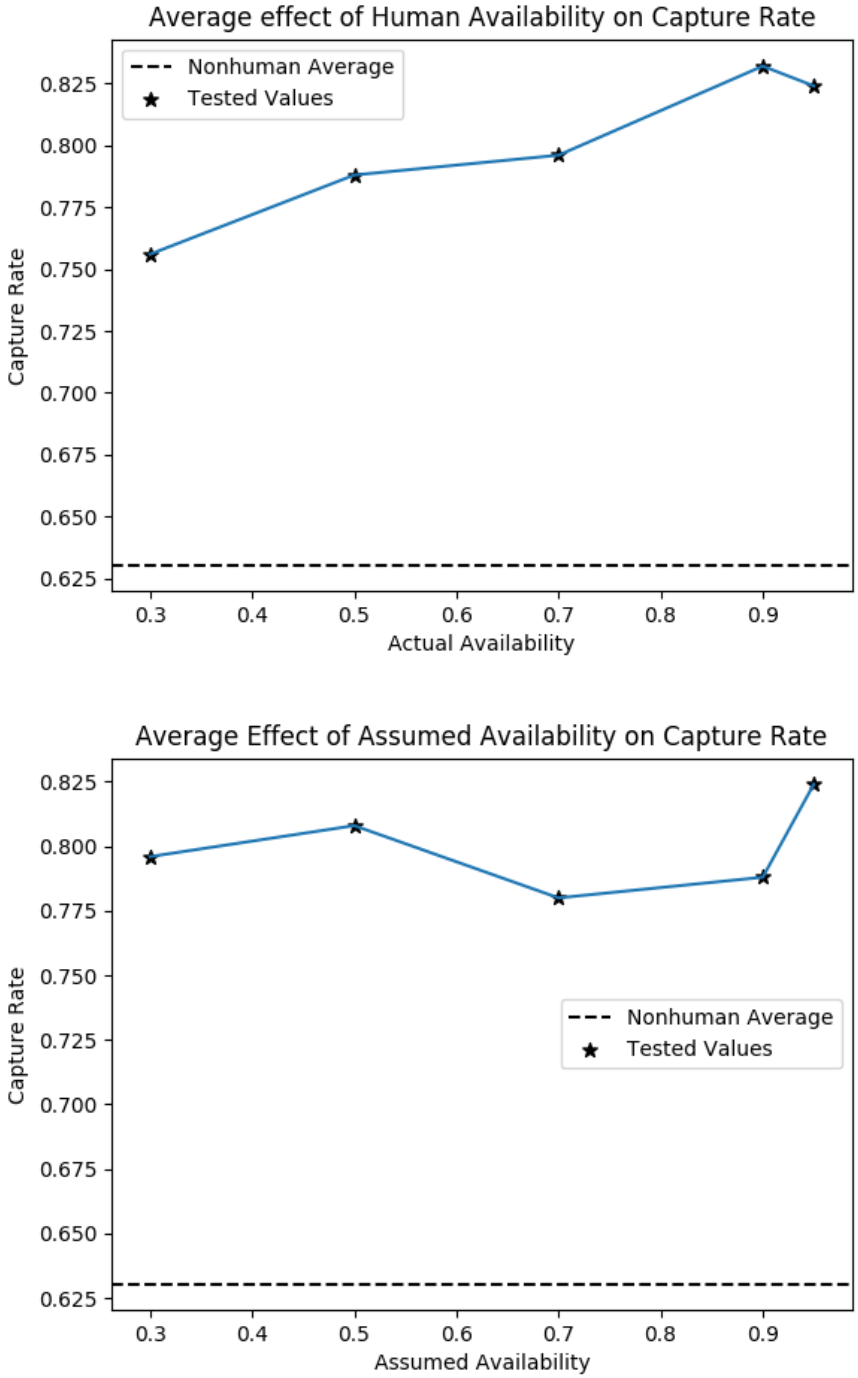


Figure 5.28: Average effect of Actual Human Availability (left) and Assumed Human Availability (right)

dependent and represents a consideration for future implementations of this work. It is possible that, having established for a particular problem the near-optimal sketch rate, human collaborators

could be coached or trained to adhere to it.

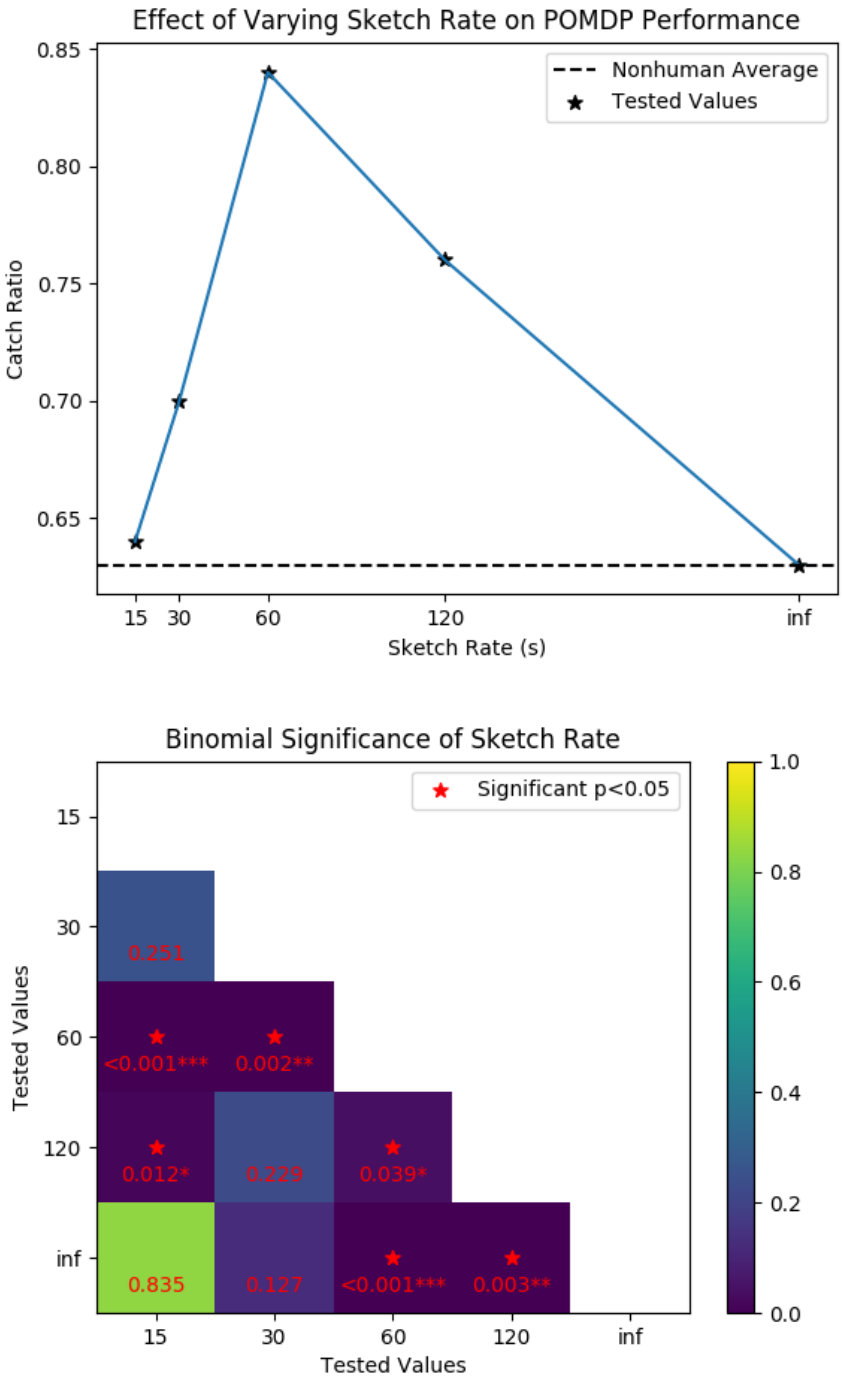


Figure 5.29: Effect of Sketch Rate in Simulated HARPS Environment

These results taken together showcase the ability of the online POMDP planner to adapt

to the human it is given, rather than requiring a collaborator with specific training or knowledge. Certainly more accurate and available humans are of more use, but in no case tested across a range of simulated humans did the drone do significantly worse than the Nonhuman baseline. This indicates that in the worst of cases the POMDP is able to mitigate the effects of a poor human, while maintaining its ability to effectively collaborate with others.

Chapter 6

Conclusion

This thesis presented advancements to the state of the art for robotic planning and sensing, particularly for the application of human-robot collaborative search. Specifically, this work develops a formal framework for combined probabilistic planning and data fusion that allows human-robot teams to collaboratively exchange rich, flexible parcels of information via semantic communication, which can be adapted and applied to a broad range of problems involving autonomous decision-making under uncertainty. The major contributions of this work, in brief, are:

- The VB-POMDP Algorithm, a variational Bayes policy approximation which allows the use of continuous state POMDPs with hybrid continuous-to-discrete semantic sensor observations modeled by softmax functions. This allows POMDP planning using semantic dictionaries which are easily modelled and scaled, leading to enhanced probabilistic planning and data fusion in previously unapproachable POMDPs.

- A scalable hierarchical framework for efficiently solving CPOMDPs in complex continuous state spaces. This framework allows large structured continuous spaces to be partitioned into subspaces, each with its own unique semantic observation set, to be solved independently and connected via a discrete POMDP. It is further used as the basis for a novel framing of an active human-robot semantic sensing problem as a POMDP with a fixed map and known semantics. This framing casts POMDP actions as joint movement/queries decisions which actively pull information and accept volunteered information using natural

language statements. The framework and problem framing are jointly validated in hardware with live human input, showcasing the ability of human-robot teams leverage two-way semantic communication within the context of optimal planning problem.

- A novel sketch-based approach to multi-level active semantic sensing and planning in human-robot teams, allowing instant information transfer of both model and state transformation from human to robot without the need for policy retraining. This approach uses human sketches to augment and modify a dynamic semantic dictionary, leading to a time varying POMDP problem formulation. Such a POMDP is solved through a novel modification of Monte Carlo online POMDP planning techniques. Human input is further used to convey multimodal information regarding not only the target state, but environmental aspects indirectly influencing state transitions. These techniques, taken together, allow a broad range of dynamic problems to be solved as POMDPs by leveraging human input to modify the planning problem online, and were validated in multiple problem scenarios with simulated human input.

First, Chapter 3 of this work presented VB-POMDP, a variational Bayes policy approximation for solving continuous state POMDPs (CPOMDPs) with hybrid continuous-to-discrete semantic sensor observation likelihoods that are modeled by softmax models. Softmax models are ideal for modeling semantic observation likelihoods, and are cheaper and simpler to construct and evaluate compared to unnormalized GM functions that have been applied for the same purpose in other CPOMDP policy approximations. To overcome the analytical intractability of using softmax models in standard Gaussian mixture point-based value iteration (PBVI) policy approximations for CPOMDPs, a variational Gaussian inference approximation was developed to maintain the closed-form recursive nature of the Gaussian mixture PBVI approximation. This approach also tends to produce far fewer mixture terms in the intermediate PBVI recursion steps, and thus requires less overall computation to approximate the optimal policy. VB-POMDP was also explicitly extended to problems with linear time-invariant state dynamics, allowing a broad set of problems

to be addressed by the Gaussian mixture PBVI framework. A novel approach to Gaussian mixture condensation was also described and studied, whereby mixture terms are pre-clustered into sub-mixtures that are then condensed in parallel. This approach was shown to be considerably faster than conventional global mixture condensation techniques, while achieving similar accuracy.

Experimental simulations for a simple mobile target search and localization problem showed that VB-POMDP performed as well as an alternative GM-based CPOMDP policy approximation method, thus indicating that the approximations used by VB-POMDP do not lead to any significant compromises in optimality versus other state of the art approximations. However, VB-POMDP was shown to be significantly more effective on more complex and higher dimensional variants of the target search and localization problem. Simulations for policies trained on target state transition models differing from the true model showed that VB-POMDP is suitably responsive and robust to instances of model mismatch. Finally, VB-POMDP was shown to scale and perform well on a complex 10-dimensional continuous state multi-robot localization/goal-seeking problem, featuring highly non-Gaussian uncertainties as well as a large action and semantic observation space.

Next, in Chapter 4 a novel collaborative human-machine sensing solution for dynamic target search was developed and validated. The approach used continuous partially observable Markov decision process (CPOMDP) planning to generate vehicle trajectories that optimally exploit imperfect detection data from onboard sensors and semantic natural language observations that can be requested from human sensors. The main innovation was a scalable hierarchical Gaussian mixture model formulation for efficiently solving CPOMDPs with semantic observations in continuous dynamic state spaces. The approach was demonstrated with a real human-robot team engaged in dynamic indoor target search and capture scenarios on a custom testbed. The results showed that combined human-robot sensing not only enhances target localization quality (as expected), but that the resulting CPOMDP policies provide sensible simultaneous search movements and semantic human sensor queries that allow the search vehicle to intercept the target more efficiently. The resulting CPOMDP policies are robust and effective even with irregular/unpredictable inputs and occasional errors from the human sensor.

Finally, in Chapter 5 a novel approach to multi-level active semantic sensing and planning in human-robot teams in unknown dynamic environments was proposed and demonstrated. The solution extends online POMDP planning frameworks to incorporate semantic soft data from a human sensor about the location of a tracked target as well as relevant terrain information. Such information is propagated from human to robot through the use of a sketch based, natural language interface. This approach provides a novel formulation of a POMDP with a "human-in-the-loop" active sensing model, as well as innovations to the use of soft-data fusion as applied to higher level modal information. The approach was demonstrated across three example problems. The first showcased the improvements a queryable human can bring to target search problems by improving both terrain knowledge and semantic observation dictionaries, while the second demonstrated that the incorporation of human information even with regard to higher level mode observations can improve a robot's target search ability. The final problem domain, the Human-Aided Robotic Planning and Sensing (HARPS) platform, showcased the ability of the online "human-in-the-loop" to adapt to a broad variety of human input types for a large scale scenario with more realistic dynamics and constraints. In each case the robot was able to make effective and timely use of simulated human sketch information and binary observations in response to language-based queries to more quickly localize and intercept a moving target.

6.1 Potential Future Work

The results presented here have many interesting implications for developing and applying autonomous probabilistic planning and control algorithms in hybrid continuous-discrete domains. VB-POMDP retains many desirable properties of other GM-based PBVI policy approximation approaches. Among these is the ability to produce deterministic policy approximations for a given set of tent-pole beliefs, as well as the ability to naturally leverage Gaussian sum filters for Bayesian belief updates in domains featuring complex continuous state dynamics and uncertainties (which have also been shown to be more robust than particle filters for several robotics applications [6, 85]). In addition to the search, localization and goal-seeking applications described here, the CPOMDP

framework developed here is being leveraged for cooperative human-robot target search and tracking applications. Building on previous work in [6, 100] and ongoing work in [27], this will enable semantic 'human sensor data' from natural language inputs can be combined with optimal robotic sensing and motion planning in hardware for tightly integrated human-robot teaming. However, VB-POMDP could also be applied to other problems where discrete semantic observations are naturally available as a function of continuous dynamic state variables, e.g. active semantic mapping/SLAM, tactile reasoning for manipulation, collision avoidance, planning/control of stochastic hybrid dynamical systems, etc.

Various relaxations of modeling assumptions made in this work are also possible. For instance, as discussed in [27], VB-POMDP can be extended to non-linear state-dependent switching mode dynamics models to accommodate more complex probabilistic state transition pdfs that are modeled with softmax functions [24]. This work also motivates study of complex CPOMDPs with even larger spaces of actions and observations than those considered here, including continuous action spaces. The various algorithmic approximations presented here also warrant further analysis when used in approximating optimal planning. In particular, it is desirable to obtain bounds on the accuracy of the K-means hybrid GM condensation method, as well as possible lower bounds on the value function via the VB inference approximation to establish approximation error bounds with respect to the exact optimal CPOMDP policy.

Additionally, building on the semantic sketching research of Chapter 6, research will examine the effect of inaccurate or incomplete sketches on the part of the human. Pairing the robot with a visual object detector has potential to allow feedback to ensure that the robot's perception of the object or region is consistent with a sketch. Additionally, a 3D physics driven simulated search environment for large scale robotics testing is being developed in parallel with our interface to extend and test the ideas contained in this work using real humans in realistic search tasks. Hardware implementations with a UAS and live human are also ongoing.

Bibliography

[1] Pieter Abbeel. Reinforcement learning for nonlinear dynamical systems and Gaussian belief space planning. 2013 Conference on Reinforcement Learning and Decision Making (RLDM 2013), page 13, 2013.

[2] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In Proceedings of the twenty-first international conference on Machine learning, page 1, 2004.

[3] William T Adler and Wei Ji Ma. Comparing bayesian and non-bayesian accounts of human confidence reports. PLoS computational biology, 14(11):e1006572, 2018.

[4] Nisar Ahmed, Mark Campbell, David Casbeer, Yongcan Cao, and Derek Kingston. Fully bayesian learning and spatial reasoning with flexible human sensor networks. In Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems, pages 80–89. ACM, 2015.

[5] Nisar R. Ahmed. Data-free/data-sparse softmax parameter estimation with structured class geometries. IEEE Signal Processing Letters, 25(9):1408–1412, Sep. 2018.

[6] Nisar R. Ahmed, Eric M. Sample, and Mark Campbell. Bayesian multicategorical soft data fusion for human–robot collaboration. IEEE Transactions on Robotics, 29(1):189–206, 2013.

[7] Nicholas Armstrong-Crews and Manuela Veloso. Oracular partially observable markov decision processes: A very special case. In Proceedings 2007 IEEE International Conference on Robotics and Automation, pages 2477–2482. IEEE, 2007.

[8] Nicholas Armstrong-Crews and Manuela Veloso. An approximate algorithm for solving oracular pomdps. In 2008 IEEE International Conference on Robotics and Automation, pages 3346–3352. IEEE, 2008.

[9] Haoyu Bai, David Hsu, Mykel J Kochenderfer, and Wee Sun Lee. Unmanned aircraft collision avoidance using continuous-state POMDPs. Robotics: Science and Systems VII, 1:1–8, 2012.

[10] Haoyu Bai, David Hsu, and Wee Sun Lee. Integrated perception and planning in the continuous space: A POMDP approach. The International Journal of Robotics Research, 33(9):1288–1302, 2014.

[11] Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A Ngo. Monte Carlo value iteration for continuous-state POMDPs. In Algorithmic Foundations of Robotics IX, pages 175–191. Springer, 2010.

[12] Yaakov Bar-Shalom, X Rong Li, and Thiagalingam Kirubarajan. Estimation with Applications to Tracking and Navigation: Theory Algorithms and Software. John Wiley & Sons, 2004.

[13] Adrian N. Bishop and Branko Ristic. Fusion of spatially referring natural language statements with random set theoretic likelihoods. IEEE Transactions on Aerospace and Electronic Systems, 49(2):932–944, 2013.

[14] Federico Boniardi, Bahram Behzadian, Wolfram Burgard, and Gian Diego Tipaldi. Robot navigation in hand-drawn sketched maps. In 2015 European conference on mobile robots (ECMR), pages 1–6. IEEE, 2015.

[15] Devin Bonnie, Salvatore Candido, Timothy Bretl, and Seth Hutchinson. Modelling search with a binary sensor utilizing self-conjugacy of the exponential family. In 2012 IEEE International Conference on Robotics and Automation, pages 3975–3982. IEEE, 2012.

[16] Frédéric Bourgault. Decentralized control in a Bayesian world. PhD thesis, University of Sydney, 2005.

[17] Frederic Bourgault, Tomonari Furukawa, and Hugh F. Durrant-Whyte. Coordinated decentralized search for a lost target in a Bayesian world. In Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), volume 1, pages 48–53. IEEE, 2003.

[18] Mario Bourgault, Nathalie Drouin, and Emilie Hamel. Decision making within distributed project teams: An exploration of formalization and autonomy as determinants of success. Project Management Journal, 39(1_suppl):S97–S110, 2008.

[19] Jeffrey M Bradshaw, Robert R Hoffman, David D Woods, and Matthew Johnson. The seven deadly myths of" autonomous systems". IEEE Intelligent Systems, 28(3):54–61, 2013.

[20] G. Bradski. The OpenCV Library. Dr. Dobb's Journal of Software Tools, 2000.

[21] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Solving continuous POMDPs: Value iteration with incremental learning of an efficient space representation. In International Conference on Machine Learning, pages 370–378, 2013.

[22] Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous POMDPs. In 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pages 392–399. IEEE, 2014.

[23] Alex Brooks, Alexei Makarenko, Stefan Williams, and Hugh Durrant-Whyte. Parametric POMDPs for planning in continuous state spaces. Robotics and Autonomous Systems, 54(11):887–897, 2006.

[24] Emma Brunskill, Leslie Pack Kaelbling, Tomás Lozano-Pérez, and Nicholas Roy. Planning in partially-observable switching-mode continuous domains. Annals of Mathematics and Artificial Intelligence, 58(3-4):185–216, 2010.

[25] Luke Burks and Nisar Ahmed. Collaborative semantic data fusion with dynamically observable decision processes. In 2019 22nd International Conference on Information Fusion. IEEE, 2019.

[26] Luke Burks, Ian Loefgren, and Nisar R Ahmed. Optimal continuous state pomdp planning with semantic observations: A variational approach. IEEE Transactions on Robotics, 2019.

[27] Luke Burks, Ian Loefgren, Luke Barbier, Jeremy Muesing, Jamison McGinley, Sousheel Vunnam, and Nisar Ahmed. Closed-loop Bayesian semantic data fusion for collaborative human-autonomy target search. In 2018 21st International Conference on Information Fusion (FUSION 2018), pages 2262–2269. IEEE, 2018.

[28] Murray Campbell, A Joseph Hoane Jr, and Feng-hsiung Hsu. Deep blue. Artificial intelligence, 134(1-2):57–83, 2002.

[29] Laurent Charlin, Pascal Poupart, and Romy Shioda. Automated hierarchy discovery for planning in partially observable environments. In Advances in Neural Information Processing Systems, pages 225–232, 2007.

[30] Sonia Chernova and Andrea L Thomaz. Robot learning from human teachers. Synthesis Lectures on Artificial Intelligence and Machine Learning, 8(3):1–121, 2014.

[31] Pierre-Arnaud Coquelin and Rémi Munos. Bandit algorithms for tree search. arXiv preprint cs/0703062, 2007.

[32] Rémi Coulom. Efficient selectivity and backup operators in Monte-Carlo tree search. In International conference on computers and games, pages 72–83. Springer, 2006.

[33] Ashwin Dani, Michael McCourt, J Willard Curtis, and Siddhartha Mehta. Information fusion in human-robot collaboration using neural network representation. In 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pages 2114–2120. IEEE, 2014.

[34] Finale Doshi and Nicholas Roy. Spoken language interaction with model uncertainty: an adaptive human–robot interaction system. Connection Science, 20(4):299–318, 2008.

[35] Anca D Dragan, Shira Bauman, Jodi Forlizzi, and Siddhartha S Srinivasa. Effects of robot motion on human-robot collaboration. In 2015 10th ACM/IEEE International Conference on Human-Robot Interaction (HRI), pages 51–58. IEEE, 2015.

[36] Maxim Egorov, Zachary N. Sunberg, Edward Balaban, Tim A. Wheeler, Jayesh K. Gupta, and Mykel J. Kochenderfer. POMDPs.jl: A framework for sequential decision making under uncertainty. Journal of Machine Learning Research, 18(26):1–5, 2017.

[37] Epic Games. Unreal engine.

[38] Tom Erez and William D. Smart. A scalable method for solving high-dimensional continuous POMDPs using local approximation. arXiv preprint arXiv:1203.3477, 2012.

[39] Jamie Frost, Alastair Harrison, Stephen Pulman, and Paul Newman. A probabilistic approach to modelling spatial language with its application to sensor models. In Proceedings of the Workshop on Computational Models of Spatial Language Interpretation at Spatial Cognition (COSLI). Citeseer, 2010.

[40] Alex Goldhoorn, Anaís Garrell, René Alquézar, and Alberto Sanfeliu. Continuous real time POMCP to find-and-follow people by a humanoid service robot. In 2014 IEEE-RAS International Conference on Humanoid Robots, pages 741–747. IEEE, 2014.

[41] Nakul Gopalan and Stefanie Tellex. Modeling and solving human-robot collaborative tasks using pomdps. In RSS Workshop on Model Learning for Human-Robot Communication, 2015.

[42] David Lee Hall and John M Jordan. Human-centered information fusion. Artech House, 2010.

[43] Kris Hauser. Randomized belief-space replanning in partially-observable continuous spaces. In Algorithmic Foundations of Robotics IX, pages 193–209. Springer, 2010.

[44] Bradley Hayes and Brian Scassellati. Discovering task constraints through observation and active learning. In 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 4442–4449. IEEE, 2014.

[45] Ruijie He, Emma Brunskill, and Nicholas Roy. Puma: Planning under uncertainty with macro-actions. In Twenty-Fourth AAAI Conference on Artificial Intelligence, 2010.

[46] Jesse Hoey, Axel Von Bertoldi, Pascal Poupart, and Alex Mihailidis. Assisting persons with dementia during handwashing using a partially observable markov decision process. In International Conference on Computer Vision Systems: Proceedings (2007), 2007.

[47] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. Artificial Intelligence, 101(1-2):99–134, 1998.

[48] Tobias Kaupp, Bertrand Douillard, Fabio Ramos, Alexei Makarenko, and Ben Upcroft. Shared environment representation for a human-robot team performing information fusion. Journal of Field Robotics, 24(11-12):911–942, 2007.

[49] Tobias Kaupp, Alexei Makarenko, and Hugh Durrant-Whyte. Human–robot communication for collaborative decision making—A probabilistic approach. Robotics and Autonomous Systems, 58(5):444–456, 2010.

[50] Bahador Khaleghi, Alaa Khamis, and Fakhreddin Karray. Random finite set theoretic based soft/hard data fusion with application for target tracking. In 2010 IEEE Conference on Multisensor Fusion and Integration, pages 50–55. IEEE, 2010.

[51] Wolfgang Koch. On 'negative'information in tracking and sensor data fusion: Discussion of selected examples. In Proceedings of the Seventh International Conference on Information Fusion, volume 1, pages 91–98. IEEE Publ. Piscataway, NJ, 2004.

[52] Wolfgang Koch. On exploiting "negative" sensor evidence for target tracking and sensor data fusion. Information Fusion, 8(1):28–39, 2007.

[53] Mykel J Kochenderfer. Decision making under uncertainty: theory and application. MIT press, 2015.

[54] Michael C. Koval, Nancy S. Pollard, and Siddhartha S. Srinivasa. Pre-and post-contact policy decomposition for planar contact manipulation under uncertainty. The International Journal of Robotics Research, 35(1-3):244–264, 2016.

[55] Vikram Krishnamurthy and Dejan V. Djonin. Optimal threshold policies for multivariate POMDPs in radar resource management. IEEE Transactions on Signal Processing, 57(10):3954–3969, 2009.

[56] Hanna Kurniawati, David Hsu, and Wee Sun Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In Robotics: Science and Systems IV, pages 336–343. Zurich, Switzerland., 2008.

[57] Hanna Kurniawati and Vinay Yadav. An online pomdp solver for uncertainty planning in dynamic environment. In Robotics Research, pages 611–629. Springer, 2016.

[58] Benjamin Lavis and Tomonari Furukawa. HyPE: Hybrid particle-element approach for recursive Bayesian searching and tracking. Robotics: Science and Systems IV, pages 135–142, 2009.

[59] Kendra Lesser and Meeko Oishi. Approximate safety verification and control of partially observable stochastic hybrid systems. IEEE Transactions on Automatic Control, 62(1):81–96, 2017.

[60] Michael L Littman, Anthony R Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In Machine Learning Proceedings 1995, pages 362–370. Elsevier, 1995.

[61] Kin Gwn Lore, Nicholas Sweet, Kundan Kumar, Nisar Ahmed, and Soumik Sarkar. Deep value of information estimators for collaborative human-machine information gathering. In Proceedings of the 7th International Conference on Cyber-Physical Systems (ICCPS 2016), pages 3–12. IEEE Press, 2016.

[62] Siddhartha S Mehta, M McCourt, Emily A Doucette, and J Willard Curtis. A touch interface for soft data modeling in bayesian estimation. In 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC), pages 3732–3737. IEEE, 2014.

[63] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781, 2013.

[64] CHRIS Moore. Social cognition in infancy. Encyclopedia on Early Childhood Development, pages 1–4, 2010.

[65] Carl Mueller, Jeff Venicx, and Bradley Hayes. Robust robot learning from demonstration and skill repair using conceptual constraints. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 6029–6036. IEEE, 2018.

[66] Jeremy Muesing, Luke Burks, Michael Iuzzolino, Danielle Szafir, and Nisar R Ahmed. Fully bayesian human-machine data fusion for robust dynamic target surveillance and characterization. In AIAA Scitech 2019 Forum, page 2208, 2019.

[67] Sam Nicol and Iadine Chadès. Which states matter? An application of an intelligent discretization method to solve a continuous POMDP in conservation biology. PloS one, 7(2):e28993, 2012.

[68] Jarurat Ousingsawat and Mark E Campbell. On-line estimation and path planning for multiple vehicles in an uncertain environment. International Journal of Robust and Nonlinear Control, 14(8):741–766, 2004.

[69] Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs. In 2003 International Joint Conference on Artificial Intelligence (IJCAI 2003), volume 3, pages 1025–1032, 2003.

[70] Joelle Pineau, Nicholas Roy, and Sebastian Thrun. A hierarchical approach to pomdp planning and execution. In In ICML Workshop on Hierarchy and Memory in Reinforcement Learning. Citeseer, 2001.

[71] Robert Platt, Leslie Kaelbling, Tomas Lozano-Perez, and Russ Tedrake. Efficient planning in non-Gaussian belief spaces and its application to robot grasping. In Robotics Research, pages 253–269. Springer, 2017.

[72] Josep M. Porta, Nikos Vlassis, Matthijs T.J. Spaan, and Pascal Poupart. Point-based value iteration for continuous POMDPs. Journal of Machine Learning Research, 7(Nov):2329–2367, 2006.

[73] Samuel Prentice and Nicholas Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. The International Journal of Robotics Research, 28(11-12):1448–1465, 2009.

[74] Mohammadhussein Rafieisakhaei, Suman Chakravorty, and PR Kumar. Near-optimal belief space planning via T-LQG. arXiv preprint arXiv:1705.09415, 2017.

[75] Matthew Rosencrantz, Geoffrey Gordon, and Sebastian Thrun. Locating moving entities in indoor environments with teams of mobile robots. In Proceedings of the 2nd International Joint Conference on Autonomous Agents and Multi-agent Systems, pages 233–240. ACM, 2003.

[76] Stephanie Rosenthal and Manuela Veloso. Modeling humans as observation providers using pomdps. In 2011 RO-MAN, pages 53–58. IEEE, 2011.

[77] Stephanie Rosenthal, Manuela Veloso, and Anind K Dey. Learning accuracy and availability of humans who help mobile robots. In Twenty-Fifth AAAI Conference on Artificial Intelligence, 2011.

[78] Stephane Ross, Brahim Chaib-draa, and Joelle Pineau. Bayes-adaptive pomdps. In Advances in neural information processing systems, pages 1225–1232, 2008.

[79] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for pomdps. Journal of Artificial Intelligence Research, 32:663–704, 2008.

[80] Nicholas Roy and Geoffrey J. Gordon. Exponential family PCA for belief compression in POMDPs. In Advances in Neural Information Processing Systems, pages 1667–1674, 2003.

[81] Andrew R. Runnalls. Kullback-Leibler approach to Gaussian mixture reduction. IEEE Transactions on Aerospace and Electronic Systems, 43(3):989–999, 2007.

[82] Allison Ryan and J. Karl Hedrick. Particle filter based information-theoretic active sensing. Robotics and Autonomous Systems, 58(5):574–584, 2010.

[83] David J. Salmond. Mixture reduction algorithms for target tracking in clutter. In Signal and Data Processing of Small Targets 1990, volume 1305, page 434. International Society for Optics and Photonics, 1990.

[84] Eric Sample, Nisar Ahmed, and Mark Campbell. An experimental evaluation of bayesian soft human sensor fusion in robotic systems. In AIAA Guidance, Navigation, and Control Conference, page 4542, 2012.

[85] Jonathan R. Schoenberg, Mark Campbell, and Isaac Miller. Posterior representation with a multi-modal likelihood using the gaussian sum filter for localization in a known map. Journal of Field Robotics, 29(2):240–257, 2012.

[86] Danelle Shah, Joseph Schneider, and Mark Campbell. A sketch interface for robust and natural robot control. Proceedings of the IEEE, 100(3):604–622, 2012.

[87] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In Field and Service Robotics, 2017.

[88] Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based pomdp solvers. Autonomous Agents and Multi-Agent Systems, 27(1):1–51, 2013.

[89] David Silver and Joel Veness. Monte-Carlo planning in large POMDPs. In Advances in neural information processing systems, pages 2164–2172, 2010.

[90] Marjorie Skubic, Craig Bailey, and George Chronis. A sketch interface for mobile robots. In SMC'03 Conference Proceedings. 2003 IEEE International Conference on Systems, Man and Cybernetics. Conference Theme-System Security and Assurance (Cat. No. 03CH37483), volume 1, pages 919–924. IEEE, 2003.

[91] Trey Smith and Reid Simmons. Heuristic search value iteration for pomdps. arXiv preprint arXiv:1207.4166, 2012.

[92] Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. DESPOT: Online POMDP planning with regularization. In Advances in neural information processing systems, pages 1772–1780, 2013.

[93] Edward J Sondik. The optimal control of partially observable markov decision processes. PhD the sis, Stanford University, 1971.

[94] Tristram Southey and James J Little. Learning qualitative spatial relations for object classification. In IROS 2007 Workshop: From Sensors to Human Spatial Concepts, 2007.

[95] Matthijs T.J. Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. Journal of Artificial Intelligence Research, 24:195–220, 2005.

[96] Mohan Sridharan, Jeremy Wyatt, and Richard Dearden. Planning to see: A hierarchical approach to planning visual actions on a robot using pomdps. Artificial Intelligence, 174(11):704–725, 2010.

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

[98] Kevin V Stefanik, Jason C Gassaway, Kevin Kochersberger, and A Lynn Abbott. Uav-based stereo vision for rapid aerial terrain mapping. GIScience & Remote Sensing, 48(1):24–49, 2011.

[99] Zachary N Sunberg and Mykel J Kochenderfer. Online algorithms for pomdps with continuous state, action, and observation spaces. In Twenty-Eighth International Conference on Automated Planning and Scheduling, 2018.

[100] Nicholas Sweet and Nisar Ahmed. Structured synthesis and compression of semantic human sensor models for Bayesian estimation. In 2016 American Control Conference (ACC), pages 5479–5485. IEEE, 2016.

[101] Sebastian Thrun. Monte carlo pomdps. In Advances in neural information processing systems, pages 1064–1070, 2000.

[102] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). The MIT Press, 2005.

[103] Marc Toussaint, Laurent Charlin, and Pascal Poupart. Hierarchical pomdp controller optimization by likelihood maximization. In UAI, volume 24, pages 562–570, 2008.

[104] Jur Van Den Berg, Pieter Abbeel, and Ken Goldberg. LQG-MP: Optimized path planning for robots with motion uncertainty and imperfect state information. The International Journal of Robotics Research, 30(7):895–913, 2011.

[105] Jur Van Den Berg, Sachin Patil, and Ron Alterovitz. Efficient approximate value iteration for continuous Gaussian POMDPs. In Twenty-Sixth AAAI Conference on Artificial Intelligence, 2012.

[106] Jason L. Williams and Peter S. Maybeck. Cost-function-based Gaussian mixture reduction for target tracking. In Proceedings of the sixth international conference of Information fusion, volume 2, pages 1047–1054. IEEE Publ. Piscataway, NJ, 2003.

[107] Lin Wu, Yang Wang, Xue Li, and Junbin Gao. Deep attention-based spatially recursive networks for fine-grained visual recognition. IEEE transactions on cybernetics, 49(5):1791–1802, 2018.

[108] Kevin Wyffels and Mark Campbell. Negative information for occlusion reasoning in dynamic extended multiobject tracking. IEEE Transactions on Robotics, 31(2):425–442, 2015.

[109] Yongquan Zhang and Hongbing Ji. Gaussian mixture reduction based on fuzzy ART for extended target tracking. Signal Processing, 97:232–241, 2014.

[110] Enlu Zhou, Michael C. Fu, and Steven I. Marcus. Solving continuous-state POMDPs via density projection. IEEE Transactions on Automatic Control, 55(5):1101–1116, 2010.