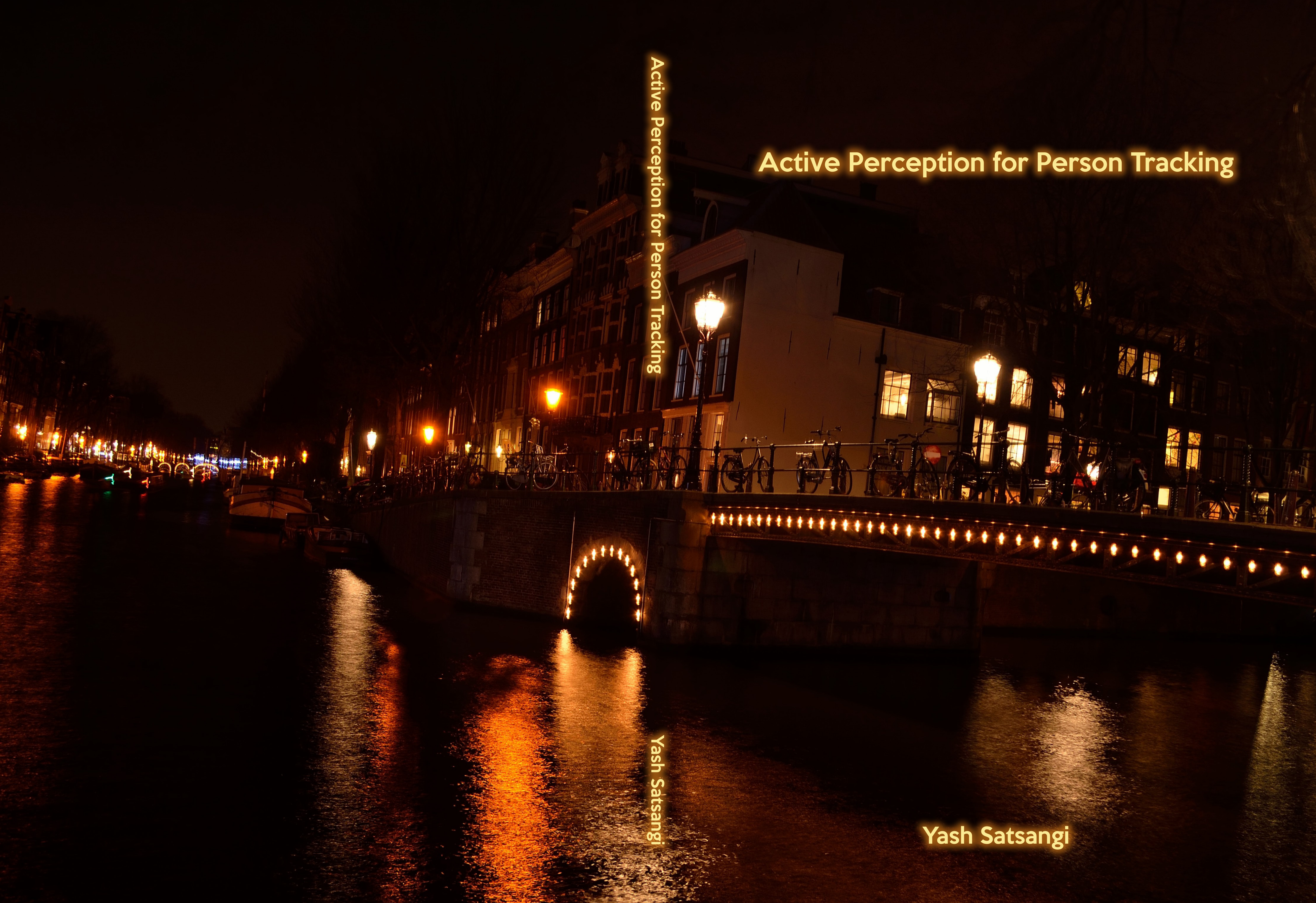
Active perception for person tracking

Satsangi, Y.



Active Perception for Person Tracking

Yash Satsangi

Active Perception for Person Tracking

Yash Satsangi

Active Perception for Person Tracking

Yash Satsangi

This research was supported by the Dutch Technology Foundation STW (project # 12622), which is part of the Netherlands Organisation for Scientific Research (NWO), and which is partly funded by the Ministry of Economic Affairs.

Printed by: ProefschriftMaken|| www.proefschriftmaken.nl
ISBN: 978-94-6380-163-8

Active Perception for Person Tracking

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. K. I. J. Maex
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Aula der Universiteit
op vrijdag 25 Januari 2019, te 11.00 uur

door

Yash Satsangi

geboren te Indore

Promotiecommissie

Promotor:		
	prof. dr. M. Welling	Universiteit van Amsterdam
Co-promotor (es):		
	dr. S. A. Whiteson	Universiteit van Amsterdam
	dr. F. A. Oliehoek	Universiteit van Amsterdam
Overige leden:		
	prof. dr. C. Szepesvari	University of Alberta
	prof. dr. T. Gevers	Universiteit van Amsterdam
	prof. dr. M. Worring	Universiteit van Amsterdam
	dr. M. T. J. Spaan	TU Delft
	dr. J. Mooij	Universiteit van Amsterdam

Faculteit der Natuurwetenschappen, Wiskunde en Informatica

Contents

1 Introduction

A key challenge in the field of Artificial Intelligence (AI) is the design of intelligent autonomous agents that can understand their environment and behave *rationally* in it [Russell and Norvig, 2009]. While there are multiple definitions of intelligence, the basic trait of any intelligent entity is the ability to perceive its environment and react appropriately to it [Stangor, 2010]. *Perception* refers to understanding the environment an entity is *acting* in at an appropriate level of abstraction. *Control* refers to taking actions that can cause a desired behaviour. Based on these basic definitions, Figure 1.1 shows the design of an agent made up of a perception and a control module.

The perception module processes the raw data collected by the agent's sensors, example images from the visual sensors or speech from the audio sensors, generally with the aide of signal processing and machine learning algorithms to infer the hidden *state* of the world. The result of processing the raw data is then passed to the control module. Depending on the state of the world and the aim of the agent, the control module takes actions so that the agent gets closer to its goal.

Generally, the perception module consists of *signal processing* and *machine learning* algorithms that can process the raw information from the sensors to infer the hidden state of the world. [Bajcsy, 1988, Bajcsy et al., 2016]. However, perception in itself is a complex process that is not limited to the processing of raw information but instead requires the interaction of an agent with its environment. For example, humans continuously explore and interact with their environment to understand it better. Consequently, questions like "where to look", "when to look", "what to remember", etc. are naturally easy for the humans to answer. Similarly, a truly intelligent and autonomous agent must have the ability to reason about its uncertainty and come up with strategies for information gathering. The perception module, thus, in itself presents a control problem the aim of which

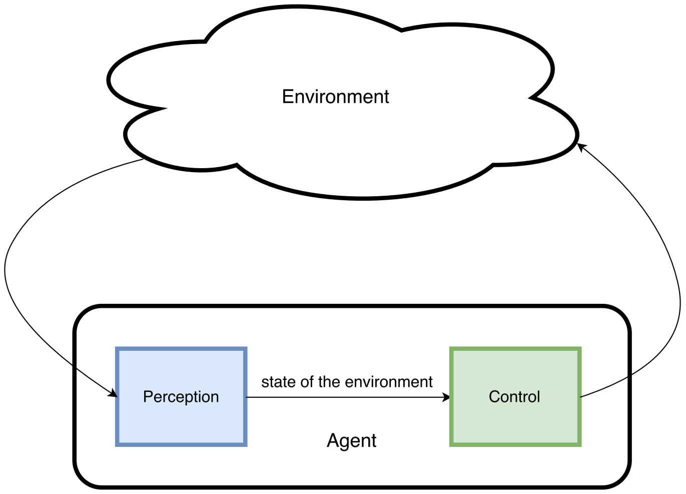


Figure 1.1: A simple agent consisting of a perception and control module.

is to come up with strategies for intelligent information acquisition so that the perception module can best infer the state of the world.

Active perception [Bajcsy, 1988, Bajcsy et al., 2016] means that an agent is capable of coming up with control strategies for information gathering while reasoning about its own limitations and constraints. These limitations can be inherent in the environment, e.g., occlusions in images [Le et al., 2008]; or it can be due to the limitations of the agent's design, e.g., faulty sensors [Spaan et al., 2015]. For example, consider an agent navigating in an unknown environment trying to map as many 3D objects as possible [Patten et al., 2015]. To do so the agent must navigate safely in the unknown environment, and to accurately detect a 3D object it must align itself in a position that is most suitable for its sensor and the object detection algorithm. Similarly, an autonomous underwater vehicle (AUV) [Binney et al., 2010] sent in the ocean to collect oceanographic data must navigate its way such that it does not crash into the ships or the landmasses that it encounters and still collect as much useful data as possible. At the same time it must monitor its battery to make sure it has sufficient power left to navigate and resurface at a point where the human operator can safely collect it. Another example of an active perception task is an agent focusing on the most relevant part of an image/video when faced with information overload that cannot be handled by the resources that are at its disposal [Satsangi et al., 2017a].

This thesis tackles the control problem for active perception with a special focus on resource allocation in multi-camera systems for surveillance and automated tracking. A key challenge in the design of multi-sensor systems is the efficient allocation of scarce resources, for example, computational power, bandwidth, energy, manpower, etc [Spaan and Lima, 2009, Satsangi et al., 2015a]. These resource constraints give rise to the *sensor selection* problem [Kreucher et al., 2005, Williams et al., 2007, Satsangi et al., 2015a] where an agent must select a subset of the available sensors to allocate the scarce resources to reduce the uncertainty about its environment. Following a decision-theoretic approach this thesis presents principled methods for efficient resource allocation in multi-sensor systems and extends them to general active perception problems. In the rest of this chapter, the key features of the active perception and the sensor selection problem are discussed; followed by the motivation for the decision-theoretic approach, the focus of this thesis and the research questions addressed in this thesis.

1.1 Active Perception

Active perception can be defined as the study of the control strategies that aim to reduce the uncertainty about the state of the world, while reasoning about various constraints imposed on an agent. Typically, reducing uncertainty is only a means to an end, for example, a robot whose goal is to reach a particular location may take sensing actions that reduce its uncertainty about its current location because doing so helps it determine what future actions will bring him closer to its goal. By contrast, in *pure* active perception reducing uncertainty is an aim in itself. For example, consider an agent trying to maintain surveillance in a shopping mall. Typically, the goal of such an agent is to ascertain the state of its environment and not use that knowledge to achieve a goal. While perception is arguably always performed to aid decision-making, in an active perception problem that decision is made by another agent such as a human, that is not necessarily modelled as a part of the agent. For example, in the surveillance task, the agent might be able to detect a suspicious activity but only the human users of the system may decide how to react to such an activity. Similarly, an autonomous underwater vehicle can collect oceanographic data, but it must be retrieved and analysed by the human user. While there are approaches [Eck and Soh, 2012, Spaan et al., 2015] that can model the decision-making and the information gathering as a part of the same agent for *hybrid* tasks, in this thesis we focus on the *pure* active perception tasks where the aim of the agent is solely to reduce the uncertainty about its environment, specifically for multi-camera networks to maintain surveillance or to track people in large spaces.

When designing a multi-sensor system a key challenge is to allocate the scarce resources efficiently. For example:

- Manpower: Often maintaining surveillance in public spaces such as a shopping mall or an airport requires human operators to monitor the video streams from a myriad number of cameras. In such cases, the manpower is a scarce resource as humans, in general, have limited attention span and there may not be enough human operators to monitor the large number of video streams from the camera network. Thus, the manpower must be utilized efficiently by selectively displaying regions of high activity or high importance.

- Computational Power: In many settings, the computational power is a scarce resource, as it is not possible to process all the images from all the cameras at every point of time due to the high computational cost of the sophisticated image processing algorithms. This is especially true when ultra high-resolution images or/and camera networks with a large number of cameras are involved. In cases like this, the agent must select the appropriate regions to which to apply the processing algorithms on while reasoning about its uncertainty about the state of the world.

- Bandwidth: In some cases, there is not sufficient bandwidth available to transfer all the images to the computational unit because of the large size of the images. In case of a human operator typically the number of displays is less than the total number of cameras deployed, thus not all the images can be displayed in front of the human operator at the same time. The agent in such situations must select the most informative set of images that must be sent to the computational unit or to the human operator.

- Energy: Energy is a scarce resource simply because the user of a multi-camera system would not want all the cameras and the displays to be 'ON' at every point of time to minimize the energy consumption.

These resource constraints give rise to the *sensor selection* [Williams et al., 2006, Spaan and Lima, 2009, Natarajan et al., 2012, Satsangi et al., 2015a] problem where an agent must select k out of the n available sensors to allocate the given resources, where k is the maximum number of sensors allowed given the resource constraints, in order to minimize the uncertainty about its environment.

When the state is static, a *myopic* approach that maximizes the immediate expected reduction in uncertainty is typically sufficient. However, when the state changes over time, the agent must reason about the consequences of its own actions over a longer period of time. For example, as the people in a shopping mall move around, the agent must reason about the consequences of its selection of k cameras depending on the motion of people in the shopping mall over the next few time steps to find the most informative subset of sensors to maintain surveillance. The agent must also reason about the limitations of its sensors, since the sensory units and the data processing algorithms are typically not 100% accurate. Thus, the agent cannot directly fully observe the world instead it must deal with partial observability. In the surveillance example, the agent must reason about the accuracy of the *person detection algorithm* [Dollár et al., 2010], that is used to detect the people in the scene when deciding the utility of a camera/sensor.

Dynamic scene, partial observability and long-term reasoning are the common features of the active perception problems. In the next section we describe the approach of this thesis to tackle the challenges of active perception.

1.2 Approach

Active perception problems require an agent to reason about the utility of its actions over a period of time when the outcome of its actions is uncertain. Decision-theoretic approaches enable an agent to identify actions that have high utility under uncertainty [Boutilier et al., 1999]. A natural decision-theoretic model for the challenges common to the active perception problems is the *partially observable Markov decision process* (POMDP) [Astrom, 1965, Sondik, 1971, Kaelbling et al., 1998, Kochenderfer, 2015]. POMDPs provide a comprehensive and powerful framework for planning and learning under uncertainty. They can model the dynamic and partially observable state and express the goals of the agent in terms of rewards associated with state-action pairs. The agent at every time step maintains the knowledge about the state of the world in terms of a probability distribution also called as a belief. Using the POMDP model of the world the agent can compute a policy that is a mapping from the beliefs to the actions such that it tells the agent which action to take when it encounters a certain belief. In the surveillance example, by modelling the motion of the people and by accounting for the accuracy of the person detection algorithms, the agent can simulate the future possibilities resulting from his actions to find the one that earns the most reward, which in this case is the one that reduces the uncertainty about the world.

In a POMDP model, at each timestep, the agent takes an action and receives an *observation* that is correlated with the true state of the world. Using this observation the agent can update its belief about the world. Depending on the true state and the action taken by the agent, the agent also receives a positive or negative scalar reward. The aim of the agent is to take actions so that it can maximize the expected sum of rewards that it collects over a period of time. Figure 1.2 shows an example POMDP model of an agent managing resources in an automated tracking system by selectively applying

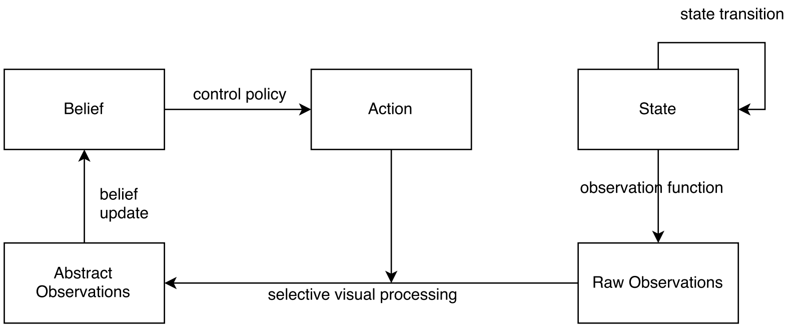


Figure 1.2: A POMDP agent selectively applying visual processing algorithms to raw observations generated by cameras/sensors. The agent's actions here are to select which visual processing algorithm to apply on the raw sensor observations.

visual processing algorithms to the raw observations. Because the agent's actions are dependent on its belief, it empowers it to adjust its visual processing according to its current uncertainty about the world.

Planning Methods: Given a POMDP model, *planning methods* [Sondik, 1971, Cheng, 1988, Ross et al., 2008, Bonet and Geffner, 2009, Shani et al., 2012] can be used to compute the *optimal value function* which specifies the expected cumulative long-term reward of an action. The *optimal policy* can be simply derived from the optimal value function by selecting for each belief the action that maximizes the value function. While numerous methods exist for computing the optimal value function exactly [Sondik, 1971, Monahan, 1982], their computational costs are prohibitively expensive for all but small problems. Approximate planning methods [Hauskrecht, 2000, Pineau et al., 2006, Shani et al., 2012] scale better by either exploiting independence represented using algebraic decision diagrams [Hoey et al., 1999] or by decomposing the POMDP and solving it hierarchically [Toussaint et al., 2008]. In particular, *point-based methods* [Pineau et al., 2006, Shani et al., 2012] have shown promising performance on POMDPs that are of reasonably large size. Point-based methods compute the optimal value function only for a sample of the belief points and use it to approximate the value function for the rest of the belief points.

The idea of planning for the entire belief space before the agent executes the task falls in the category of *offline planning*. For large problems, *online planning* [Ross et al., 2008] can be a better option as the agent here plans while executing the task. Consequently, the agent aims to find the optimal action only for the belief that it encounters as it executes certain actions. In general, online planners rely on the simulations generated using the

POMDP model of the world and the results of the simulations are then used to estimate the value of each action. Monte Carlo tree search (MCTS) [Silver and Veness, 2010], one of the most popular online planners uses simulations to construct a tree estimating the effect of taking each action in the current state and its successor state encountered during the simulations. Using *confidence intervals* on the value of each action, the agent can focus on the more promising actions while only sufficiently exploring the highly uncertain actions. The main challenge with planning online is that tracking people is a real-time task and thus the online planner must be able to plan for large POMDPs in real-time, which is a computationally intensive task. Moreover, when the computational power is a scarce resource, MCTS might end up using the same scarce computational resources that the agent aims to intelligently allocate.

Learning POMDP Model: Applying a POMDP planning method requires a model of the POMDP, i.e., description of the probability distributions governing the reward, the state transition function and the observation function. Depending on the individual case, the reward function in many situations is known in advance as it can be hand-designed by the user. In principle, the reward function in a POMDP is expressed as a function of the state and the action. For active perception the aim of the agent is to minimize the uncertainty about its environment. One way to formulate uncertainty reduction as an end in itself is to define a reward function whose additive inverse is some measure of the agent's uncertainty, e.g., *belief entropy* [Cover and Thomas, 1991]. The traditional POMDP framework does not allow for the reward to be expressed as a function of the belief. ρPOMDP [Araya-lópez et al., 2010] and *POMDP with Information Rewards* (POMDP-IR) [Spaan et al., 2015] are two frameworks that can approximately model the reward function as a function of the belief and thus we use them to model the belief-based rewards in our setting.

The transition function in a POMDP describes the probability of transitioning from one state to another under the Markov assumption, i.e., the next state is solely dependent on the previous state and the action the agent takes. Learning it in the POMDP setting poses a unique challenge, as the state changes are not directly observable with the camera network, which only records the observations. Since in our setting, the state transition mainly involves modelling the motion of people in public spaces, it can be achieved by exploiting the vast computer vision literature on person tracking and motion modelling, especially which are used for decision-making [Joseph et al., 2011, Denil et al., 2012, Kooij et al., 2012, Smeulders et al., 2014]. The observation function describes the probability of receiving an observation after taking an action in a state. Depending on the person detector [Dollár et al., 2012, 2014] and given some data about the true position of the people in certain images probability of getting a right/wrong observation can be approximated.

Learning the optimal policy: Instead of learning a POMDP model and then computing the optimal policy, it is also possible to directly learn the optimal policy without explicitly modelling and learning the transition and observation functions. In contrast to planning methods (also called as *model based* reinforcement learning methods), *model free* reinforcement learning (RL) [Sutton and Barto, 1998] methods such as Q learning [Watkins and Dayan, 1992] and temporal difference (TD) learning [Sutton, 1988] aim to directly learn the optimal policy from the past experience of the agent. The key idea behind a model free RL approach is that the agent starts with a random policy to interact with the world and as it accumulates experience it updates the policy according to a certain update rule. With sufficient experience the agent is able to learn the optimal policy from its experience.

Model free approaches tend to be data intensive as the agent must explore enough possibilities before finding the best among all. However, they eliminate the need to make explicit assumptions and design choices for the model of the world that when not satisfied in the real-world may lead to bad performance. For active perception tasks, learning a policy directly from the experience is challenging because the agent must visit the infinitely many possible belief states and explore relations among the large action space to learn the optimal policy. However, with the recently developed *deep reinforcement learning* methods like deep Q networks (DQN) [Mnih et al., 2013] and deep recurrent Q networks (DRQN) [Hausknecht and Stone, 2015] it is possible to address this challenge by learning efficient representations of the belief space to approximately learn the optimal policy.

While the above mentioned approaches are principled and promising in terms of the empirical performance, scaling them to large problems remains a challenge. In the next section we sum up the research questions and challenges addressed in this thesis to scale decision-theoretic approaches to large problems.

1.3 Research Questions & Contributions

The broad question that this thesis tackles is *"How can a multi-sensor system allocate its own resources in a principled and efficient way?"* We formulate the problem of resource allocation as selecting k out of the n available sensors to allocate the available resources. We develop and learn a POMDP model of the world that an agent can use to identify the actions with high utility where the aim of the agent is to minimize the uncertainty in its belief about the world. In the attempt to build a tracking system that can effectively allocate its own resources, we focus on the following research questions:

- RQ 1 "How can we efficiently formulate uncertainty as an end reward in POMDPs?"

  Traditionally, the reward in a POMDP is specified in terms of state-action tuples, which is exploited by most existing POMDP planners. As uncertainty is a function of the belief of the agent, the reward in such cases is a function of the belief of the agent and not the state of the world. Araya-lópez et al. [2010] introduced ρPOMDP that extends the traditional POMDP framework to allow belief-based reward functions (reward functions conditioned on the belief of the agent instead of the state of the world). Similarly, POMDP-IR [Spaan et al., 2015] rewards the agent for correctly predicting the state of the world leading to beliefs with less uncertainty. However, it is not clear out of ρPOMDP and POMDP-IR, which is the 'better' framework to model belief-based rewards in POMDPs? Moreover, since both ρPOMDP and POMDP-IR are relatively new, it is not clear what are the advantages or disadvantages of using belief based rewards against state based rewards in POMDPs. Chapter 3 answers these questions by showing that ρPOMDP and POMDP-IR are equivalent frameworks, i.e., any POMDP-IR can be translated to be a ρPOMDP (and vice-versa) that preserves the value function for equivalent policies. We also introduce a simple factorization applicable to the action space of any POMDP-IR, leading to better computational efficiency of the planning methods for POMDP-IR and ρPOMDP. Finally, we give a detailed empirical analysis to compare state based rewards and belief based rewards that identifies the critical factors relevant to the performance and the behaviour of an agent for the active perception tasks.

- RQ 2 "How can an agent perform long-term planning for active perception POMDPs with large combinatorial action spaces?"

  Existing planning methods for POMDPs are able to address a large state space in POMDPs [Pineau et al., 2006, Shani et al., 2012]. However, planning for POMDPs with large combinatorial action spaces still remains a challenge. Chapter 4 tackles the challenge of long-term planning for large action space POMDPs. First, we propose a new point-based planning method, greedy point-based value iteration (PBVI) that scales much better in the action space of a POMDP. The main idea is to replace the maximization operator in the *Bellman optimality equation* with greedy maximization. We present theoretical results bounding the error in the value function computed by greedy PBVI. Furthermore, we prove that, under certain conditions including submodularity the value function computed using POMDP backups based on greedy maximization has bounded error. We achieve this by extending the existing results [Nemhauser et al., 1978] on greedy maximization, which are valid only for single time step, to a full sequential decision making setting where the greedy operator is employed multiple time over multiple timesteps. In addition, we show that the conditions required for such a guarantee to hold are met, or approximately met, if the reward is defined as negative belief entropy. Experiments on a real-life dataset show that greedy PBVI outperforms a *myopic* baseline and almost matches the performance of existing point-based planners at a fraction of the computational cost leading to overall better scalability.

- RQ 3 "How can an agent select k out of n available cameras online in multi-camera networks to track people in large public spaces?"

  When large spaces such as airports and shopping malls are involved maintaining exact probability distribution over the hidden state is not possible due to the large size of the state space (all possible configurations of the hidden state). Consequently, it is not possible to evaluate the information-theoretic definitions of the utility/value function exactly. Greedy maximization and its variants [Mirzasoleiman et al., 2015] assume an *oracle* access to the function that is being maximized. However, information-theoretic definitions of the utility functions are typically hard to evaluate exactly, especially for large state spaces. Thus, Chapter 5 presents *probably approximately correct* greedy maximization (PACGM) that rather than computing the value function exactly assumes access to confidence bounds on it. These confidence bounds are used to prune actions that with high probability are sub-optimal. Furthermore, we provide a PAC analysis that shows that with high probability PACGM returns an approximately optimal set. Given an unbiased estimator for a function, it is possible to use traditional methods like the Hoeffding's inequality to obtain confidence bound on the function. However, the information-theoretic definitions that can quantify the uncertainty in a probability distribution require computing the discrete entropy of the posterior distribution, which is hard to estimate in an unbiased way. Therefore, we present novel and cheap confidence bounds on *conditional entropy*. Finally, we apply PACGM with these new confidence bounds to a real-life dataset to show empirically that PACGM scales better than greedy maximization.

- RQ 4 "How can an agent focus on the relevant parts of a ultra high-resolution video to track people in real-time?"

  In settings that involve tracking objects in ultra high-resolution images (aerial shots of an area) even detecting an object in the image can take many seconds since a trained object detector must be applied at all possible locations/pixels in the image to detect the object. For real-time object tracking in such systems it is not possible to apply the object detector to all the possible locations in every image. Chapter 6 focuses on resource allocation for real-time tracking in tracking systems with ultra high-resolution images [Satsangi et al., 2017a]. The main focus of this chapter is to make real-time tracking possible in ultra high-resolution videos by selective detection that applies a trained person detector only to a handful of locations in an image instead of applying it at all possible locations. First we propose a new utility function called particle coverage for selective detection, which is much faster to compute than the traditional utility functions that exist in literature. To quickly identify the relevant locations to apply a trained detector on, we propose a new algorithm called PartiMax that maintains and updates the particle coverage of each candidate location in the image in each iteration of greedy maximization instead of computing it from scratch. Moreover, PartiMax subsamples a subset of locations with high particle coverage without computing their particle coverage and then selects the best location out of this subset of locations, thus, avoiding the computation cost of going over all available locations. Since sampling locations with high particle coverage without computing it is not trivial, we propose a sampling algorithm that can sample high particle coverage locations in constant time. This is achieved by employing *tile coding*, a popular reinforcement learning method for discretizing continuous spaces. Furthermore, we prove that our sampling algorithm is guaranteed to sample a location with a probability that is directly proportional to its particle coverage. We show that given access to our sampling algorithm, PartiMax is guaranteed to return a solution with error bounds that are independent of the problem size. Finally, we use PartiMax to build a real-time tracking system that is able to retain 80% of original tracking performance by processing only 10% of the image.

- RQ 5 “How can an agent controlling a multi-camera network learn to allocate its own resources without having to explicitly learn a model of the world?”

  Planning methods require an accurate model of the world to compute the optimal policy. In some cases acquiring or designing such a model of the world is infeasible. Thus, the agent must learn an optimal behaviour directly from its experience. Chapter 7 tackles this challenge by proposing an *end-to-end* approach for active perception that enables an agent to take actions to minimize its uncertainty directly from the raw sensor data. In this chapter, we propose deep anticipatory networks (DAN) that enable an agent to take actions to best predict the current and future state of the world. DAN consists of two deep neural network architectures: a DQN and a model network. The Q network select k sensors out of the n available sensors. The raw observations from these k sensors are fed to the model network that predicts the true state of the world. The Q network is in turn rewarded if the model network correctly predicts the state of the world, thus, encouraging the Q network to select the potentially most informative sensors. Furthermore, we propose a training algorithm that given some data trains DAN to learn the optimal policy in an end-to-end manner. Finally we employ DAN to select k out of n sensors to track people in a simulated setting.

2 Background

This chapter introduces the background information essential for understanding this thesis. First we introduce the POMDP framework and the existing methods for finding the optimal policy given a POMDP model. Then, we give a formal definition of the active perception POMDP for the sensor selection task. In the traditional POMDP framework the reward function is specified as a function of the state and the action. For active perception the reward is a function of the belief of the agent, thus, we describe ρPOMDP and POMDP-IR, two frameworks that allow a belief-based definition of the reward function. The action space of the active perception POMDP is modelled as $\binom{n}{k}$ subsets of sensors, each of which denotes a choice of k out of the n available sensors. Identifying the optimal subset of sensors among these many choices can be computationally intensive. To overcome this challenge, we exploit submodularity to propose approximate and computationally tractable solutions for selecting k items out of n. Thus, at the end of this chapter we give a brief introduction to submodular function maximization and the related results.

2.1 Partially Observable Markov Decision Process

POMDPs provide a decision theoretic framework for modelling partial observability and dynamic environments [Kaelbling et al., 1998]. The goal of the agent can be expressed in terms of a reward function that specifies the reward associated with each state-action pair. The agent maintains its knowledge about the world in form of a probability distribution called the belief and by reasoning with the POMDP model can compute a policy (a mapping from the beliefs to the actions) such that it maximizes the expected sum of rewards over a period of time.

Formally, a POMDP is defined by a tuple $\langle S, A, \Omega, T, O, R, b_0, h\rangle$. At each time step, the environment is in a state $s \in S$, the agent takes an action $a \in A$ and receives a reward whose expected value is $R(s,a)$, and the system transitions to a new state $s' \in S$ according to the transition function $T(s,a,s') = \Pr(s'|s,a)$. Then, the agent receives an observation $z \in \Omega$ according to the observation function $O(s',a,z) = \Pr(z|s',a)$. Starting from an initial belief b_0, the agent maintains a *belief* $b(s)$ about the state which is a probability distribution over all possible states. If the agent took action a in belief b and got an observation z, then the updated belief $b_z^a(s)$ can be computed using Bayes rule:

$$b_z^a(s') = \Pr(z|s',a)\frac{\sum_s \Pr(s'|s,a)b(s)}{\Pr(z|b,a)} \;\forall\, s' \;\in\; S, \tag{2.1}$$

where

$$\Pr(z|b,a) = \sum_{s'}\sum_{s}\Pr(z|s',a)\Pr(s'|a,s)b(s). \tag{2.2}$$

A policy π is a mapping from the beliefs to the actions that specifies how the agent acts in each belief. Given $b(s)$ and $R(s,a)$, belief-based reward, $\rho(b,a)$ can be defined as:

$$\rho(b,a) \triangleq \sum_s b(s)R(s,a). \tag{2.3}$$

The optimal policy denoted by π^* maximizes the expected cumulative future reward: $\pi^* = \arg\max_\pi \mathbb{E}[\sum_{t=0}^{h-1} r_t | a_t = \pi(b_t)]$, where h is a finite time horizon after which the decision-making process (alternatively called an episode) ends and r_t, a_t and b_t are the reward, action and belief at time t.

The t-step value function of a policy V_t^π can be characterized recursively using the *Bellman equation*:

$$V_t^\pi(b) \triangleq \left[\rho(b,a_\pi) + \sum_{z\in\Omega}\Pr(z|a_\pi,b)V_{t-1}^\pi(b_z^{a_\pi})\right], \tag{2.4}$$

where $a_\pi = \pi(b)$ and $V_0^\pi(b) = \rho(b,a_\pi)$ (or it can be 0 as well). The action-value function $Q_t^\pi(b,a)$ is the value of taking action a and following π thereafter:

$$Q_t^\pi(b,a) \triangleq \rho(b,a) + \sum_{z\in\Omega}\Pr(z|a,b)V_{t-1}^\pi(b_z^a). \tag{2.5}$$

The value function corresponding to the optimal policy is called the *optimal value func-*

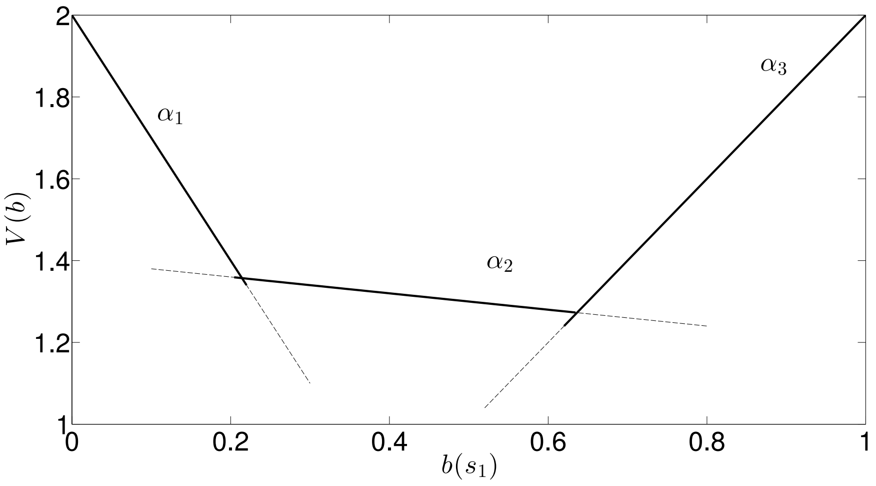


Figure 2.1: Illustration of the PWLC property of the value function for a 2-state POMDP. The value function is the upper surface indicated by the solid lines.

tion V_t^*. The *optimal value function* $V_t^*(b)$ can be characterized recursively as:

$$V_t^*(b) = \max_a \left[\rho(b,a) + \sum_{z \in \Omega} \Pr(z|a,b) V_{t-1}^*(b_z^a) \right]. \tag{2.6}$$

We can also define the *Bellman optimality operator* $\mathfrak{B}^*$:

$$(\mathfrak{B}^* V_{t-1})(b) = \max_a [\rho(b,a) + \sum_{z \in \Omega} \Pr(z|a,b) V_{t-1}(b_z^a)],$$

and write (2.6) as $V_t^*(b) = (\mathfrak{B}^* V_{t-1}^*)(b)$.

An important consequence of these equations is that the t-step optimal value function (V_t^*) is *piecewise-linear and convex* (PWLC), as shown in Figure 2.1, a property exploited by most POMDP planners. Sondik [1971] showed that a PWLC value function at any finite time step t can be expressed as a set of vectors: $\Gamma_t = \{\alpha_0, \alpha_1, \ldots, \alpha_m\}$. Each α_i represents an $|S|$-dimensional hyperplane defining the value function over a bounded region of the belief space. The value of a given belief point can be computed from the vectors as: $V_t^*(b) = \max_{\alpha_i \in \Gamma_t} \sum_s b(s) \alpha_i(s)$.

2.2 POMDP Planning Methods

POMDP planning methods [White, 1991, Shani et al., 2012] aim to compute the optimal value function or the optimal policy given the POMDP model of the world. Broadly, planning can be divided into two categories: *offline* and *online* planning. Offline planning methods compute the optimal policy for the entire belief space before the agent actually executes the task. The agent when executing the task uses the optimal policy to find the optimal action as it encounters a belief. Online planning circumvents the complexity of computing the optimal policy for all possible beliefs beforehand, instead the agent plans *online* while executing the task only for the beliefs that it encounters.

In this subsection we given a brief introduction to the offline planning methods for POMDPs divided into two categories: *exact methods* and *point-based methods*.

2.2.1 Exact Methods

Exact methods [Monahan, 1982] compute the value function for all possible belief points by computing the optimal $\mathbf{\Gamma}_t$ using the following recursive algorithm. For each action a and observation z, an intermediate $\Gamma_t^{a,z}$ set of backprojection vectors $\alpha_i^{a,z}$ is computed from Γ_{t-1}:

$$\Gamma_t^{a,z} = \{\alpha_i^{a,z} \ : \ \alpha_i \in \Gamma_{t-1}\}, \tag{2.7}$$

where for all $s \in S$,

$$\alpha_i^{a,z}(s) = \sum_{s' \in S} T(s,a,s')O(s',a,z)\alpha_i(s'). \tag{2.8}$$

The next step is to take a cross-sum[1] over $\Gamma_t^{a,z}$ sets.

$$\Gamma_t^a = \{R(s,a)\} \oplus \Gamma_t^{a,z_1} \oplus \Gamma_t^{a,z_2} \oplus \dots \tag{2.10}$$

Then we take union of all the Γ_t^a-sets and prune any dominated α-vectors:

$$\mathbf{\Gamma}_t = \texttt{prune}(\cup_{a\in A}\Gamma_t^a) \tag{2.11}$$

The above algorithm can compute the optimal value function of a POMDP exactly given the time horizon h. The computational complexity of each iteration of

[1]The cross-sum of two sets A and B contains all the values resulting from summing one element from each set:

$$A \oplus B = \{\mathbf{a}+\mathbf{b} : \mathbf{a} \in A \wedge \mathbf{b} \in B\} \tag{2.9}$$

this algorithm, that is computing $\mathbf{\Gamma}_t$ from $\mathbf{\Gamma}_{t-1}$ is $\mathcal{O}(|S|^2|A||\Gamma_{t-1}|^{|\Omega|})$, since there are $\mathcal{O}(|A||\Gamma_{t-1}|^{|\Omega|})$ cross-sums required to compute Γ_t^a [Pineau et al., 2006]. Example of other methods that aim to compute the value function exactly are Sondik's One-pass algorithm [Sondik, 1971, Smallwood and Sondik, 1973], Cheng's Linear Support [Cheng, 1988] and Witness algorithm [Kaelbling et al., 1998]. However, the high computational cost of these methods for large POMDPs limits their application in practice.

2.2.2 Point-Based Methods

Point-based planners [Spaan and Vlassis, 2005, Pineau et al., 2006, Shani et al., 2012] avoid the expense of solving for all belief points by computing Γ_t, an approximate optimal set, only for a set of sampled beliefs B. Point-based methods compute Γ_t using the following recursive algorithm.

At each iteration (starting from t = 1), for each action a and observation z, an intermediate $\Gamma_t^{a,z}$, a set of backprojection vectors $\alpha_i^{a,z}$, is computed from Γ_{t-1}:

$$\Gamma_t^{a,z} = \{\alpha_i^{a,z} : \alpha_i \in \Gamma_{t-1}\}, \tag{2.12}$$

Next, Γ_t^a is computed only for the sampled beliefs, i.e., $\Gamma_t^a = \{\alpha_b^a : b \in B\}$, where:

$$\alpha_b^a = R(s,a) + \sum_{z\in\Omega} \operatorname*{arg\,max}_{\alpha\in\Gamma_t^{a,z}} \sum_s b(s)\alpha(s). \tag{2.13}$$

Finally, the best α-vector for each $b \in B$ is selected:

$$\alpha_b = \operatorname*{arg\,max}_{\alpha_b^a} \sum_s b(s)\alpha_b^a(s), \tag{2.14}$$

$$\Gamma_t = \cup_{b\in B}\alpha_b. \tag{2.15}$$

The above algorithm at each timestep t, generates $|A||\Omega||\Gamma_{t-1}|$ alpha vectors in $\mathcal{O}(|S|^2|A||\Omega||\Gamma_{t-1}|)$ time and then reduces them to $|B|$ vectors in $\mathcal{O}(|S||B||A||\Omega||\Gamma_{t-1}|)$ [Pineau et al., 2006]. The success of point-based methods in many practical tasks [Spaan and Vlassis, 2005, Sridharan et al., 2010, Shani et al., 2012] makes them an attractive starting point for solving the sensor selection task.

In this section we described the POMDP framework and the existing planning methods for POMDPs that can compute the optimal policy given the POMDP model of the world. In the next section we describe the main components of the active perception POMDP for modelling the sensor selection task.

2.3 Active Perception POMDP

The goal in an active perception POMDP is to reduce the uncertainty about a *feature of interest* that is not directly observable. In general, the feature of interest may be only a part of the state, e.g., if a surveillance system cares only about the people's positions, not their velocities, or higher-level features derived from the state. However, for simplicity and without the loss of generality, we focus on the case where the feature of interest is just the state. As it will be clear from the following chapters none of our results require this assumption. For simplicity, we also focus on *pure* active perception tasks in which the agent's only goal is to reduce the uncertainty about the state, as opposed to hybrid tasks where the agent may also have other goals. For such cases, *hybrid* rewards [Eck and Soh, 2012], which combine the advantage of belief-based and state-based rewards, are appropriate. While we do not cover the hybrid rewards in this thesis, extending the results presented in this thesis to hybrid rewards is pretty straightforward as we show later in Chapter 3.

We model the sensor selection task as an active perception POMDP in which an agent must choose a subset of available sensors at each time step. We assume that all the selected sensors must be chosen simultaneously, i.e., it is not possible within a timestep to condition the choice of one sensor on the observations generated by another sensor. This corresponds to the common setting where generating each sensor's observation is time consuming, e.g., in the surveillance task, because it requires applying expensive computer vision algorithms, and thus all observations from the selected cameras must be generated in parallel. Formally, an active perception POMDP has the following components:

- Actions $\mathbf{a} = \langle a_1, \ldots, a_n \rangle$ are vectors of n binary *action features*, each of which specifies whether a given sensor is selected or not. For each $\mathbf{a}$, we also define its set equivalent $\mathcal{A} = \{i : a_i = 1\}$, i.e., the set of indices of the selected sensors. Due to the resource constraints, the set of all actions $\mathcal{A}^+ = \{\mathcal{A} : |\mathcal{A}| \leq k\}$ contains only sensor subsets of size k or less. $\mathcal{X} = \{1, \ldots, n\}$ indicates the set of all sensors.

- Observations $\mathbf{z} = \langle z_1, \ldots, z_n \rangle$ are vectors of n *observation features* [2], each of which specifies the sensor reading obtained by the given sensor. If sensor i is not selected, then $z_i = \emptyset$. The set equivalent of $\mathbf{z}$ is $\mathcal{Z} = \{z_i : z_i \neq \emptyset\}$. To prevent ambiguity about which sensor generated which observation in $\mathcal{Z}$, we assume that, for all i and j, the domains of z_i and z_j share only $\emptyset$. This assumption is only made

[2]Each of these features can be a vector in itself.

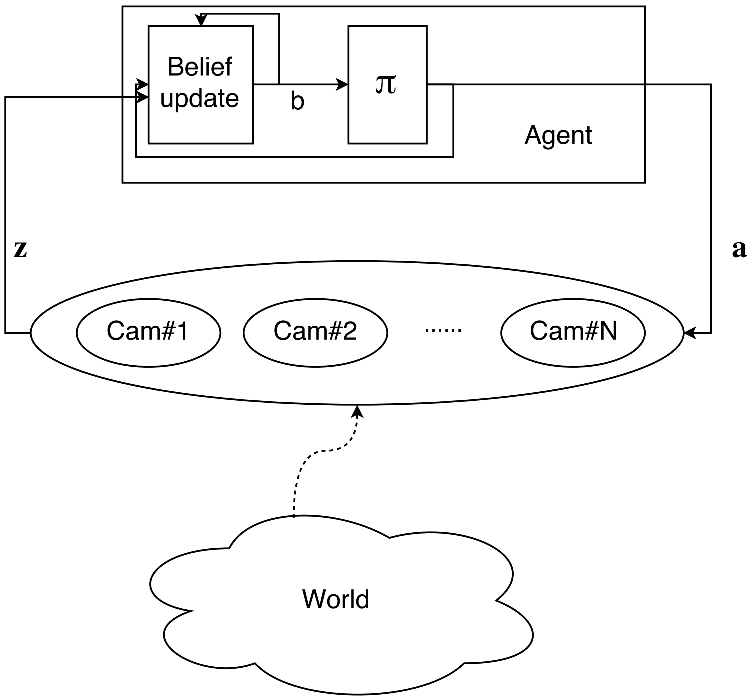


Figure 2.2: Model for sensor selection problem

for notational convenience and does not restrict the applicability of our methods in any way.

For example, in the surveillance task, $\mathcal{A}$ indicates the set of cameras that are active and $\mathcal{Z}$ are the observations received from the cameras in $\mathcal{A}$. The model for the sensor selection problem for the surveillance task is shown in Figure 2.2. Here, we assume that actions involve only selecting k out of the n available sensors. The transition function is thus independent of the actions, as selecting sensors cannot change the state. However, as we outline later in Chapter 4, subsection 4.2, it is possible to extend our results to arbitrary transition functions, that can model, e.g., mobile sensors that, by moving, change the state.

A challenge in these settings is properly formalizing the reward function. Because the goal is to reduce the uncertainty, reward is a direct function of the belief, not the state, i.e., the agent has no preference for one state over another, so long as it knows what that state is. Hence, there is no meaningful way to define a state-based reward function

$R(s, \mathbf{a})$. Directly defining $\rho(b, \mathbf{a})$ using, e.g., negative *belief entropy*:

$$\rho(b, \mathbf{a}) = -H_b(s) = \sum_s b(s) \log(b(s)), \tag{2.16}$$

results in a value function that is not piecewise-linear. Since $\rho(b, \mathbf{a})$ is no longer a convex combination of a state-based reward function, it is no longer guaranteed to be PWLC, a property most POMDP solvers rely on. In the following subsections, we describe two recently proposed frameworks designed to address this problem.

2.3.1 ρPOMDPs

A ρPOMDP [Araya-lópez et al., 2010], defined by a tuple $\langle S, A, T, \Omega, O, \Gamma_\rho, b_0, h \rangle$, is a standard POMDP except that the state-based reward function $R(s, \mathbf{a})$ has been omitted and Γ_ρ has been added. Γ_ρ is a set of vectors, that defines the immediate reward for ρPOMDP. Since we consider only pure active perception tasks, ρ depends only on b, not on $\mathbf{a}$ and can be written as $\rho(b)$. Given Γ_ρ, $\rho(b)$ can be defined as: $\rho(b) \triangleq \max_{\alpha \in \Gamma_\rho} \sum_s b(s)\alpha(s)$. If the true reward function is not PWLC, e.g., negative belief entropy, it can be approximated by defining Γ_ρ as a set of vectors, each of which is a tangent to the true reward function. Figure 2.3 illustrates approximating negative belief entropy with different numbers of tangents. Solving a ρPOMDP requires a minor change to the existing algorithms[3]. In particular, since Γ_ρ is a set of vectors, instead of a single vector, an additional cross-sum is required to compute $\Gamma_t^{\mathbf{a}}$: $\Gamma_t^{\mathbf{a}} = \Gamma_\rho \oplus \Gamma_t^{\mathbf{a},\mathbf{z}_1} \oplus \Gamma_t^{\mathbf{a},\mathbf{z}_2} \oplus \ldots$.

Araya-lópez et al. [2010] showed that the error in the value function computed by this approach, relative to the true reward function, whose tangents were used to define Γ_ρ, is bounded. However, the additional cross-sum increases the computational complexity of computing $\Gamma_t^{\mathbf{a}}$ in each iteration to $\mathcal{O}(|S||A||\Gamma_{t-1}||\Omega||B||\Gamma_\rho|)$ with point-based methods.

Though the original definition ρPOMDP do not put any constraints on the definition of ρ, we restrict the definition of ρ for an active perception POMDP to be a set of vectors ensuring that ρ is PWLC, which in turn ensures that the value function is PWLC. This is not a severe restriction because solving a ρPOMDP using *offline planning* requires a PWLC approximation of ρ anyway (since most POMDP planning methods require a PWLC value function to solve the POMDP).

[3]Arguably, there is a counter-intuitive relation between the general class of POMDPs and the sub-class of pure active perception problems: on the one hand, the class of POMDPs is a more general set of problems, and it is intuitive to assume that there might be harder problems in the class. On the other hand, many POMDP problems admit a representation of the value function using a finite set of vectors. In contrast, the use of entropy would require an infinite number of vectors to merely represent the reward function. Therefore, even though we consider a specific sub-class of POMDPs, this class has properties that make it difficult to address using existing methods.

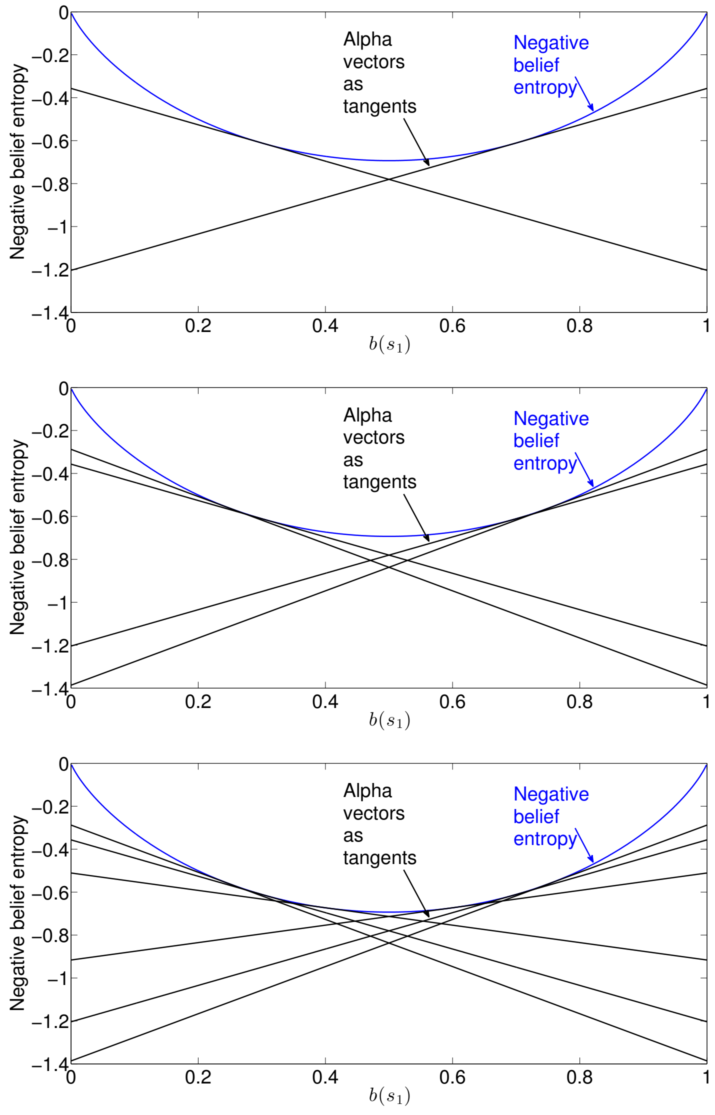


Figure 2.3: Defining Γ_ρ^a with different sets of tangents to the negative belief entropy curve in a 2-state POMDP.

2.3.2 POMDPs with Information Rewards

Spaan et al. [2015] proposed *POMDPs with information rewards* (POMDP-IR), an alternative framework for modelling active perception tasks that relies only on the standard POMDP framework. Instead of directly rewarding low uncertainty in the belief, the agent is given the chance to make predictions about the hidden state and rewarded, via a standard state-based reward function, for making accurate predictions. Formally, a POMDP-IR is a POMDP in which each action $\mathrm{a} \in A$ is a tuple $\langle \mathbf{a}_n, a_p \rangle$ where $\mathbf{a}_n \in A_n$ (A_n is set of all possible $\mathbf{a}_n$) is a *normal action*, e.g., moving a robot or turning on a camera (in our case $\mathbf{a}_n$ is $\mathbf{a}$), and $a_p \in A_p$ (A_p is set of all possible a_p) is a *prediction action*, which expresses predictions about the state. The joint action space is thus the Cartesian product of A_n and A_p, i.e., $A = A_n \times A_p$.

Prediction actions have no effect on states or observations but can trigger rewards via the standard state-based reward function $R(s, \langle \mathbf{a}_n, a_p \rangle)$. While there are many ways to define A_p and R, a simple approach is to create one prediction action for each state, i.e., $A_p = S$, and give the agent positive reward if and only if it correctly predicts the true state:

$$R(s, \langle \mathbf{a}_n, a_p \rangle) = \begin{cases} 1, & \text{if } s = a_p \\ 0, & \text{otherwise.} \end{cases} \tag{2.17}$$

Thus, POMDP-IR indirectly rewards beliefs with low uncertainty, since these enable more accurate predictions and thus more expected reward. Furthermore, since a state-based reward function is explicitly defined, ρ can be defined as a convex combination of R, as in (2.3), guaranteeing a PWLC value function, as in a regular POMDP. Thus, a POMDP-IR can be solved with standard POMDP planners. However, the introduction of prediction actions leads to a blow-up in the size of the joint action space $|A| = |A_n||A_p|$ of POMDP-IR. Replacing $|A|$ with $|A_n||A_p|$ in the complexity analysis of point-based methods yields a computational complexity of $\mathcal{O}(|S||A_n||\Gamma_{t-1}||\Omega||B||A_p| +$ $|S|^2|A_n||A_p||\Omega||\Gamma_{t-1}|)$ for computing Γ_t from Γ_{t-1} for POMDP-IR.

Note that, though not made explicit in Spaan et al. [2015], several independence properties are inherent to the POMDP-IR framework, as shown in Figure 3.2 in Chapter 3. In Chapter 3, we introduce a new technique to show that these independence properties can be exploited to solve a POMDP-IR much more efficiently and thus avoid the blow-up in the size of the action space caused by the prediction actions.

In this section we formulated the active perception POMDP and described ρPOMDP and POMDP-IR, two frameworks that can model uncertainty as an end reward in POMDPs without breaking the PWLC property of the value function. In the next section we introduce submodularity and greedy maximization that we use to propose bounded approximate solutions for the sensor selection task. We also describe the existing variants of greedy maximization that are computationally cheaper than the greedy maximization and the associated error bounds with these algorithms.

2.4 Submodular Function Maximization

Submodularity is a property of set functions that formalizes the notion of *diminishing returns*, i.e., adding an element to a set increases the value of the set function by a smaller or equal amount than adding that same element to a subset as illustrated in the Figure 2.4. Formally, given a ground set $\mathcal{X} = \{1, 2 \ldots n\}$, a set function $F : 2^{\mathcal{X}} \to \mathbb{R}$, is submodular if for every $\mathcal{A}_M \subseteq \mathcal{A}_N \subseteq \mathcal{X}$ and $i \in \mathcal{X} \setminus \mathcal{A}_N$,

$$\Delta_F(i|\mathcal{A}_M) \geq \Delta_F(i|\mathcal{A}_N), \tag{2.18}$$

where $\Delta_F(i|\mathcal{A}) = F(\mathcal{A} \cup i) - F(\mathcal{A})$ is the *marginal gain* of adding i to $\mathcal{A}$. Typically, the aim is to find an $\mathcal{A}^*$ that maximizes F subject to certain constraints. Here, we consider a constraint on $\mathcal{A}^*$'s size: $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} F(\mathcal{A})$. Submodularity is a naturally occurring property in many real-life situations. Consequently, submodular function maximization finds application in many real-world problems, e.g., summarizing text [Takamura and Okumura, 2009, Lin and Bilmes, 2010], selecting subsets of training data for classification [Chen and Krause, 2013], or selecting sensors to minimize uncertainty about a hidden variable [Satsangi et al., 2015a].

Algorithm 1 `greedy-argmax`$(F, \mathcal{X}, k)$

```
$\mathcal{A}^G \leftarrow \emptyset$
**for** $m = 1$ *to* $k$ **do**
    $\mathcal{A}^G \leftarrow \mathcal{A}^G \cup \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}^G} \Delta_F(i|\mathcal{A}^G)$
**end for**
return $\mathcal{A}^G$
```

As n increases, the $\binom{n}{k}$ possibilities for $\mathcal{A}^*$ grow rapidly, rendering naive maximization intractable. Instead, *greedy maximization* finds an approximate solution $\mathcal{A}^G$ faster by iteratively adding to a partial solution the element that maximizes the marginal gain. Given a set function F, greedy maximization computes a subset $\mathcal{A}^G \subseteq \mathcal{X}$ that approximates $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} F(\mathcal{A})$. As shown in Algorithm 1, it does so by repeatedly adding to $\mathcal{A}^G$ the element i that maximizes the marginal gain $\Delta_F(i|\mathcal{A}^G)$. Greedy maximization computes $\mathcal{A}^G$ in $\mathcal{O}(n \times k)$ which is faster than the naive maximization. Nemhauser et al. [1978] showed that, under certain conditions, greedy maximization has

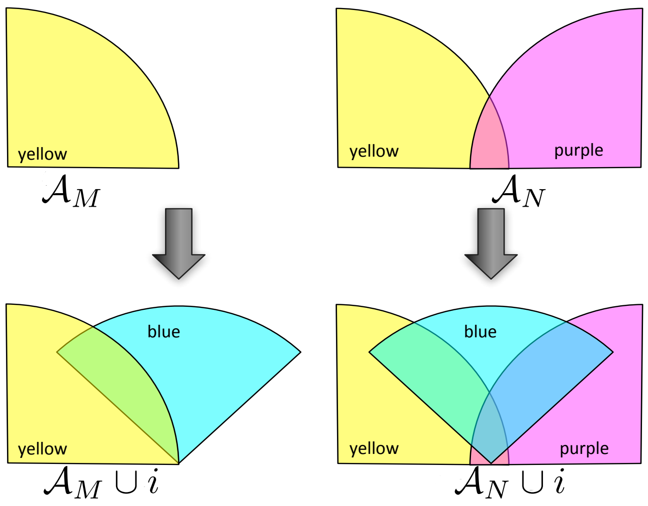


Figure 2.4: Figure illustrating submodularity as the notion of diminishing returns. $\mathcal{A}_M$ is the set of sensor only containing the yellow sensor and $\mathcal{A}_N$ is the set of sensors containing the yellow and the purple sensor. Thus, $\mathcal{A}_M \subset \mathcal{A}_N$ The blue sensor is added to both $\mathcal{A}_M$ and $\mathcal{A}_N$. The increase in the area covered caused by the addition of the blue sensor to $\mathcal{A}_N$ is bounded by the increase in the area covered caused by the addition of the blue sensor to $\mathcal{A}_M$. Thus, illustrating the notion of diminishing returns as adding the blue sensor to a set can only cause the increase in the area covered by the resulting set of sensors by a smaller or equal amount than adding the same blue sensor to a (smaller) subset.

bounded error.

Theorem 1. *[Nemhauser et al., 1978] Let* $\mathcal{X} = \{1, 2, \ldots, n\}$ *and let* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$*. Let* F *be a non-negative, monotone and submodular function* $F : 2^{\mathcal{X}} \to \mathbb{R}_+$*,* $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} F(\mathcal{A})$ *and* $\mathcal{A}^G = \texttt{greedy-argmax}(F, \mathcal{X}, k)$ *, then* $F(\mathcal{A}^G) \geq (1 - e^{-1})F(\mathcal{A}^*)$*.*

Proof. This proof is a constant and important feature of this thesis, thus, we present it here. This proof is taken from Krause and Golovin [2014].

Let $\Delta_F(i|\mathcal{A})$ denote the marginal gain of adding any element i to a set $\mathcal{A}$:

$$\Delta_F(i|\mathcal{A}) = F(\mathcal{A} \cup i) - F(\mathcal{A}). \tag{2.19}$$

Let $\mathcal{A}^G_m$ denote the greedily build subset of size m, that is,

$$\mathcal{A}^G_m = \texttt{greedy-argmax}(F, \mathcal{X}, m). \tag{2.20}$$

Consequently, by definition, $\mathcal{A}^G = \mathcal{A}^G_k$. Let $\{i^*_1, i^*_2, \ldots, i^*_k\}$ (arbitrary order), be the k elements of $\mathcal{A}^*$, that is,

$$\mathcal{A}^* = \{i^*_1, i^*_2, \ldots, i^*_k\}. \tag{2.21}$$

We start with the following set of inequalities and then provide explanation for them:

$$\begin{aligned}
F(\mathcal{A}^*) &\leq F(\mathcal{A}^* \cup \mathcal{A}^G_m) && (2.22)\\
&= F(\mathcal{A}^G_m) + \sum_{j=1}^{k} \Delta_F(i^*_j | \mathcal{A}^G_m \cup \{i^*_1, i^*_2, \ldots, i^*_{j-1}\}) && (2.23)\\
&\leq F(\mathcal{A}^G_m) + \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}^G_m} \Delta_F(i|\mathcal{A}^G_m) && (2.24)\\
&\leq F(\mathcal{A}^G_m) + \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}^G_m} (F(\mathcal{A}^G_{m+1}) - F(\mathcal{A}^G_m)) && (2.25)\\
&\leq F(\mathcal{A}^G_m) + k(F(\mathcal{A}^G_{m+1}) - F(\mathcal{A}^G_m)). && (2.26)
\end{aligned}$$

Eq (2.22) follows from monotonicity of F. Eq (2.23) is a straightforward telescopic sum. Eq (2.24) is true because F is submodular, which implies that:

$$\Delta_F(i^*_j | \mathcal{A}^G_m \cup \{i^*_1, i^*_2, \ldots, i^*_{j-1}\}) \leq \Delta_F(i^*_j | \mathcal{A}^G_m), \tag{2.27}$$

is true for all $1 \leq j \leq k$[4]. Thus,

$$\sum_{j=1}^{k} \Delta_F(i_j^*|\mathcal{A}_m^G \cup \{i_1^*, i_2^*, \ldots, i_{j-1}^*\}) \leq \sum_{j=1}^{k} \Delta_F(i_j^*|\mathcal{A}_m^G) = \sum_{i\in\mathcal{A}^*} \Delta_F(i|\mathcal{A}_m^G). \tag{2.28}$$

$\sum_{i\in\mathcal{A}^*} \Delta_F(i|\mathcal{A}_m^G)$ is same as $\sum_{i\in\mathcal{A}^*\setminus\mathcal{A}_m^G} \Delta_F(i|\mathcal{A}_m^G)$ because for any i in $\mathcal{A}^* \cap \mathcal{A}_m^G$, $\Delta_F(i|\mathcal{A}_m^G) = F(\mathcal{A}_m^G \cup i) - F(\mathcal{A}_m^G) = F(\mathcal{A}_m^G) - F(\mathcal{A}_m^G) = 0$.

Eq (2.25) holds because $\mathcal{A}_{m+1}^G$ is built greedily from $\mathcal{A}_m^G$ by adding the element that maximizes $\Delta_F(i|\mathcal{A}_m^G)$. Eq (2.26) is true because $|\mathcal{A}^* \setminus \mathcal{A}_m^G| \leq k$.

Thus,

$$F(\mathcal{A}^*) - F(\mathcal{A}_m^G) \leq k(F(\mathcal{A}_{m+1}^G) - F(\mathcal{A}_m^G)) \tag{2.29}$$
$$= k(F(\mathcal{A}^*) - F(\mathcal{A}_m^G) - (F(\mathcal{A}^*) - F(\mathcal{A}_{m+1}^G))). \tag{2.30}$$

Lets define $\beta_m = F(\mathcal{A}^*) - F(\mathcal{A}_m^G)$ then Eq (2.30) can be written as:

$$\beta_m \leq k(\beta_m - \beta_{m+1}). \tag{2.31}$$

This implies,

$$\beta_{m+1} \leq (1 - \frac{1}{k})\beta_m. \tag{2.32}$$

Applying the above equation for $m = 0$ to $k - 1$,

$$\beta_k \leq (1 - \frac{1}{k})^k \beta_0. \tag{2.33}$$

$\beta_0 = F(\mathcal{A}^*) - F(\emptyset) \leq F(\mathcal{A}^*)$ since F is non-negative.

$$\beta_k \leq (1 - \frac{1}{k})^k \beta_0 \leq (1 - \frac{1}{k})^k F(\mathcal{A}^*). \tag{2.34}$$

Since $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$, we have

$$\beta_k \leq e^{\frac{-k}{k}} F(\mathcal{A}^*). \tag{2.35}$$

Since $\beta_k = F(\mathcal{A}^*) - F(\mathcal{A}^G)$,

$$F(\mathcal{A}^G) \geq (1 - e^{-1})F(\mathcal{A}^*). \tag{2.36}$$

□

[4] i_0^* is defined to be empty set.

The above theorem states the $F(\mathcal{A}^G)$ is guaranteed to be more than 63% of $F(\mathcal{A}^*)$, in practice it frequently performs even better [Chen and Krause, 2013, Satsangi et al., 2015a].

Lazy greedy maximization [Minoux, 1978] accelerates greedy maximization by pruning elements whose marginal gain cannot be maximal by maintaining a priority queue of all elements in which each element's priority is its marginal gain computed in the *previous* iteration. If in the current iteration, the marginal gain of the element with the highest priority is higher than the priority of the next element, then the current iteration is terminated since submodularity guarantees that the marginal gain of the remaining elements can only decrease. Lazy greedy maximization computes the same $\mathcal{A}^G$ as greedy maximization and is much faster in practice [Minoux, 1978].

Algorithm 2 `stochastic-greedy-argmax`$(F, \mathcal{X}, k, r)$

```
$\mathcal{A}^S \leftarrow \emptyset$
for $m = 1$ to $k$ do
    $\mathcal{R} \leftarrow$ a subset of size $r$ obtained by sampling elements uniformly randomly from $\mathcal{X} \setminus \mathcal{A}^S$
    $\mathcal{A}^S \leftarrow \mathcal{A}^S \cup \arg\max_{i \in \mathcal{R}} \Delta_F(i|\mathcal{A}^S)$
end for
return $\mathcal{A}^S$
```

Stochastic greedy maximization is even faster than greedy maximization as it samples a subset $\mathcal{R}$ of size r from $\mathcal{X}$ in each iteration of greedy maximization and then selects the element from $\mathcal{R}$ that maximizes the marginal gain. It computes a subset $\mathcal{A}^S$ by adding in each iteration $\arg\max_{i \in \mathcal{R}} \Delta_F(i|\mathcal{A}^S)$, where $\mathcal{R}$ is a subset of $\mathcal{X} \setminus \mathcal{A}^S$ of size r. Mirzasoleiman et al. [2015] showed that stochastic greedy maximization is also guaranteed to have bounded error.

Theorem 2. *[Mirzasoleiman et al., 2015] Let F be a non-negative, monotone and submodular set function, $F : 2^{\mathcal{X}} \to \mathbb{R}_+$, $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} F(\mathcal{A})$ and $\mathcal{A}^S =$* `stochastic-greedy-argmax`$(F, \mathcal{X}, k, r)$ *and let $r = \frac{n}{k} \log(\frac{1}{\epsilon})$ then,*

$$\mathbb{E}[F(\mathcal{A}^S)] \geq (1 - e^{-1} - \epsilon)F(\mathcal{A}^*). \tag{2.37}$$

3 Rewarding Certainty in POMDPs

POMDPs can model the dynamic and partially observable state, which is common to many active perception problems. However, the aim of the agent in active perception tasks is to reduce the uncertainty in its belief. The traditional POMDP framework allows the reward to be expressed only as a function of the state and action and not as a function of the belief. ρPOMDP and POMDP-IR are two frameworks that make it possible to model the reward function in POMDPs that reward agents to reduce the uncertainty in its belief without breaking the PWLC property of the value function. ρPOMDP starts from a "true" belief-based reward function such as the negative entropy and then seeks to find a PWLC approximation via a set of tangents to the curve. By contrast, POMDP-IR starts from the queries that the user of the system will pose, e.g., "What is the position of everyone in the room?" or "How many people are in the room" and creates prediction actions that reward the agent for correctly answering such queries. However, the relative pros and cons of these frameworks are not clear from the existing literature. To the best of our knowledge no previous research has examined the relationship between these two approaches, their respective pros and cons, or their efficacy in realistic tasks. In this chapter, we address this by presenting a theoretical and empirical analysis of ρPOMDP and POMDP-IR.

First, we establish the relationship between these two frameworks by proving the *equivalence* of ρPOMDP and POMDP-IR. By equivalence of ρPOMDP and POMDP-IR, we mean that given a ρPOMDP and a policy, we can construct a corresponding POMDP-IR and a policy such that the value function for both the policies is exactly the same. This equivalence shows that both POMDP-IR and ρPOMDP are equally effective frameworks to model active perception tasks. Furthermore, it shows that any result that holds for ρPOMDP also holds for POMDP-IR and vice-versa. Next, we exploit the independence properties inherent in POMDP-IR to factorize the Bellman optimality equation thereby reducing the computational cost of solving POMDP-IR and ρPOMDP for active perception tasks. We give two algorithms that incorporate the factorization in the exact methods and point-based methods for solving POMDP-IR [Satsangi et al., 2015b, 2017b].

Finally, we provide an extensive empirical analysis to establish the critical factors for the performance of belief-based rewards for the active perception POMDPs on simulated and real-world datasets. We also compare the performance of belief-based rewards to a state based *coverage* reward that is a popular utility function for the sensor selection task. Our results identify the cases where the belief-based rewards dominate the traditional coverage based reward and also discusses the nature of the policies generated by the different reward functions.

3.1 ρPOMDP and POMDP-IR Equivalence

In this section we establish the equivalence between POMDP-IR and ρPOMDP as mentioned before. We show this equivalence by starting with a ρPOMDP and a policy and introducing a *reduction* procedure for both ρPOMDP and the policy (and vice-versa). Using the reduction procedure, we reduce the ρPOMDP to a POMDP-IR and the policy for ρPOMDP to an equivalent policy for POMDP-IR. We then show that the value function for the ρPOMDP we started with and the reduced POMDP-IR is the same for the given and the reduced policy. To complete our proof, we repeat the same process by starting with a POMDP-IR and then reducing it to a ρPOMDP. We show that the value function for the POMDP-IR and the corresponding ρPOMDP is the same.

To start with we first define the reduction procedure for reducing a given ρPOMDP to an 'equivalent' POMDP-IR:

Definition 1. Given a ρPOMDP $\mathbf{M}_\rho = \langle S, A_\rho, \Omega, T_\rho, O_\rho, \Gamma_\rho, b_0, h\rangle$ the REDUCE-POMDP-ρ-IR($\mathbf{M}_\rho$) produces a POMDP-IR $\mathbf{M}_{IR} = \langle S, A_{IR}, \Omega, T_{IR}, O_{IR}, R_{IR}, b_0, h\rangle$ via the following procedure.

- The set of states, set of observations, initial belief and horizon remain unchanged. Since the set of states remain unchanged, the set of all possible beliefs is also the same for $\mathbf{M}_{IR}$ and $\mathbf{M}_\rho$.
- The set of normal actions in $\mathbf{M}_{IR}$ is equal to the set of actions in $\mathbf{M}_\rho$, i.e., $A_{n,IR} = A_\rho$.
- The set of prediction actions $A_{p,IR}$ in $\mathbf{M}_{IR}$ contains one prediction action for each $\alpha_\rho^{a_p} \in \Gamma_\rho$. ($a_p$ on $\alpha_\rho^{a_p}$ denotes the prediction action in $A_{p,IR}$ that $\alpha_\rho^{a_p}$ corresponds

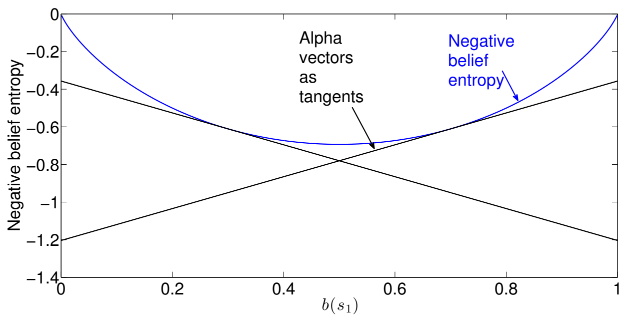


Figure 3.1: Approximating belief-entropy with tangents for a POMDP with 2 states. Each tangent can be viewed as the reward vector drawn by a corresponding prediction action in an equivalent POMDP-IR. Here, the vectors are drawn at points $b(s_1) = 0.3$ and $b(s_1) = 0.7$

.

to.)

- The transition and observation functions in $\mathbf{M}_{IR}$ behave the same as in $\mathbf{M}_\rho$ for each $\mathbf{a}_n$ and ignore the a_p, i.e., for all $\mathbf{a}_n \in A_{n,IR}$: $T_{IR}(s, \mathbf{a}_n, s') = T_\rho(s, \mathbf{a}, s')$ and $O_{IR}(s', \mathbf{a}_n, \mathbf{z}) = O_\rho(s', \mathbf{a}, \mathbf{z})$, where $\mathbf{a} \in A_\rho$ corresponds to $\mathbf{a}_n$ (that is, they are the same, $\mathbf{a} = \mathbf{a}_n$).

- The reward function in $\mathbf{M}_{IR}$ is defined such that $\forall a_p \in A_p$, $R_{IR}(s, a_p) = \alpha_\rho^{a_p}(s)$, where $\alpha_\rho^{a_p}$ is the α-vector corresponding to a_p.

For example, consider a ρPOMDP with 2 states, let ρ be defined using tangents to belief entropy at $b(s_1) = 0.3$ and $b(s_1) = 0.7$. When reduced to a POMDP-IR, the resulting reward function gives a small negative reward for correct predictions and a larger one for incorrect predictions, with the magnitudes determined by the value of the tangents when $b(s_1) = 0$ and $b(s_1) = 1$:

$$R_{IR}(s, a_p) = \begin{cases} -0.35, & \text{if } s = a_p \\ -1.21, & \text{otherwise.} \end{cases} \tag{3.1}$$

This is illustrated in Figure 3.1.

Next we define the reduction procedure for reducing a policy for ρPOMDP to a policy for POMDP-IR.

Definition 2. Given a policy π_ρ for a ρPOMDP, $\mathbf{M}_\rho$, the REDUCE-POLICY-ρ-IR(π_ρ)

procedure produces a policy π_{IR} for a POMDP-IR as follows. For all b,

$$\pi_{IR}(b) = \langle \pi_\rho(b), \arg\max_{a_p} \sum_s b(s)R(s, a_p)\rangle = \langle \pi^n_{IR}(b), \pi^p_{IR}(b)\rangle. \tag{3.2}$$

That is, π_{IR} selects the same normal action as $\pi_\rho(b)(= \pi^n_{IR}(b))$ and the prediction action that maximizes the expected immediate reward.

Using these definitions, we prove that solving $\mathbf{M}_\rho$ is the same as solving $\mathbf{M}_{IR}$.

Theorem 3. *Let* $\mathbf{M}_\rho = \langle S, A_\rho, \Omega, T_\rho, O_\rho, \Gamma_\rho, b_0, h\rangle$ *be a ρPOMDP, and* π_ρ *an arbitrary policy for* $\mathbf{M}_\rho$*. Furthermore let* $\mathbf{M}_{IR}$ = REDUCE-POMDP-ρ-IR$(\mathbf{M}_\rho)$ *and* π_{IR} = REDUCE-POLICY-ρ-IR(π_ρ)*. Then, for all b,*

$$V^{IR}_t(b) = V^\rho_t(b), \tag{3.3}$$

where

$$V^{IR}_t(b) = \max_{a_p} \sum_s b(s)R_{IR}(s, a_p) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi^n_{IR}(b))V^{IR}_{t-1}(b^{\pi^n_{IR}(b)}_{\mathbf{z}}), \tag{3.4}$$

is the t-step value function of policy π_{IR} *in POMDP-IR* $\mathbf{M}_{IR}$ *(let* $V^{IR}_0(b) = \max_{a_p}\sum_s b(s)R_{IR}(s, a_p)$ *for all b) and*

$$V^\rho_t(b) = [\rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_\rho(b))V^\rho_{t-1}(b^{\pi_\rho(b)}_{\mathbf{z}})], \tag{3.5}$$

is the t-step value function of policy π_ρ *in ρPOMDP* $\mathbf{M}_\rho$ *(let* $V^\rho_0(b) = \rho(b) = \max_{\alpha\in\Gamma_\rho}\sum_s b(s)\alpha(s)$ *for all b).*

Proof. Proof in Appendix. □

To complete the proof, we repeat the same procedure for reducing a POMDP-IR to a ρPOMDP:

Definition 3. Given a POMDP-IR $\mathbf{M}_{IR} = \langle S, A_{IR}, \Omega, T_{IR}, O_{IR}, R_{IR}, b_0, h\rangle$ the REDUCE-POMDP-IR-$\rho(\mathbf{M}_{IR})$ produces a ρPOMDP $\mathbf{M}_\rho = \langle S, A_\rho, \Omega, T_\rho, O_\rho, \Gamma_\rho, b_0, h\rangle$ via the following procedure.

- The set of states, set of observations, initial belief and horizon remain unchanged. Since the set of states remain unchanged, the set of all possible belief is also the same for $\mathbf{M}_{IR}$ and $\mathbf{M}_\rho$.

- The set of actions in $\mathbf{M}_\rho$ is equal to the set of normal actions in $\mathbf{M}_{IR}$, i.e., $A_\rho = A_{n,IR}$.
- The transition and observation functions in $\mathbf{M}_\rho$ behave the same as in $\mathbf{M}_{IR}$ for each $\mathbf{a}_n$ and ignore the a_p, i.e., for all $\mathbf{a} \in A_\rho$: $T_\rho(s, \mathbf{a}, s') = T_{IR}(s, \mathbf{a}_n, s')$ and $O_\rho(s', \mathbf{a}, \mathbf{z}) = O_{IR}(s', \mathbf{a}_n, \mathbf{z})$ where $\mathbf{a}_n \in A_{n,IR}$ is the action corresponding to $\mathbf{a} \in A_\rho$ (that is, they are the same $\mathbf{a} = \mathbf{a}_n$).
- The Γ_ρ in $\mathbf{M}_\rho$ is defined such that, for each prediction action in $A_{p,IR}$, there is a corresponding α vector in Γ_ρ, i.e., $\Gamma_\rho = \{\alpha_\rho^{a_p}(s) : \alpha_\rho^{a_p}(s) = R(s, a_p) \text{ for each } a_p \in A_{p,IR}\}$. Consequently, by definition, ρ is defined as: $\rho(b) = \max_{\alpha_\rho^{a_p}} [\sum_s b(s)\alpha_\rho^{a_p}(s)]$.

Definition 4. Given a policy $\pi_{IR}(b) = \langle \pi_{IR}^n(b), \pi_{IR}^p(b) \rangle = \langle \mathbf{a}_n, a_p \rangle$ for a POMDP-IR, $\mathbf{M}_{IR}$, the REDUCE-POLICY-IR-$\rho(\pi_{IR})$ procedure produces a policy π_ρ for a ρPOMDP as follows. For all b,

$$\pi_\rho(b) = \pi_{IR}^n(b). \tag{3.6}$$

That is, π_ρ takes the same action as the normal action in π_{IR}.

Theorem 4. *Let* $\mathbf{M}_{IR} = \langle S, A_{IR}, \Omega, T_{IR}, O_{IR}, R_{IR}, b_0, h \rangle$ *be a POMDP-IR and* $\pi_{IR}(b) = \langle \pi_{IR}^n(b), \pi_{IR}^p(b) \rangle = \langle \mathbf{a}_n, a_p \rangle$ *a policy for* $\mathbf{M}_{IR}$*, such that* $\pi_{IR}^p(b) = a_p = \arg\max_{a'_p} \sum_s b(s) R_{IR}(s, a'_p)$*. Furthermore let* $\mathbf{M}_\rho$ = REDUCE-POMDP-IR-ρ($\mathbf{M}_{IR}$) *and* π_ρ = REDUCE-POLICY-IR-ρ(π_{IR})*. Then, for all b,*

$$V_t^\rho(b) = V_t^{IR}(b), \tag{3.7}$$

where

$$V_t^{IR}(b) = \max_{a_p} \sum_s b(s) R_{IR}(s, a_p) + \sum_{\mathbf{z}} \Pr(\mathbf{z} | b, \pi_{IR}^n(b)) V_{t-1}^{IR}(b_{\mathbf{z}}^{\pi_{IR}^n(b)}), \tag{3.8}$$

is the value of following π_{IR} *in* $\mathbf{M}_{IR}$ *(let* $V_0^{IR}(b) = \max_{a_p} \sum_s b(s) R_{IR}(s, a_p)$ *for all b) and*

$$V_t^\rho(b) = [\rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z} | b, \pi_\rho(b)) V_{t-1}^\rho(b_{\mathbf{z}}^{\pi_\rho(b)})], \tag{3.9}$$

is the value of following π_ρ *in* $\mathbf{M}_\rho$ *(let* $V_0^\rho(b) = \rho(b) = \max_{\alpha \in \Gamma_\rho} \sum_s b(s)\alpha(s)$ *for all b).*

Proof. Proof in Appendix. □

The main implication of these theorems is that any result that holds for either ρPOMDP or POMDP-IR also holds for the other framework. For example, the results presented in Theorem 4.3 in Araya-lópez et al. [2010] that bound the error in the value function of ρPOMDP also hold for POMDP-IR. Furthermore, with this equivalence, the computational complexity of solving ρPOMDP and POMDP-IR comes out to be the same, since POMDP-IR can be converted into ρPOMDP (and vice-versa) trivially, without any significant blow-up in representation. Although we have proved the equivalence of ρPOMDP and POMDP-IR only for pure active perception task where the reward is solely conditioned on the belief, it is straightforward to extend it to hybrid active perception tasks, where the reward is conditioned both on belief and the state. Although, the resulting active perception POMDP for sensor selection is such that the action does not affect the state, the results from this section do not use that property at all and thus are valid for active perception POMDPs where an agent might take an action which can affect the state in the next time step.

In the next section we exploit the independence properties of POMDP-IR framework to solve POMDP-IR efficiently.

3.2 Decomposed Maximization for POMDP-IR

The POMDP-IR framework enables us to formulate uncertainty as an objective, but it does so at the cost of additional computation, as adding prediction actions enlarges the action space. The computational complexity of performing a point-based backup for solving POMDP-IR is $\mathcal{O}(|S|^2|A_n||A_p||\Omega||\Gamma_{t-1}|)+\mathcal{O}(|S||B||A_n||\Gamma_{t-1}||\Omega||A_p|)$. In this section, we present a new technique that exploits the independence properties of the POMDP-IR to reduce the computational costs. We also show that the same principle is applicable to ρPOMDPs.

The increased computational cost of solving POMDP-IR arises from the size of the action space, $|A_n||A_p|$. However, as shown in Figure 3.2, prediction actions only affect the reward function and normal actions only affect the observation and transition function. We exploit this independence to decompose the maximization in the Bellman optimality equation:

$$\begin{aligned} V_t^*(b) &= \max_{\langle \mathbf{a}_n, a_p\rangle \in A} \Big[\sum_s b(s)R(s,a_p) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathbf{a}_n, b) V_{t-1}^*(b_{\mathbf{z}}^{\mathbf{a}_n}) \Big] \\ &= \max_{a_p \in A_p} \sum_s b(s)R(s,a_p) + \max_{\mathbf{a}_n \in A_n} \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathbf{a}_n, b) V_{t-1}^*(b_{\mathbf{z}}^{\mathbf{a}_n}) \end{aligned} \tag{3.10}$$

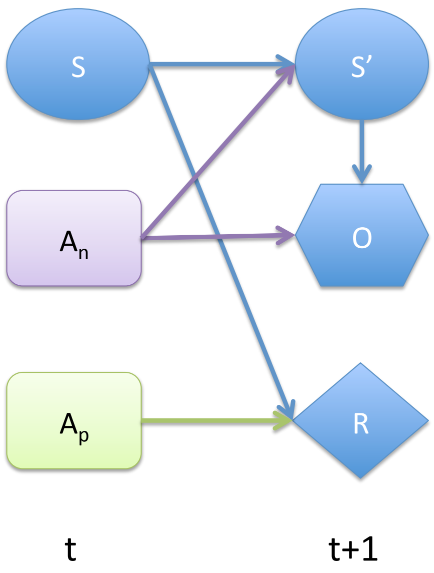


Figure 3.2: Influence diagram for POMDP-IR, highlighting the independence between normal actions and prediction actions.

3.2.1 Exact Methods

Exact methods cannot directly exploit the decomposition as they do not perform an explicit maximization. However, they can be made faster by separating the pruning steps that they employ. First, we generate a set of vectors just for the prediction actions: $\Gamma^R = \{\alpha^{a_p} : a_p \in A_p\}$, where for all $s \in S, \alpha^{a_p}(s) = R(s, a_p)$. Then, we generate another set of vectors for the normal actions, as in the standard exact solver:

$$\begin{aligned}
\Gamma_t^{\mathbf{a}_n,\mathbf{z}} &= \{\alpha_i^{\mathbf{a}_n,\mathbf{z}}(s) : \alpha_i \in \Gamma_{t-1}\}, \\
\alpha_i^{\mathbf{a}_n,\mathbf{z}}(s) &= \sum_{s' \in S} T(s, \mathbf{a}_n, s') O(s', \mathbf{a}_n, \mathbf{z}) \alpha_i(s'), \\
\Gamma_t^{\mathbf{a}_n} &= \Gamma^{\mathbf{a}_n,\mathbf{z}_1} \oplus \Gamma^{\mathbf{a}_n,\mathbf{z}_2} \oplus \dots
\end{aligned} \tag{3.11}$$

Finally, we can compute the optimal set Γ_t as follows:

$$\Gamma_t = \texttt{prune}(\texttt{prune}(\Gamma^R) \oplus \texttt{prune}(\cup_{\mathbf{a}_n \in A_n} \Gamma_t^{\mathbf{a}_n})). \tag{3.12}$$

This is essentially a special case of *incremental pruning* [Cassandra et al., 1997] made possible due to the special structure of POMDP-IR. The independence properties enable the normal and the prediction actions to be treated separately. Thus, in turn allow us to prune Γ^R and $\Gamma_t^{\mathbf{a}_n}$ separately resulting in faster pruning (because of smaller size) and hence faster computation of the final Γ-set.

3.2.2 Point-based Methods

Point-based methods can exploit the decomposition by computing $\Gamma_t^{a,z}$ only for normal actions, $\mathbf{a}_n$ and α^{a_p} only for prediction actions. That is, (2.12) can be changed to:

$$\Gamma_t^{\mathbf{a}_n,\mathbf{z}} = \{\alpha_i^{\mathbf{a}_n,\mathbf{z}} : \alpha_i \in \Gamma_{t-1}\}. \tag{3.13}$$

For each prediction action, we compute the vector specifying the immediate reward for performing the prediction action in each state: $\Gamma^{A_p} = \{\alpha^{a_p}\}$, where $\alpha^{a_p}(s) = R(s, a_p)$ $\forall\ a_p \in A_p$. The next step is to modify (2.13) to separately compute the vectors maximizing expected reward induced by prediction actions and the expected return induced by the normal action:

$$\alpha_b^{\mathbf{a}_n} = \operatorname*{arg\,max}_{\alpha^{a_p} \in \Gamma^{A_p}} \sum_s b(s)\alpha^{a_p}(s) + \sum_{\mathbf{z}} \operatorname*{arg\,max}_{\alpha^{\mathbf{a}_n,\mathbf{z}} \in \Gamma_t^{\mathbf{a}_n,\mathbf{z}}} \sum_s \alpha^{\mathbf{a}_n,\mathbf{z}}(s)b(s).$$

By decomposing the maximization, this approach avoids iterating over all $|A_n||A_p|$ joint actions. At each timestep t, this approach generates $|A_n||\Omega||\Gamma_{t-1}| + |A_p|$ backprojections in $\mathcal{O}(|S|^2|A_n||\Omega||\Gamma_{t-1}| + |S||A_p|)$ time and then prunes them to $|B|$ vectors, with a computational complexity of $\mathcal{O}(|S||B|(|A_p| + |A_n||\Gamma_{t-1}||\Omega|))$.

The same principle can be applied to ρPOMDP by changing (2.13) such that it maximizes over immediate reward independently from future return:

$$\alpha_b^{\mathbf{a}} = \operatorname*{arg\,max}_{\alpha^\rho \in \Gamma_\rho} \sum_s b(s)\alpha^\rho(s) + \sum_{\mathbf{z}} \operatorname*{arg\,max}_{\alpha^{\mathbf{a},\mathbf{z}} \in \Gamma_t^{\mathbf{a},\mathbf{z}}} \sum_s \alpha^{\mathbf{a},\mathbf{z}}(s)b(s). \tag{3.14}$$

The computational complexity of solving ρPOMDP with this approach is $\mathcal{O}(|S|^2|A||\Omega||\Gamma_{t-1}| + |S||\Gamma_\rho|) + \mathcal{O}(|S||B|(|\Gamma_\rho| + |A||\Gamma_{t-1}||\Omega|)$. Thus, even though both POMDP-IR and ρPOMDP use extra actions or vectors to formulate belief-based rewards, they can both be solved at only minimal additional computational cost.

3.3 Experiments

In this section, we present an analysis of the behaviour and performance of belief-based rewards for active perception tasks. We present the results of experiments designed to

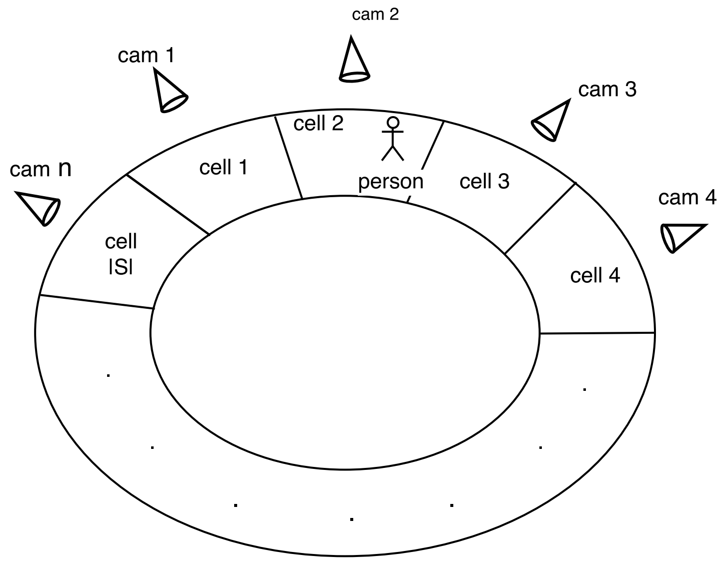


Figure 3.3: Problem setup for the task of tracking one person. We model this task as a POMDP with one state for each cell. Thus the person can move among $|S|$ cells. Each cell is adjacent to two other cells and each cell is monitored by a single camera. Thus, in this case there are $n = |S|$ cameras. The agent must select k out of n cameras and the task is to predict the state of the person correctly using noisy observations from the k cameras. There is one prediction action for each state and the agent gets a reward of $+1$ if it correctly predicts the state and 0 otherwise. At each time step the agent can stay in the same cell with a probability p or move to one of the neighboring cells with equal probability. An observation is a vector of n *observation features*, each of which specifies the person's position as estimated by the given camera. If a camera is not selected, then the corresponding observation feature has a value of null.

study the effect on performance of the choice of prediction actions/tangents, and compare the costs and benefits of myopic versus non-myopic planning. We consider the task of tracking people in a surveillance area with a multi-camera tracking system as shown in Figure 3.3. The goal of the system is to select a subset of cameras, to correctly predict the position of people in the surveillance area, based on the observations received from the selected cameras. In the following subsections, we present results on real-data collected from a multi-camera system in a shopping mall..

We compare the performance of POMDP-IR with decomposed maximization to a naive POMDP-IR that does not decompose the maximization. Thanks to Theorems 3 and 4, these approaches have performance equivalent to their ρPOMDP counterparts. We also compare against two baselines. The first is a weak baseline we call the *rotate policy* in which the agent simply keeps switching between cameras on a turn-by-turn basis. The second is a stronger baseline we call the *coverage policy*, which was developed in earlier work on active perception [Spaan, 2008, Spaan and Lima, 2009]. The coverage policy is obtained after solving a POMDP that rewards the agent for observing the person, i.e., the agent is encouraged to select the cameras that are most likely to generate positive observations. Thanks to the decomposed maximization, the computational cost of solving for the coverage policy and belief-based rewards is the same.

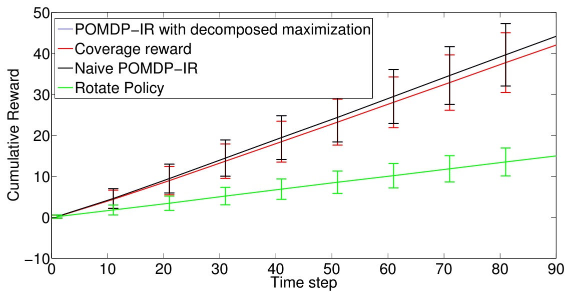
3.3.1 Simulated Setting

We start with experiments conducted in a simulated setting, first considering the task of tracking a single person with a multi-camera system and then considering the more challenging task of tracking multiple people.

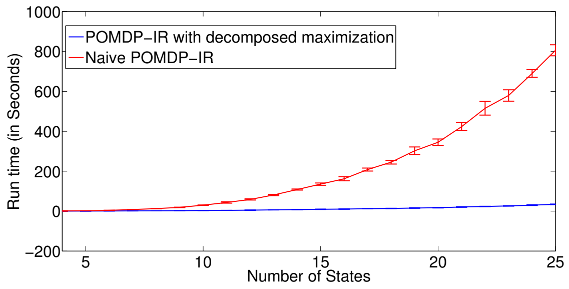
Single-Person Tracking

We start by considering the task of tracking one person walking in a grid-world composed of $|S|$ cells and n cameras as shown in Figure 3.3. At each timestep, the agent can select only k cameras, where $k \leq n$. Each selected camera generates a noisy observation of the person's location. The agent's goal is to minimize its uncertainty about the person's position. In the experiments in this section, we fixed $k = 1$ and $n = 10$. The problem setup and the POMDP model is shown and described in Figure 3.3.

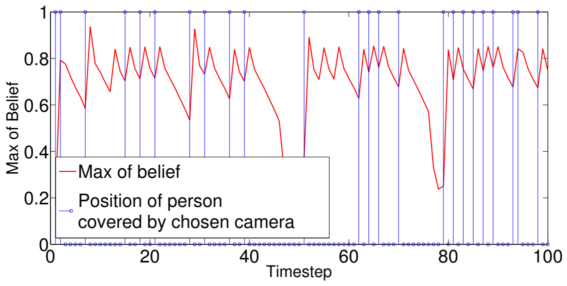
To compare the performance of POMDP-IR to the baselines, 100 trajectories were simulated from the POMDP. The agent was asked to guess the person's position at each time step. Figure 3.4(a) shows the cumulative reward collected by all four methods. POMDP-IR with decomposed maximization and naive POMDP-IR perform identically as the lines indicating their respective performance lie on top of each other in figure



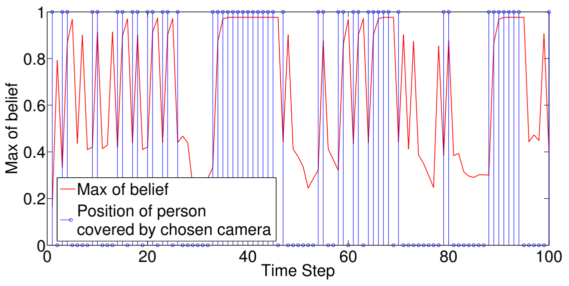
(a)



(b)



(c)



(d)

Figure 3.4: (a) Performance comparison between POMDP-IR with decomposed maximization, naive POMDP-IR, coverage policy, and rotate policy; (b) Runtime comparison between POMDP-IR with decomposed maximization and naive POMDP-IR; (c) Behaviour of POMDP-IR policy; (d) Behaviour of coverage policy. (Best viewed in color.)

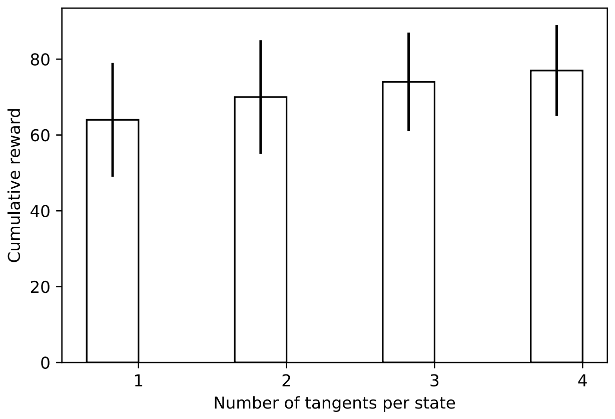
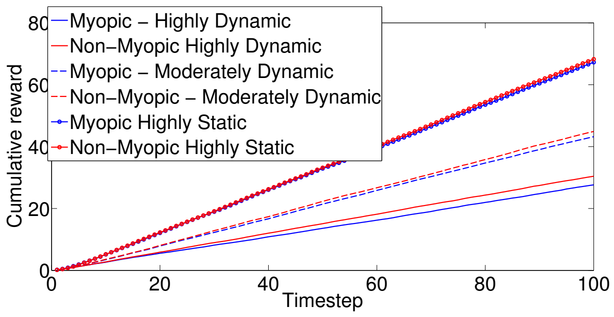


Figure 3.5: Performance comparison as negative belief entropy is better approximated.

3.4(a). However, Figure 3.4(b), which compares the runtimes of POMDP-IR with decomposed maximization and naive POMDP-IR, shows that decomposed maximization yields a large computational savings. Figure 3.4(a) also shows that POMDP-IR greatly outperforms the rotate policy and modestly outperforms the coverage policy.

Figures 3.4(c) and 3.4(d) illustrate the qualitative difference between POMDP-IR and the coverage policy. The blue lines mark the points in trajectory when the agent selected the camera that observes the person's location. If the agent selected a camera such that the person's location is not covered then the blue vertical line is not there at that point in the trajectory in the figure. The agent has to select one out of n cameras and does not have an option of not selecting any camera. The red line plots the max of the agent's belief. The main difference between the two policies is that once POMDP-IR gets a good estimate of the state, it proactively observes neighbouring cells to which the person might transition. This helps it to more quickly find the person when she moves. By contrast, the coverage policy always looks at the cell where it believes her to be. Hence, it takes longer to find her again when she moves. This is evidenced by the fluctuations in the max of the belief, which often drops below 0.5 for the coverage policy but rarely does so for POMDP-IR. The presence of false positives and negatives can also be seen in the figure, when max of the belief goes down even though the agent selected the camera which can observe the person's location and in some cases even though the agent did not select the camera which can observe the person's location but still the max of belief shoots up.

Next, we examine the effect of approximating a true reward function like belief en-



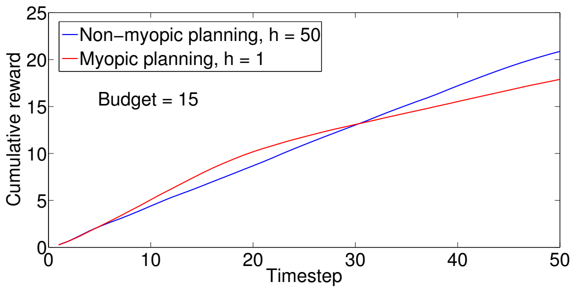


Figure 3.6: (top) Performance comparison for myopic vs. non myopic policies; (bottom) Performance comparison for myopic vs non myopic policies in budget-based setting.

tropy with more and more tangents. Figure 2.3 illustrates how adding more tangents can better approximate negative belief entropy. To test the effects of this, we measured the cumulative reward when using between one and four tangents per state. Figure 3.5 shows the results and demonstrates that, as more tangents are added, the performance in terms of the true reward function improves. However, performance also quickly saturates, as four tangents perform no better than three.

Next, we compare the performance of POMDP-IR to a myopic variant that seeks only to maximize immediate reward, i.e., $h = 1$. We perform this comparison in three variants of the task. In the *mostly static* variant, the state changes very slowly: the probability of staying is the same state is 0.9. In the *moderately dynamic* variant, the state changes more frequently, with a same-state transition probability of 0.7. In the *highly dynamic* variant, the state changes rapidly (with a same-state transition probability of 0.5). Figure 3.6 (top) shows the results of these comparisons. In each setting, non-myopic POMDP-IR outperforms myopic POMDP-IR. In the highly static variant, the difference is marginal. However, as the task becomes more dynamic, the importance of look-ahead planning grows. Because the myopic planner focuses only on immediate reward, it ignores the increase in uncertainty in its belief when the state changes, which happens more often in dynamic settings.

We also compare the performance of myopic and non-myopic planning in a *budget-constrained* environment. This specifically corresponds to an energy constrained environment, where cameras can be employed only a few times over the entire trajectory. This is augmented with resource constraints, so that the agent has to plan not only when to use the camera, but also decide which camera to select. Specifically, the agent can only employ the multi-camera system a total of 15 times across all 50 timesteps and the agent can select which camera (out of the multi-camera system) to employ at each of the 15 instances. On the other timesteps, it must select an action that generates only a null observation. Figure 3.6 (bottom) shows that non-myopic planning is of critical importance in this setting. Whereas myopic planning greedily consumes the budget as quickly as possible, thus earning more reward in the beginning, non-myopic planning saves the budget for situations in which it is highly uncertain about the state.

Finally, we compare the performance of myopic and non-myopic planning when the multi-camera system can communicate with a mobile robot that also has sensors. This setting is typical of a networked robot system [Spaan et al., 2015] in which a robot coordinates with a multi-camera system to perform surveillance of a building, detect any emergency situations like fire, or help people navigate to their destination. Here, the task is to minimize uncertainty about the location of one person who is moving in the space monitored by the robot and the cameras. The robot's sensors are assumed to be more accurate than the stationary cameras. Specifically, the sensors attached to robot can detect if a person is in the current cell with 90% accuracy compared to the stationary cameras, each of which has an accuracy of 75% of detecting a person in the cell it observes. The robot's sensor can observe the presence or absence of a person only for the cell that the robot occupies. In addition to using its sensors to generate observations about its current cell, the robot can also move forward or backward to an adjacent cell or choose to stay at the current cell. To model this task, the action vector introduced earlier is augmented with another action feature that indicates the direction of the robot's motion, which can take three values: forward, backward or stay.

Performance is quantified as the total number of times the correct location of the person is predicted by the system. Figure 3.7, which shows the performance of myopic and non-myopic policies for this task, demonstrates that non-myopic planning is able to plan and utilize the accurate sensors more effectively than myopic planning.

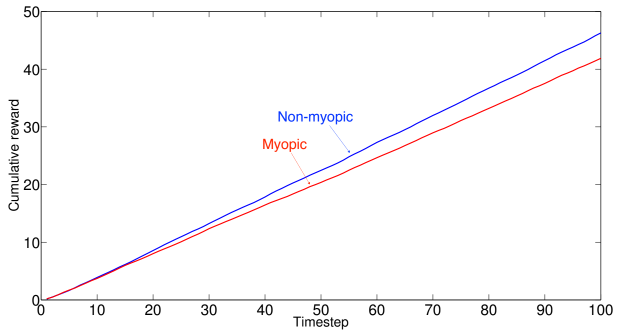


Figure 3.7: Performance comparison for myopic vs. non myopic policies when camera system is assisting a moving robot.

Multi-Person Tracking

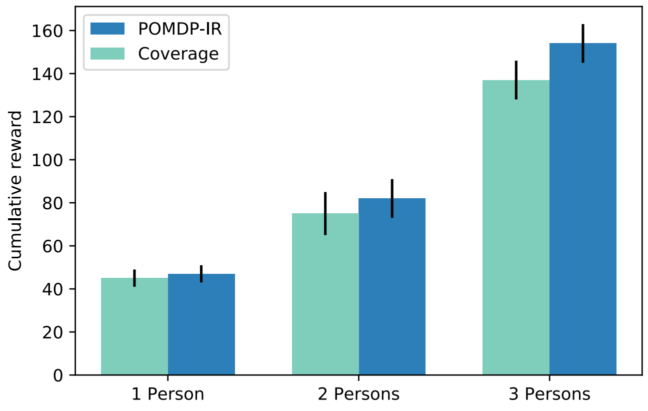
To extend our analysis to a more challenging problem, we consider a simulated setting in which multiple people must be tracked simultaneously. Since $|S|$ grows exponentially in the number of people, the resulting POMDP quickly becomes intractable. Therefore, we compute instead a factored value function

$$V_t(b) = \sum_i V_t^i(b^i), \tag{3.15}$$

where $V_t^i(b^i)$ is the value of the agent's current belief b^i about the i-th person. Thus, $V_t^i(b^i)$ needs to be computed only once, by solving a POMDP of the same size as that in the single-person setting. During action selection, $V_t(b)$ is computed using the current b^i for each person. This kind of factorization corresponds to the assumption that each person's movement and observations is independent of that of other people. Although violated in practice, such an assumption can nonetheless yield good approximations.

Figure 3.8 (top), which compares POMDP-IR to the coverage policy with one, two, and three people, shows that the advantage of POMDP-IR grows substantially as the number of people increases. Whereas POMDP-IR tries to maintain a good estimate of everyone's position, the coverage policy just tries to look at the cells where the maximum number of people might be present, ignoring other cells completely.

Finally, we compare POMDP-IR and the coverage policy in a setting in which the goal is only to reduce uncertainty about a set of "important cells" that are a subset of the whole state space. For POMDP-IR, we prune the set of prediction actions to allow predictions only about important cells. For the coverage policy, we reward the agent only for observing people in important cells. The results, shown in Figure 3.8 (bottom),



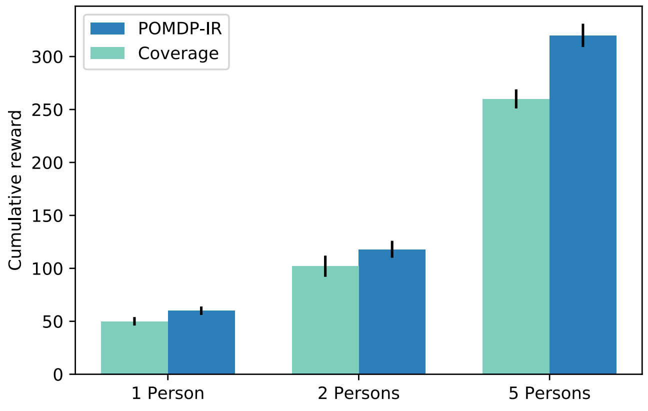


Figure 3.8: (top) Multi-person tracking performance for POMDP-IR and coverage policy; (bottom) Performance of POMDP-IR and coverage policy when only important cells must be tracked.

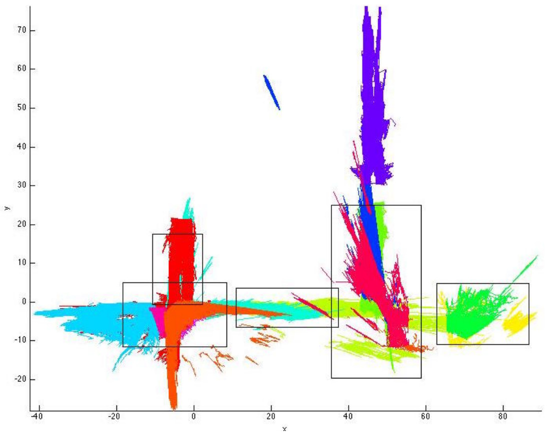


Figure 3.9: Sample tracks for all the cameras. Each color represents all the tracks observed by a given camera. The boxes denote regions of high overlap between cameras.

demonstrate that the advantage of POMDP-IR over the coverage policy is even larger in this variant of the task. POMDP-IR makes use of information coming from cells that neighbour important cells (which is of critical importance if the important cells do not have good observability), while the coverage policy does not. As before, the difference gets larger as the number of people increases.

3.3.2 Real Data

Finally, we extended our analysis to a real-life dataset collected in a shopping mall. This dataset was gathered over 4 hours using 13 CCTV cameras located in a shopping mall [Bouma et al., 2013]. Each camera uses a FPDW [Dollár et al., 2010] pedestrian detector to detect people in each camera image and in-camera tracking [Bouma et al., 2013] to generate tracks of the detected people's movements over time.

The dataset consists of 9915 tracks each specifying one person's x-y position over time. Figure 3.9 shows the sample tracks from all of the cameras.

To learn a POMDP model from the dataset, we divided the continuous space into 20 cells ($|S| = 21$: 20 cells plus an external state indicating the person has left the shopping mall). Using the data, we learned a maximum-likelihood tabular transition function. However, we did not have access to the ground truth of the observed tracks so we constructed them using the overlapping regions of the camera.

Because the cameras have many overlapping regions (see Figure 3.9), we were able to manually match tracks of the same person recorded individually by each camera. The "ground truth" was then constructed by taking a weighted mean of the matched tracks.

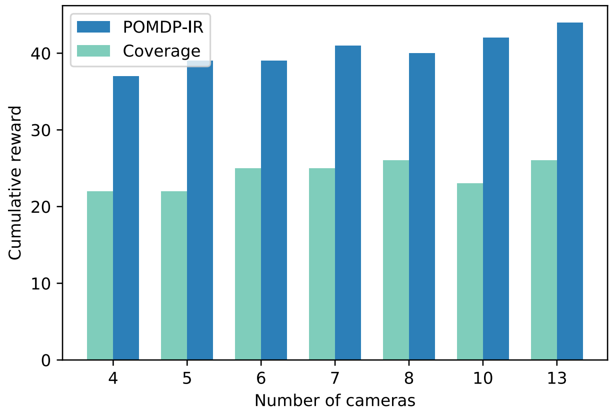


Figure 3.10: Performance of POMDP-IR and the coverage policy on the shopping mall dataset.

Finally, this ground truth was used to estimate noise parameters for each cell (assuming zero-mean Gaussian noise), which was used as the observation function. Figure 3.10 shows that, as before, POMDP-IR substantially outperforms the coverage policy for various numbers of cameras. In addition to the reasons mentioned before, the high overlap between cameras contributes to POMDP-IR's superior performance. The coverage policy has difficulty ascertaining people's exact locations because it is rewarded only for observing them somewhere in a camera's large overlapping region, whereas POMDP-IR is rewarded for deducing their exact locations.

3.4 Conclusions & Future Work

In this chapter we addressed the challenge of modelling active perception problems. Since dynamic state and partial observability are common features of many active perception problems, POMDPs offer a good choice for modelling active perception problems. The main challenge we addressed in this chapter is to express the belief-based rewards in POMDPs without breaking the PWLC property of the value function. ρPOMDP and POMDP-IR are two frameworks that allow formulating uncertainty reduction as an end in itself and do not break the PWLC property. We showed that ρPOMDP and POMDP-IR are two equivalent frameworks for modelling active perception task. Thus, results that apply to one framework are also applicable to the other. Furthermore, we showed that both frameworks admit a decomposition of the maximization performed in the Bellman optimality equation, thereby, avoiding the additional computational cost due to the inflated action space required to model belief-based rewards. Our empirical analysis established the critical factors for good performance for active perception tasks. We showed that a belief-based formulation of uncertainty reduction beats a corresponding popular state-based reward baseline as well as other simple policies. While, the non-myopic policy beats the myopic one, the gain in certain cases is marginal. However, in cases involving mobile sensors and budget constraints, non-myopic policies become critically important.

An interesting point of difference between ρPOMDP and POMDP-IR is in their representation of the belief-based reward. ρPOMDP approximates an explicit function of the belief of the agent with tangent vectors, while POMDP-IR uses additional prediction actions to reward low uncertainty. In cases where the belief of the agent is not explicitly represented as in deep recurrent neural networks [Hausknecht and Stone, 2015], it is not possible to measure the uncertainty in the belief of the agent directly. In Chapter 7 we exploit this insight to propose a deep neural network architecture that enables an agent to reduce its uncertainty without an explicit representation of its belief. Thanks to the equivalence we establish we can be sure that the rewarding low uncertainty in belief of the agent with prediction actions is the same as reducing the entropy of the agent's belief.

In this chapter we considered only the pure active perception tasks where the agent is rewarded solely for reducing the uncertainty in its belief. An interesting line of future work would be to extend and experiment with hybrid tasks where the agent can serve multiple-objectives, for example, maintaining surveillance at an airport while helping the people at the airport navigate around. A big challenge for such setting would be to balance the many objectives the agent can have to extract maximum utility.

The experiments in this chapter analyse the performance and behaviour of the belief-based rewards in POMDPs. To this end, we assumed a simple model of the world where the original state space was coarsely discretized, the agent was required to select only one camera out of the n available cameras and simple tabular representations of the transition and observations functions were used. In the next chapters we move to more realistic settings. In Chapter 4 we propose a new POMDP planning method that scales much better in the combinatorial action space of the active perception POMDP when an agent is required to select multiple cameras out of the n available cameras. Chapter 5 focuses on approximating continuous state and observation spaces with sample-based planning. Chapter 6 focusses on real-time online decision making where the agent must select k cameras out of the n available cameras in milliseconds.

4 Point-Based Planning

In the previous chapter we showed that POMDP-IR and ρPOMDP are equivalent frameworks to model the active perception tasks. Thus, now we can directly employ the traditional POMDP planners to compute the optimal value function for the active perception POMDP. While point-based methods scale linearly in the size of the action space of a POMDP, for the active perception POMDP the action space is made up of $\binom{n}{k}$ subsets of sensors, thus, as the number of sensors n grows the computational cost of point-based methods grows exponentially with it, making the use of traditional point-based methods infeasible.

In this chapter we tackle of the problem of scaling the point-based methods to the combinatorial action space of the active perception POMDP. By employing greedy maximization instead of full maximization we propose a new POMDP planning method called greedy point-based value iteration [Satsangi et al., 2017b, 2014]. To analyse the theoretical properties of greedy PBVI we exploit submodularity and show that if the value function of the active perception POMDP possess certain properties, mainly submodularity, then the value function computed using the backups based on greedy maximization is guaranteed to have bounded error with respect to the optimal value function. Furthermore, we establish the sufficient conditions for submodularity of the value functions for the active perception POMDP. One of the sufficient conditions for the value function to be submodular is that the immediate belief-based reward be defined as the negative belief entropy. Since both POMDP-IR and ρPOMDP approximate the negative belief entropy by drawing tangents to the continuous curve, we extend our analysis to show that this tangent-based approximation still keeps the error between the value function computed by greedy PBVI and the optimal value function bounded.

Our definition of the active perception POMDP assumes that the state transition function is independent of the actions of the agent. While this assumption might hold for the sensor selection task, many active perception tasks involve mobile agents that by moving can change the state of the world. Thus, for such active perceptions tasks the transition function is not independent of the actions of the agent. Thus, we extend greedy PBVI and its analysis to these general active perception tasks to show similar error bounds exist even if the agent's action can change the state of the world.

Finally, we model the sensor selection problem as an active perception POMDP and use data recorded by a multi-camera tracking system in a shopping mall to learn the POMDP model of the world. The experiments show that greedy PBVI achieves similar performance to the traditional PBVI but at a fraction of the computational cost leading to better scalability in the action space of the active perception POMDP. In the next sections we introduce greedy PBVI and its analysis, followed by its extension to the general active perception POMDPs and finally the experiments.

4.1 Greedy PBVI

In this section, we propose *greedy PBVI*, a new point-based planner for solving active perception POMDPs which scales better in the size of its action space. To facilitate the explication of greedy PBVI, we present the final step of PBVI, described earlier in (2.13), (2.14) and (2.15), in a different way. For each $b \in B$, and $\mathcal{A} \in \mathcal{A}^+$, we must find the best $\alpha_b^{\mathcal{A}} \in \Gamma_t^{\mathcal{A}}$.

$$\alpha_b^{\mathcal{A},*} = \underset{\alpha_b^{\mathcal{A}} \in \Gamma_t^{\mathcal{A}}}{\arg\max} \sum_s \alpha_b^{\mathcal{A}}(s) b(s), \tag{4.1}$$

and simultaneously record its value $Q(b, \mathcal{A}) = \sum_s \alpha_b^{\mathcal{A},*}(s) b(s)$. Then, for each b we find the best vector across all actions: $\alpha_b = \alpha_b^{\mathcal{A}^*}$, where

$$\mathcal{A}^* = \underset{\mathcal{A} \in \mathcal{A}^+}{\arg\max}\, Q(b, \mathcal{A}). \tag{4.2}$$

The main idea of greedy PBVI is to exploit *greedy maximization* [Nemhauser et al., 1978], which is much faster than full maximization as it avoids going over the $\binom{n}{k}$ choices and instead constructs a subset of k elements iteratively. Thus, we replace the maximization operator in the Bellman optimality equation with greedy maximization. Algorithm 1 in Chapter 2 shows the argmax variant, which constructs a subset $\mathcal{A}^G \subseteq \mathcal{X}$ of size k by iteratively adding sensors from $\mathcal{X}$ to $\mathcal{A}^G$. At each iteration, it adds the element that maximally increases *marginal gain* $\Delta_Q(i|\mathcal{A})$ of adding a sensor i to a subset of sensors

$\mathcal{A}^G$:

$$\Delta_Q(i|\mathcal{A}) = Q(b, i \cup \mathcal{A}) - Q(b, \mathcal{A}). \tag{4.3}$$

To exploit greedy maximization in PBVI, we need to replace the argmax over $\mathcal{A}^+$ with `greedy-argmax`. Our alternative description of PBVI above makes this straightforward: (4.2) contains such an argmax and $Q(b, .)$ has been intentionally formulated to be a set function over $\mathcal{A}^+$. Thus, implementing greedy PBVI requires only replacing (4.2) with:

$$\mathcal{A}^G = \texttt{greedy-argmax}(Q(b, \cdot), \mathcal{A}^+, k). \tag{4.4}$$

Since the complexity of `greedy-argmax` is only $\mathcal{O}(|n||k|)$, the complexity of greedy PBVI is only $\mathcal{O}(|S||B||n||k||\Gamma_{t-1}||\Omega|)$ (for computing $\Gamma_t^{\mathcal{A}}$).

Using point-based methods as a starting point is essential to our approach. Algorithms like Monahan's enumeration algorithm [Monahan, 1982] that rely on pruning operations to compute V^* instead of performing an explicit argmax, cannot directly use `greedy-argmax`. Thus, it is precisely because PBVI operates on a finite set of beliefs that argmax is performed, opening the door to using `greedy-argmax` instead.

In the next subsection we show that the value function computed by greedy PBVI is guaranteed to have bounded error with respect to the optimal value function under certain conditions.

4.1.1 Bounds given submodular value function

In this subsection, we present the theoretical analysis of greedy PBVI, which shows that, under certain conditions, the most important of which is *submodularity*, the error in the value function computed by backups based on greedy maximization is bounded. Later subsections discuss when a reward based on negative belief entropy or an approximation thereof meets those conditions.

Submodularity is a property of set functions that corresponds to diminishing returns, i.e., adding an element to a set increases the value of the set function by a smaller or equal amount than adding that same element to a subset. In our notation, this is formalized as follows. The set function $Q_t^\pi(b, \mathcal{A})$ is submodular in $\mathcal{A}$, if for every $\mathcal{A}_M \subseteq \mathcal{A}_N \subseteq \mathcal{A}^+$ and $i \in \mathcal{A}^+ \setminus \mathcal{A}_N$,

$$\Delta_{Q_b}(i|\mathcal{A}_M) \geq \Delta_{Q_b}(i|\mathcal{A}_N), \tag{4.5}$$

where $\Delta_{Q_b}(i|\mathcal{A}) = Q_t^\pi(b, \mathcal{A} \cup \{i\}) - Q_t^\pi(b, \mathcal{A})$ is the *marginal gain* of adding i to $\mathcal{A}$. Equivalently, $Q_t^\pi(b, \mathcal{A})$ is submodular if for every $\mathcal{A}_M, \mathcal{A}_N \subseteq \mathcal{A}^+$,

$$Q_t^\pi(b, \mathcal{A}_M \cap \mathcal{A}_N) + Q_t^\pi(b, \mathcal{A}_M \cup \mathcal{A}_N) \leq Q_t^\pi(b, \mathcal{A}_M) + Q_t^\pi(b, \mathcal{A}_N). \tag{4.6}$$

Submodularity is an important property because of the following result by [Nemhauser et al., 1978]:

Theorem 5. *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π denote a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$. Let $Q_t^\pi(b, \mathcal{A})$ be the t-step value function for following a policy π, that is,*

$$Q_t^\pi(b, \mathcal{A}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}}), \tag{4.7}$$

where $V_{t-1}^\pi(b)$ is the value of belief b at $t-1$ time steps to go, given recursively by:

$$V_{t-1}^\pi(b) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \pi(b)) V_{t-2}^\pi(b_{\mathbf{z}}^{\pi(b)}), \tag{4.8}$$

and $V_0^\pi(b) = \rho(b)$.

If $Q_t^\pi(b, \mathcal{A})$ is non-negative, monotone and submodular in $\mathcal{A}$, then

$$Q_t^\pi(b, \mathcal{A}^G) \geq (1 - e^{-1}) Q_t^\pi(b, \mathcal{A}^*), \tag{4.9}$$

where $\mathcal{A}^G = \texttt{greedy-argmax}(Q_t^\pi(b, \cdot), \mathcal{A}^+, k)$ and $\mathcal{A}^* = \arg\max_{\mathcal{A}\in\mathcal{A}^+} Q_t^\pi(b, \mathcal{A})$, and $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$, and $\mathcal{X} = \{1, 2, \ldots, n\}$ is the set of n sensors.

However, Theorem 5 gives a bound only for a single application of `greedy-argmax`, not for applying it within each backup, as greedy PBVI does. In this section, we establish such a bound. Let the *greedy Bellman operator* $\mathfrak{B}^G$ be:

$$(\mathfrak{B}^G V_{t-1})(b) = \overset{G}{\max_{\mathcal{A}}}[\rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathcal{A}, b) V_{t-1}(b_{\mathbf{z}}^{\mathcal{A}})], \tag{4.10}$$

where $\max_{\mathcal{A}}^G$ refers to greedy maximization. This immediately implies the following corollary to Theorem 5:

Corollary 1. *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π denote a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \ldots, n\}$ denotes the set of n sensors. Let $Q_t^\pi(b, \mathcal{A})$ be the t-step value function for following a policy π, that is, $Q_t^\pi(b, \mathcal{A}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})$, where $V_{t-1}^\pi(b)$ is the value of belief b at $t-1$ time steps to go, given recursively by:*

$$V_{t-1}^\pi(b) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \pi(b)) V_{t-2}^\pi(b_{\mathbf{z}}^{\pi(b)}), \tag{4.11}$$

and $V_0^\pi(b) = \rho(b)$.

If $Q_t^\pi(b, \mathcal{A})$ *is non-negative, monotone, and submodular in* $\mathcal{A}$*, then,*

$$(\mathfrak{B}^G V_{t-1}^\pi)(b) \geq (1 - e^{-1})(\mathfrak{B}^* V_{t-1}^\pi)(b). \tag{4.12}$$

Proof. From Theorem 5 since $(\mathfrak{B}^G V_{t-1}^\pi)(b) = \max_{\mathcal{A}}^G[\rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathcal{A}, b) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})] = Q_t^\pi(b, \mathcal{A}^G)$ and $(\mathfrak{B}^* V_{t-1}^\pi)(b) = \max_{\mathcal{A}}[\rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathcal{A}, b) V_{t-1}^*(b_{\mathbf{z}}^{\mathcal{A}})] = Q_t^\pi(b, \mathcal{A}^*)$. Here $\max_{\mathcal{A}}^G$ denote greedy maximization over $\mathcal{A}$. □

In addition, we can prove that the error in the value function remains bounded after application of $\mathfrak{B}^G$.

Lemma 1. *Let* $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP. Let* π *be a policy that maps beliefs to actions in* $\mathcal{A}^+$*,* $\pi(b) = \mathcal{A}$*, where* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ *and* $\mathcal{X} = \{1, 2, \ldots, n\}$ *denotes the set of* n *sensors. Let* $Q_t^\pi(b, \mathcal{A})$ *be the* t*-step value function for following a policy* π*, that is,* $Q_t^\pi(b, \mathcal{A}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})$*, where* $V_{t-1}^\pi(b)$ *is the value of belief* b *at* $t-1$ *time steps to go , given recursively by:*

$$V_{t-1}^\pi(b) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \pi(b)) V_{t-2}^\pi(b_{\mathbf{z}}^{\pi(b)}), \tag{4.13}$$

and $V_0^\pi = \rho(b)$.

If, for all b*,* $\rho(b) \geq 0$ *and,*

$$V_{t-1}^\pi(b) \geq (1-\epsilon) V_{t-1}^*(b), \tag{4.14}$$

and if $Q_t^\pi(b, \mathcal{A})$ *is non-negative, monotone, and submodular in* $\mathcal{A}$ *then, for* $\epsilon \in [0, 1]$*,*

$$(\mathfrak{B}^G V_{t-1}^\pi)(b) \geq (1 - e^{-1})(1-\epsilon)(\mathfrak{B}^G V_{t-1}^*)(b), \tag{4.15}$$

where V_{t-1}^* *denotes the optimal value function, given by* $V_{t-1}^*(b) = \max_{\mathcal{A}\in\mathcal{A}^+}[\rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-2}^*(b_{\mathbf{z}}^{\mathcal{A}})]$.

Proof. Starting from (4.14) and, for a given $\mathcal{A}$, on both sides taking the expectation over $\mathbf{z}$, and adding $\rho(b)$ (since $\rho(b) \geq 0$ and $\epsilon \leq 1$):

$$\rho(b) + \mathbb{E}_{\mathbf{z}|b,\mathcal{A}}[V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})] \geq (1-\epsilon)(\rho(b) + \mathbb{E}_{\mathbf{z}|b,\mathcal{A}}[V_{t-1}^*(b_{\mathbf{z}}^{\mathcal{A}})]).$$

From the definition of Q_t^π, we thus have:

$$Q_t^\pi(b, \mathcal{A}) \geq (1 - \epsilon) Q_t^*(b, \mathcal{A}). \tag{4.16}$$

From Theorem 5, we know, if $Q_t^\pi(b, \mathcal{A})$ is non-negative, monotone and submodular then,

$$Q_t^\pi(b, \mathcal{A}_\pi^G) \geq (1 - e^{-1}) Q_t^\pi(b, \mathcal{A}_\pi^*), \tag{4.17}$$

where $\mathcal{A}_\pi^G = \texttt{greedy-argmax}(Q_t^\pi(b, \cdot), \mathcal{A}^+, k)$ and $\mathcal{A}_\pi^* = \arg\max_{\mathcal{A}} Q_t^\pi(b, \mathcal{A})$. Since $Q_t^\pi(b, \mathcal{A}_\pi^*) \geq Q_t^\pi(b, \mathcal{A})$ for any $\mathcal{A}$,

$$Q_t^\pi(b, \mathcal{A}_\pi^G) \geq (1 - e^{-1}) Q_t^\pi(b, \mathcal{A}_*^G), \tag{4.18}$$

where $\mathcal{A}_*^G = \texttt{greedy-argmax}(Q_t^*(b, \cdot), \mathcal{A}^+, k)$. Finally, let $\mathcal{A}$ be $\mathcal{A}_*^G$ in (4.16) which implies that $Q_t^\pi(b, \mathcal{A}_*^G) \geq (1 - \epsilon) Q_t^*(b, \mathcal{A}_*^G)$, so:

$$Q_t^\pi(b, \mathcal{A}_\pi^G) \geq (1 - e^{-1})(1 - \epsilon) Q_t^*(b, \mathcal{A}_*^G) \tag{4.19}$$

$Q_t^\pi(b, \mathcal{A}_\pi^G)$ is the same as $\mathfrak{B}^G V_{t-1}^\pi(b) = \max_{\mathcal{A}}^G[\rho(b) + \sum_{\mathbf{z} \in \Omega} \Pr(\mathbf{z} | \mathcal{A}, b) V_{t-1}(b_{\mathbf{z}}^{\mathcal{A}})] = Q_t^\pi(b, \mathcal{A}_\pi^G)$. Applying the same equations with $\pi = \pi^*$ (optimal policy), we get $(\mathfrak{B}^G V_{t-1}^*)(b) = Q_t^*(b, \mathcal{A}_*^G)$. Thus,

$$(\mathfrak{B}^G V_{t-1}^\pi)(b) \geq (1 - e^{-1})(1 - \epsilon)(\mathfrak{B}^G V_{t-1}^*)(b). \tag{4.20}$$

□

Next, we define the *greedy Bellman equation*: $V_t^G(b) = (\mathfrak{B}^G V_{t-1}^G)(b)$, where $V_0^G = \rho(b)$. Note that V_t^G is the true value function obtained by greedy maximization, without any point-based approximations. Using Corollary 1 and Lemma 1, we can bound the error of V^G with respect to V^*.

Theorem 6. *Let* $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h \rangle$ *be an active perception POMDP. Let* π *be a policy that maps beliefs to actions in* $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, *where* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ *and* $\mathcal{X} = \{1, 2, \ldots, n\}$ *denotes the set of* n *sensors. Let* $Q_t^\pi(b, \mathcal{A})$ *be the t-step value function for following a policy* π, *that is,* $Q_t^\pi(b, \mathcal{A}) = \rho(b) + \sum_{\mathbf{z} \in \Omega} \Pr(\mathbf{z} | b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})$, *where* $V_{t-1}^\pi(b)$ *is the value of belief* b *at* $t - 1$ *time steps to go, given recursively by:*

$$V_{t-1}^\pi(b) = \rho(b) + \sum_{\mathbf{z} \in \Omega} \Pr(\mathbf{z} | b, \pi(b)) V_{t-2}^\pi(b_{\mathbf{z}}^{\pi(b)}), \tag{4.21}$$

and $V_0^\pi = \rho(b)$.

If, for all b, $\rho(b) \geq 0$ *and if, for all policies* π*, beliefs,* b *and* t, $Q_t^\pi(b, \mathcal{A})$ *is non-negative, monotone and submodular in* $\mathcal{A}$*, then for all* b*,*

$$V_t^G(b) \geq (1 - e^{-1})^{2t} V_t^*(b). \tag{4.22}$$

Proof. By induction on t. The base case, $t = 0$, holds because $V_0^G(b) = \rho(b) = V_0^*(b)$.

In the inductive step, for all b, we assume that

$$V_{t-1}^G(b) \geq (1 - e^{-1})^{2t-2} V_{t-1}^*(b), \tag{4.23}$$

and must show that

$$V_t^G(b) \geq (1 - e^{-1})^{2t} V_t^*(b). \tag{4.24}$$

Lemma 1 says that for $\epsilon \in [0, 1]$, if $Q_t^\pi(b, \mathcal{A})$ is non-negative, monotone and submodular and if $\rho(b) \geq 0$ for all b then

$$(\mathfrak{B}^G V_{t-1}^\pi)(b) \geq (1 - e^{-1})(1 - \epsilon)(\mathfrak{B}^G V_{t-1}^*)(b). \tag{4.25}$$

Thus, applying Lemma 1 with $V_{t-1}^\pi = V_{t-1}^G$, and $(1 - \epsilon) = (1 - e^{-1})^{2t-2}$ (and since the statement of the Theorem states that if $Q_t^\pi(b, \mathcal{A})$ is non-negative, monotone and submodular for all π, t, thus, since $Q_t^G(b, \mathcal{A})$ is non-negative, monotone and submodular),

$$(\mathfrak{B}^G V_{t-1}^G)(b) \geq (1 - e^{-1})(1 - e^{-1})^{2t-2}(\mathfrak{B}^G V_{t-1}^*)(b) \tag{4.26}$$

$$V_t^G(b) \geq (1 - e^{-1})^{2t-1}(\mathfrak{B}^G V_{t-1}^*)(b). \tag{4.27}$$

Eq. (4.27) follows because $\mathfrak{B}^G V_{t-1}^G(b)$ is $V_t^G(b)$ by definition.

According to Corollary 1, if $Q_t^\pi(b, \mathcal{A})$ is non-negative, monotone and submodular then:

$$(\mathfrak{B}^G V_{t-1}^\pi)(b) \geq (1 - e^{-1})(\mathfrak{B}^* V_{t-1}^\pi)(b). \tag{4.28}$$

Applying Corollary 1 with $V_{t-1}^\pi = V_{t-1}^*$ (and assuming $Q_t^*(b, \mathcal{A})$ is non-negative, monotone and submodular) we get,

$$(\mathfrak{B}^G V_{t-1}^*)(b) \geq (1 - e^{-1})(\mathfrak{B}^* V_{t-1}^*)(b). \tag{4.29}$$

Using (4.29) and (4.27), we get,

$$\begin{aligned} V_t^G(b) &\geq (1-e^{-1})^{2t-1}(1-e^{-1})(\mathfrak{B}^* V_{t-1}^*)(b) \\ V_t^G(b) &\geq (1-e^{-1})^{2t} V_t^*(b). \end{aligned} \tag{4.30}$$

□

Theorem 6 extends Nemhauser's result to a full sequential decision making setting where multiple applications of greedy maximization are employed over multiple time steps. This theorem gives a theoretical guarantee on the performance of greedy PBVI. Given a POMDP with a submodular value function, greedy PBVI is guaranteed to have bounded error with respect to the optimal value function. Moreover, this performance comes at a computational cost that is much less than that of solving the same POMDP with traditional solvers. Thus, greedy PBVI scales much better in the size of the action space of active perception POMDPs, while still retaining bounded error.

The results presented in this subsection are applicable only if the value function for a POMDP is submodular. In the following subsections, we establish the submodularity of value function for active perception POMDP under certain conditions.

4.1.2 Submodularity of value functions

In this subsection, we establish the sufficient conditions for the submodularity of the value function of the active perception POMDP. Specifically, we show that, when using negative belief entropy as the immediate belief-based reward, i.e., $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$, then under certain conditions $Q_t^\pi(b, \mathcal{A})$ is submodular as required by Theorem 6. Please note that the additional term $\log(\frac{1}{|S|})$ is only required (and sufficient) to guarantee non-negativity, but is independent of the actual beliefs or actions. For the sake of conciseness, in the remainder of this thesis we will omit this term. We start by observing that: $Q_t^\pi(b, \mathcal{A}) = \rho(b) + \sum_{j=1}^{t-1} G_j^\pi(b^t, \mathcal{A}^t)$, where $G_j^\pi(b^t, \mathcal{A}^t)$ is the expected immediate reward with j steps to go, conditioned on the belief and action with t steps to go and assuming policy π is followed in between t and j:

$$G_j^\pi(b^t, \mathcal{A}^t) = \sum_{\mathbf{z}^{t:j}} \Pr(\mathbf{z}^{t:j} | b^t, \mathcal{A}^t, \pi)(-H_{b^j}(s^j)). \tag{4.31}$$

where $\mathbf{z}^{t:j}$ is a vector of observations received in the interval from t steps to j steps to go, b^t is the belief at t steps to go, $\mathcal{A}^t$ is the action taken at t steps to go, and $\rho(b^j) = -H_{b^j}(s^j)$, where s^j is the state at j steps to go.

Proving that $Q_t^\pi(b, \mathcal{A})$ is submodular in $\mathcal{A}$ requires three steps. First, we show that

$G^\pi_j(b^t, \mathcal{A}^t)$ equals the *conditional entropy* of b^j over s^j given $\mathbf{z}^{t:j}$. Second, we show that, under certain conditions, conditional entropy is a submodular set function. Third, we combine these two results to show that $Q^\pi_t(b, \mathcal{A})$ is submodular.

The *conditional entropy* [Cover and Thomas, 1991] of a distribution b over s given some observations **z** is defined as:

$$H_b(s|\mathbf{z}) = -\sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \mathcal{A}) H_{b^{\mathcal{A}}_{\mathbf{z}}}(s) \tag{4.32}$$

Thus, conditional entropy is the expected entropy given **z** has been observed but marginalizing across the values it can take on. Equivalently, conditional entropy is also defined as:

$$H_b(s|\mathbf{z}) = -\sum_{s}\sum_{\mathbf{z}} \Pr(s, \mathbf{z}|b, \mathcal{A}) \log(\frac{\Pr(s, \mathbf{z}|b, \mathcal{A})}{\Pr(\mathbf{z}|b, \mathcal{A})}) \tag{4.33}$$

Lemma 2. *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π be a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \ldots, n\}$ denotes the set of n sensors. Let b^t and $\mathcal{A}^t$ denote the belief and action at t time steps to go. If the immediate belief-based reward ρ in $\mathbf{M}$ is defined as the negative belief entropy, that is, $\rho(b) = -H_b(s)$ then the expected reward at $j(< t)$ time steps to go, as a consequence of taking action $\mathcal{A}^t$ in belief b^t and following policy π in between the time interval t to j equals the negative discounted conditional entropy of b^j over s^j given $\mathbf{z}^{t:j}$:*

$$G^\pi_j(b^t, \mathcal{A}^t) = \sum_{\mathbf{z}^{t:j}} \Pr(\mathbf{z}^{t:j}|b^t, \mathcal{A}^t, \pi)(-H_{b^j}(s^j)) = -H_{b^j}(s^j|\mathbf{z}^{t:j}). \tag{4.34}$$

Here b^j is the belief at j time steps to go, s^j denote the hidden state at j time steps to go and $\mathbf{z}^{t:j}$ denote the sequence of observations that can be obtained in between the time interval from t to j.

Proof. Proof in Appendix. □

Next, we identify the conditions under which $G^\pi_j(b^t, \mathcal{A}^t)$ is submodular in $\mathcal{A}^t$. We use the set equivalent $\mathcal{Z}$ of **z** since submodularity is a property of set functions. Thus:

$$G^\pi_j(b^t, \mathcal{A}^t) = -H_{b^j}(s|\mathcal{Z}^{t:j}), \tag{4.35}$$

where $\mathcal{Z}^{t:j}$ is a set of observation features observed between t and j time steps to go. The key condition sufficient for submodularity of $G^\pi_j(b^t, \mathcal{A}^t)$ is *conditional independence* [Krause and Guestrin, 2005b].

Definition 5. The observation set $\mathcal{Z} = \{z_1, z_2, \dots z_n\}$ is conditionally independent given s if any pair of observation features (or variables) z_i, z_j in $\mathcal{Z}$ are conditionally independent given the hidden state s, i.e.,

$$\Pr(z_i, z_j|s) = \Pr(z_i|s)\Pr(z_j|s), \quad \forall z_i, z_j \in \mathcal{Z}. \tag{4.36}$$

Lemma 3. *Let $\mathcal{Z} = \{z_1, z_2, \dots z_n\}$ be the set of all observation features about a hidden variable s. If $\mathcal{Z}$ is conditionally independent given s then $-H(s|\mathcal{Z})$ is submodular in $\mathcal{Z}$, i.e., for any two observations set $\mathcal{Z}_M, \mathcal{Z}_N \subseteq \mathcal{Z}$,*

$$H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) + H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) \geq H(s|\mathcal{Z}_M) + H(s|\mathcal{Z}_N). \tag{4.37}$$

Proof. Proof in Appendix. □

Lemma 4. *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π be a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \dots, n\}$ denotes the set of n sensors. Let b^t and $\mathcal{A}^t$ denote the belief and action at t time steps to go. Let $\mathcal{Z}^{t:j} = \{z_1^t, z_2^t, \dots, z_n^t, z_1^{t-1}, z_2^{t-1}, \dots, z_n^j\}$ be the set of observation features that can be obtained between the time interval from t to j. If the immediate belief-based reward ρ in $\mathbf{M}$ is defined as the negative belief entropy, that is, $\rho(b) = -H_b(s)$, and if $\mathcal{Z}^{t:j}$ is conditionally independent given s^j, then $G_j^\pi(b^t, \mathcal{A}^t) = -H_{b^j}(s^j|\mathcal{Z}^{t:j})$ is submodular in $\mathcal{A}^t$ $\forall$ π. Here b^j is the belief at j time steps to go, s^j denote the hidden state at j time steps to go.*

Proof. Proof in Appendix. □

Now we can establish the submodularity of Q_t^π.

Theorem 7. *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π be a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \dots, n\}$ denotes the set of n sensors. Let b^t and $\mathcal{A}^t$ denote the belief and action at t time steps to go. Let $\mathcal{Z}^{t:j} = \{z_1^t, z_2^t, \dots, z_n^t, z_1^{t-1}, z_2^{t-1}, \dots, z_n^j\}$ be the set of observation features that can be obtained between the time interval from t to j. If the immediate belief-based reward ρ in $\mathbf{M}$ is defined as the negative belief entropy, that is, $\rho(b) = -H_b(s)$, and if $\mathcal{Z}^{t:j}$ is conditionally independent given s^j for all j, $1 < j < t$, then $Q_t^\pi(b, \mathcal{A}) = Q_t^\pi(b^t, \mathcal{A}^t) = \rho(b) + \sum_{j=1}^{t-1} G_j^\pi(b^t, \mathcal{A}^t)$ is submodular in $\mathcal{A}$, for all π.*

Proof. $\rho(b)$ is trivially submodular in $\mathcal{A}$ because it is independent of $\mathcal{A}$. Furthermore, Lemma 4 shows that $G_j^\pi(b^t, \mathcal{A}^t)$ is submodular in $\mathcal{A}^t$, for all (or any) π. Since a positively weighted sum of submodular functions is also submodular [Krause and Golovin, 2014], this implies that $\sum_{j=1}^{t-1} G_j^\pi(b^t, \mathcal{A}^t)$ and thus $Q_t^\pi(b, \mathcal{A})$ are also submodular in $\mathcal{A}$ for all π. □

While the conditional independence of $\mathcal{Z}^j$ given s^j is easy to satisfy, the conditional independence of $\mathcal{Z}^{t:j}$, a whole sequence of observations, given s^j is more difficult. For $\mathcal{Z}^{t:j}$ to be conditionally independent given s^j, s^j must contain enough information to predict the past sequence of states $s^{t:j}$. One way to achieve this is by defining s^j such that it encodes all the information in the state history that is correlated with the observations $\mathcal{Z}^{t:j}$. Unfortunately, this typically is not practical to do unless the transition function is deterministic and invertible. However, note that the conditions required by Theorem 7 are only sufficient, not necessary, conditions for the value function to be submodular. An important goal for future work is thus to identify weaker conditions for establishing submodularity of value functions based on belief entropy or other belief-based reward functions. Figure 4.1 shows that for a simulated setting even if the state of the world is changing the increase in expected immediate reward (entropy in this case) still shows the notion of diminishing returns as more and more observations are made. The same figure (bottom plot) shows the expected increase in value function $Q_t^\pi(b, \mathcal{A})$ as the size of $\mathcal{A}$ increases. Finally, we show in the Experiments section, greedy PBVI performs well in practice on real-world data for which the conditions in Theorem 7 might not hold, which suggests that establishing submodularity under weaker conditions may indeed be possible.

While Theorem 7 shows that $Q_t^\pi(b, \mathcal{A})$ is submodular, Theorem 6 also requires that it be monotone, which we now establish:

Lemma 5. *Let* $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP. Let* π *be a policy that maps beliefs to actions in* $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, *, where* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ *and* $\mathcal{X} = \{1, 2, \ldots, n\}$ *denotes the set of* n *sensors. Let* $Q_t^\pi(b, \mathcal{A}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})$ *be the t-step action value function of policy* π. *If* $V_{t-1}^\pi(b')$ *is a convex function of* b', *then* $Q_t^\pi(b, \mathcal{A})$ *is monotone in* $\mathcal{A}$, *i.e., for all* b *and* $\mathcal{A}_M \subseteq \mathcal{A}_N$ *(*$\mathcal{A}_M, \mathcal{A}_N \in \mathcal{A}^+$*),* $Q_t^\pi(b, \mathcal{A}_M) \leq Q_t^\pi(b, \mathcal{A}_N)$.

Proof. Proof in Appendix. □

Tying together our results so far:

Theorem 8. *Let* $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP. Let* π *be a policy that maps beliefs* b *to actions* $\mathcal{A}$ *in* $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, *, where* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ *and* $\mathcal{X} = \{1, 2, \ldots, n\}$ *denotes the set of* n *sensors.*

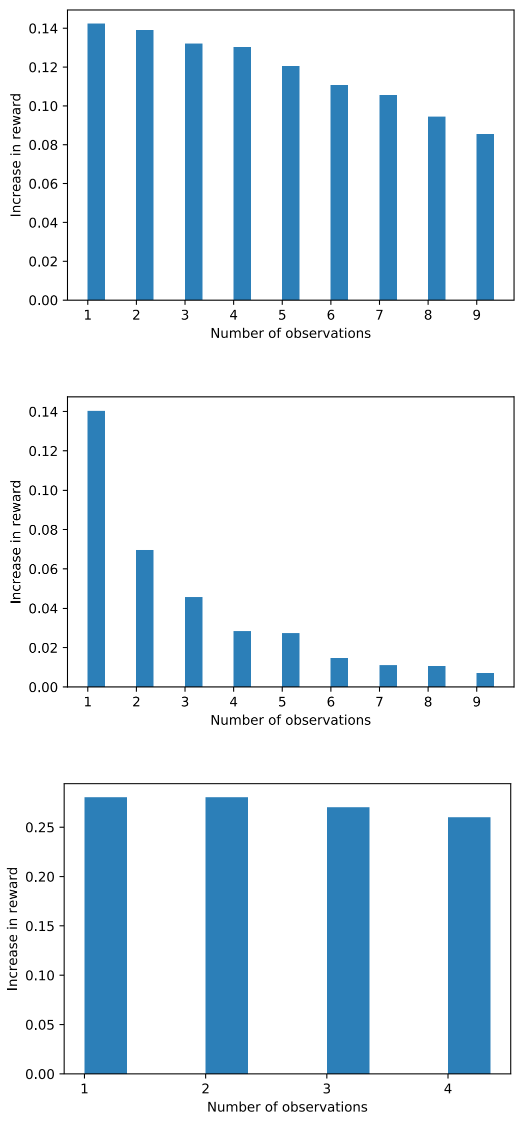


Figure 4.1: The figure shows the average (averaged over more than 20000 runs) increase in the expected immediate reward and the value function in a simulated setting (Chapter 3, Figure 3.3), for an agent that follows random policy for the sensor selection task. The top figure shows the increase in the expected immediate reward as more and more observations are added (randomly) when the state is static (the person is not moving), the middle figure shows the increase in the expected immediate reward when the state is changing, the bottom figure shows the increase in the value function (sum of expected reward over the next 3 time steps), as more and more observations are added at the first time step. Even as the state of the world changes in the simulated setting no large deviation from submodularity of the value function was observed.

- *If the set of observation features that can be obtained between the time interval t and j, $\mathcal{Z}^{t:j} = \{z_1^t, z_2^t, \ldots, z_n^t, z_1^{t-1}, \ldots z_n^j\}$ is conditionally independent given s^j, for all $j(1 < j < t)$, and*
- *if $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$, and*
- *if $V_t^\pi(b)$ is convex in b for all π, t, where V_t^π is the value function for policy π, given recursively by:*

$$V_t^\pi(b) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \pi(b)) V_{t-1}^\pi(b_{\mathbf{z}}^{\pi(b)}), \tag{4.38}$$

and $V_0^\pi = \rho(b)$,

then for all b,

$$V_t^G(b) \geq (1 - e^{-1})^{2t} V_t^*(b), \tag{4.39}$$

where $V_t^G = \max^G_{\mathcal{A}\in\mathcal{A}^+}[\rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})]$, $\max^G$ denotes greedy maximization (Algorithm 1), that returns $\mathcal{A}^G$ build greedily by iteratively adding elements out of $\mathcal{X} = \{1, 2, \ldots, n\}$, and V_t^* denote the value function for the optimal policy.

Proof. Follows from Theorem 6, given $Q_t^\pi(b, \mathcal{A}) = [\rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathcal{A}})]$ is non-negative, monotone and submodular for all π. For $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$, it is easy to see that $Q_t^\pi(b, \mathcal{A})$ is non-negative, since maximum value of $H_b(s)$ is $-\log(\frac{1}{|S|})$ [Cover and Thomas, 1991] and $Q_t^\pi(b, \mathcal{A})$ is a positive sum of $\rho(b^t)$ at different time steps t. Theorem 7 showed that $Q_t^\pi(b, \mathcal{A})$ is submodular if $\rho(b) = -H_b(s)$ and if $\mathcal{Z}^{t:j}$ is conditionally independent given s^j. The monotonicity of $Q_t^\pi(b, \mathcal{A})$ follows from Lemma 5 if $V_t^\pi(b)$ is convex in b for all π, t. □

As discussed earlier, (below Theorem 7) the conditional independence of $\mathcal{Z}^{t:j}$ given s^j is difficult to meet in real-life. Only if s^j is modelled such that it contains all the information of the state history then such a condition might be true. However, this is a sufficient condition and not a necessary one and we hope to relax this condition in future. Also, V_t^π is not guaranteed to be convex for all π. Both these conditions can be relaxed in future as a quick look to the proof of Theorem 6 reveals that ideally, only $Q_t^G(b, \mathcal{A})$ and $Q_t^*(b, \mathcal{A})$ are required to be monotone and submodular (for all t). Now, $V_t^*(b)$ is convex in b but $V_t^G(b)$ is not necessarily convex in b. However, note that V_t^G is the value function that is without the point-based approximation. In practice, the value function greedy PBVI returns, is represented as a set of vectors and the policy that is executed using this value function involves taking a maximization over the dot product of the given belief and the set of vectors (Figure 4.2). Thus, to relax this condition, in

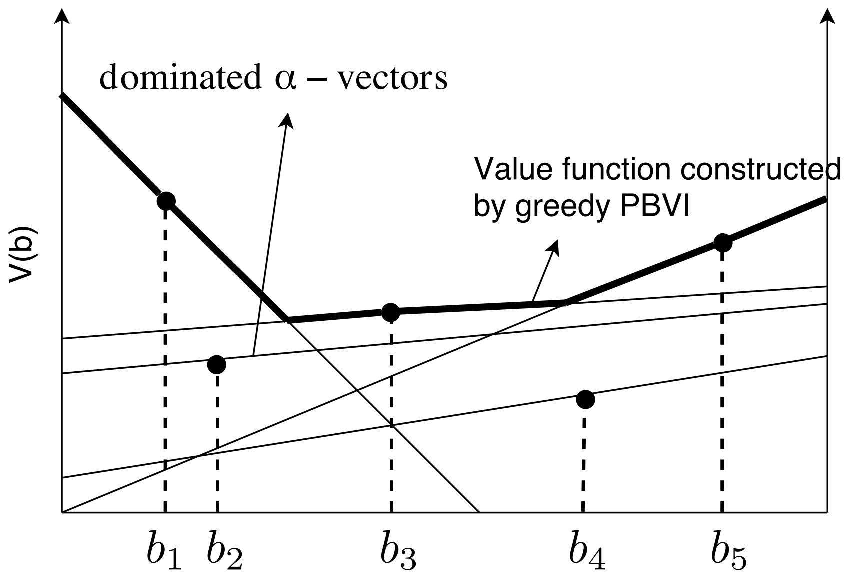


Figure 4.2: Figure explaining the working of greedy PBVI for a 2-state POMDP (x-axis is the belief space). The black circles denote the value of the belief computed by greedy PBVI. The figure also shows a case where the black circles can be (inter/extra)-polated so that it eventually leads to a non-convex value function. However, point-based methods return a set of α-vectors, one for each belief that is sampled. The final policy that is executed by an agent takes the maximum (or argmax to get the action that projects the vector with highest value for a belief) over these α vectors and thus even if greedy maximization computes sub-optimal actions as shown in figure, the executed policy might still have higher value than the one suggested by greedy maximization. The figure also shows that because of a final maximization over all alpha vectors the resulting value function turns out to be convex.

addition to the greedy operator an extra operator can be introduced that represents V_t^G as the upper surface of a set of vectors and hence a convex function. Figure 4.2 explains the good performance of greedy PBVI as observed in the Experiment section, since the policy executed using the value function obtained from greedy PBVI has a value that is greater that the current definition of $V_t^G(b)$.

In this subsection we showed that if the immediate belief-based reward $\rho(b)$ is defined as negative belief entropy, then the value function of an active perception POMDP is guaranteed to be submodular under certain conditions. However, as mentioned earlier, to solve active perception POMDP, we approximate the belief entropy with vector tangents. This might interfere with the submodularity of the value function. In the next subsection, we show that, even though the PWLC approximation of belief entropy might interfere with the submodularity of the value function, the value function computed by greedy PBVI is still guaranteed to have bounded error.

4.1.3 Bounds given approximated belief entropy

While Theorem 8 bounds the error in $V_t^G(b)$, it does so only on the condition that $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$. However, as discussed earlier, our definition of active perception POMDPs instead defines ρ using a set of vectors $\Gamma^\rho = \{\alpha_1^\rho, \ldots, \alpha_{|B|}^\rho\}$, each of which is a tangent to $-H_b(s)$, as suggested by Araya-lópez et al. [2010], in order to preserve the PWLC property. While this can interfere with the submodularity of $Q_t^\pi(b, \mathcal{A})$, here we show that the error generated by this approximation is still bounded in this case.

Let $\tilde{\rho}(b) = \max_{\alpha \in \Gamma^\rho} \sum_s b(s)\alpha(s)$ denote the PWLC approximated entropy and $\tilde{V}_t^*$ denote the optimal value function when using a PWLC approximation to negative entropy for the belief-based reward, i.e.,

$$\tilde{V}_t^*(b) = \max_{\mathcal{A}}[\tilde{\rho}(b) + \sum_{\mathbf{z} \in \Omega} \Pr(\mathbf{z}|b, \mathcal{A})\tilde{V}_{t-1}^*(b_{\mathbf{z}}^{\mathcal{A}})]. \tag{4.40}$$

Araya-lópez et al. [2010] showed that, if $\rho(b)$ satisfies the α-Hölder condition [Gilbarg and Trudinger, 2001], a generalization of the Lipschitz condition, then the following relation holds between V_t^* and $\tilde{V}_t^*$:

$$||V_t^* - \tilde{V}_t^*||_\infty \leq C\delta^\alpha, \tag{4.41}$$

where V_t^* is the optimal value function, C is a constant and $\delta_B = \min_{b \in \square} \max_{b' \in B} ||b - b'||_1$ is the *density* of the set of belief points B at which tangent are drawn to the belief entropy and $\square$ is the belief simplex[1].

Let $\tilde{V}_t^G(b)$ be the value function computed with greedy Bellman equation when immediate belief-based reward is $\tilde{\rho}(b)$:

$$\tilde{V}_t^G(b) = \overset{G}{\max_{\mathcal{A}}}[\tilde{\rho}(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A})\tilde{V}_{t-1}^G(b_{\mathbf{z}}^{\mathcal{A}})], \tag{4.43}$$

then the error between $\tilde{V}_t^G(b)$ and $V_t^*(b) = \max_{\mathcal{A}}[\rho(b) + \sum_{\mathbf{z}\in\Omega}\Pr(\mathbf{z}|b,\mathcal{A})V_{t-1}^*(b_{\mathbf{z}}^{\mathcal{A}})]$ is bounded as stated in the following theorem.

Theorem 9. *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π be a policy that maps beliefs b to actions $\mathcal{A}$ in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$. Let the true immediate belief-based reward be negative belief entropy $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$. Let $\Gamma^\rho = \{\alpha_1^\rho, \dots, \alpha_{|B|}^\rho\}$ be a set of $|B|$ $|S|$-dimensional tangent planes to $\rho(b)$ at $|B|$ belief points where B is the set of belief points at which tangents ($|S|$-dimensional plane) are drawn to $\rho(b)$. For all beliefs, the error between $\tilde{V}_t^G(b)$ and $V_t^*(b)$ is bounded, if $\mathcal{Z}^{t:j} = \{z_1^t, z_2^t, \dots, z_1^{t-1}, \dots z_n^j\}$ is conditionally independent given s^j for all j, if $\rho(b)$ satisfies the α-Hölder condition and if $V_t^\pi(b)$ is convex in b for all π.*

Proof. To prove this theorem we first argue that the error between

$$\tilde{V}_t^\pi(b) = [\tilde{\rho}(b) + \sum_{\mathbf{z}\in\Omega}\Pr(\mathbf{z}|b,\pi(b))\tilde{V}_{t-1}^\pi(b_{\mathbf{z}}^{\pi(b)})], \tag{4.44}$$

and

$$V_t^\pi(b) = [\rho(b) + \sum_{\mathbf{z}\in\Omega}\Pr(\mathbf{z}|b,\pi(b))V_{t-1}^\pi(b_{\mathbf{z}}^{\pi(b)})], \tag{4.45}$$

is bounded for all b and π: Theorem 4.3 in Araya-López et. al., 2010 shows that the error between $\rho(b)$ and $\tilde{\rho}(b)$ is bounded by $C\delta_B^\alpha$ (if ρ is convex and satisfies the α-Holder condition). We assume $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$ satisfies the α-Holder condition (more details on this can be found in Araya-lópez et al. [2010]) for some α. Here C is a scalar constant and δ_B is the *density* of the set of belief points B at which tangents hyperplanes are drawn to approximate ρ. Specifically, $\delta_B = \min_{b\in\square}\max_{b'\in B}||b - b'||_1$, where $\square$ is the belief simplex. Since for any policy π and for a finite horizon h, $\tilde{V}_t^\pi(b)$ is the sum of expected $\tilde{\rho}(b^t)$ obtained at each step, (each of which has bounded error with

[1]More concretely, Araya-lópez et al. [2010] showed that

$$||V_t^* - \tilde{V}_t^*||_\infty \leq \frac{C\delta^\alpha}{1-\gamma}, \tag{4.42}$$

where γ is a discount factor in an infinite horizon POMDP (or ρPOMDP). We include the $1-\gamma$ into the constant term because we consider a finite horizon setting and thus do not have a formal γ defined.

the true $\rho(b^t)$), the error between $\tilde{V}_t^{\pi}(b)$ and $V_t^{\pi}(b)$ is bounded by $C\delta_B^{\alpha}$ for some constant C. Mathematically,

$$\begin{aligned} V_t^{\pi}(b) &= \mathbb{E}[\sum_{j=0}^{t-1} \rho(b^j)|\pi] \\ &\leq \mathbb{E}[\sum_{j=0}^{t-1} (\tilde{\rho}(b^j) + C\delta_B^{\alpha})|\pi] \\ &= \tilde{V}_t^{\pi}(b) + hC\delta_B^{\alpha}. \end{aligned} \tag{4.46}$$

Earlier, Theorem 8 showed

$$V_t^G(b) \geq (1-e^{-1})^{2t} V_t^*(b) \ \forall \ b, \tag{4.47}$$

if $\rho(b) = -(H_b(s) + \log(\frac{1}{|S|}))$ and if $\mathcal{Z}^{t:j}$ is conditionally independent given s^j and if $V_t^{\pi}(b)$ is convex in b for all π.

We just argued that, $V_t^{\pi}(b) \leq \tilde{V}_t^{\pi}(b) + hC\delta_B^{\alpha} \ \forall \ b, \pi$. This implies,

$$V_t^G(b) \leq \tilde{V}_t^G(b) + hC\delta_B^{\alpha} \ \forall \ b. \tag{4.48}$$

Since $V_t^G(b) \geq (1-e^{-1})^{2t} V_t^*(b) \ \forall \ b$,

$$\tilde{V}_t^G(b) \geq (1-e^{-1})^{2t} V_t^*(b) - hC\delta_B^{\alpha} \ \forall \ b. \tag{4.49}$$

□

In this subsection we showed that if the negative entropy is approximated using tangent vectors, greedy PBVI still computes a value function that has bounded error. In the next subsection we outline how greedy PBVI can be extended to general active perception tasks.

4.2 General Active Perception POMDPs

The results presented in the previous section apply to a subclass of active perception POMDPs in which the evolution of state over time is independent of the actions of the agent. Here, we outline how these results can be extended to general active perception POMDPs without many changes. The main application for such an extension is in tasks involving a mobile robot coordinating with sensors to intelligently take actions to perceive its environment. In such cases, the robot's actions, by causing it to move, can change the state of the world.

The algorithms we proposed can be extended to such settings by making small modifications to the greedy maximization operator. The greedy algorithm can be run for $k+1$ iterations and in each iteration the algorithm would choose to add either a sensor (only if fewer than k sensors have been selected), or a movement action (if none has been selected so far). Formally, using the work of Fisher et al. [1978], which extends that of Nemhauser et al. [1978] on submodularity to combinatorial structures such as *matroids*.

Definition 6. (Matroid) A matroid $\mathcal{M} = (V, I)$ is a finite ground set V together with I, that is a collection of subsets of V, called *independent* sets such that:

- if a set $\mathcal{B} \in I$ and $\mathcal{A} \subseteq \mathcal{B}$, then $\mathcal{A} \in I$
- if $\mathcal{A} \in I, \mathcal{B} \in I$ and $|\mathcal{A}| \leq |\mathcal{B}|$ then there exists a $i \in \mathcal{B} \setminus \mathcal{A}$ such that $\mathcal{A} \cup i \in I$.

Fischer et. al. showed that greedy maximization of a submodular function F subject to matroid constraints, that is, $\max_{\mathcal{A} \in I} F(\mathcal{A})$ (or even subject to the constraints that is intersection of multiple matroids) returns a constant factor approximation to the optimal value.

The action space of a POMDP involving a mobile robot can be modelled as a *partition matroid* (or as an intersection of multiple matroids) and greedy maximization subject to matroid constraints [Fisher et al., 1978] can be used to maximize the value function approximately. Below we describe this extension in detail for the case where a mobile robot coordinates with a deployed set of sensors to maintain surveillance in a shopping mall.

Definition 7. (Partition Matroid) $P = (V, I)$ is a partition matroid in which V is partitioned into l disjoint sets $A_1, A_2, \ldots, A_l$ and

$$I = \{\mathcal{A} \subseteq V : |\mathcal{A} \cap A_i| \leq k_i \; \forall \, i = 1, 2, \ldots, l\}, \tag{4.50}$$

for some given parameters $k_1, k_2 \ldots k_l$.

To model the active perception POMDP for the case where a mobile robot (mounted with its own sensors) is interacting with a set of fixed sensors we make the following main changes:

- Actions $\mathbf{a} = \langle a_1 \ldots a_n, a_{stay}, a_{left}, a_{right}, a_{top}, a_{bottom} \rangle$ are modelled as vectors of $m = n + 5$ *action features*, the first n of which are binary action features each of which specify whether a given sensor is selected or not (assuming n sensors). The last 5 action features, specify the action chosen for the movement of the mobile robot. Since for the case of mobile sensors, the constraints on the agent is more complicated than just selecting k out of n, we take help of *partition matroids* to model the action space of the agent. $\mathcal{A}$ is defined as before $\mathcal{A} = \{i : a_i = 1 \vee i \leq n\}$, i.e., the set of indices of the selected sensors and $\mathcal{M} = \{stay, left, right, top, bottom\}$, i.e., the set of all possible actions for the mobile sensor to move. $\mathcal{A}^+$ is the same as before, i.e., the set of sensor subsets of size k or less, $\mathcal{A}^+ = \{\mathcal{A} : |\mathcal{A}| \leq k\}$. The agent can select only one action out of $\mathcal{M}$, and it is denoted by $\mathcal{A}_m$, $\mathcal{A}_m \subset \mathcal{M} : |\mathcal{A}_m| = 1$. As stated before, $\mathcal{X} = \{1, \ldots, n\}$ indicates the set of all sensors.

Using the definition of partition matroids, we can define the action space of above POMDP as a partition matroid, $P = (V, I)$ where $V = \mathcal{X} \times \mathcal{M}$ and

$$I = \{\mathfrak{a} \subseteq V : |\mathfrak{a} \cap \mathcal{X}| \leq k \ \wedge \ |\mathfrak{a} \cap \mathcal{M}| \leq 1\} \tag{4.51}$$

- Observations $\mathbf{z} = \langle z_1 \ldots z_N, z_{n+1} \rangle$ are modelled as vectors of $n+1$ *observation features*, first n of which specifies the sensor reading obtained by the given sensor and the z_{n+1} specifies the observation from the mobile sensor. If sensor i is not selected, then $z_i = \emptyset$. The set equivalent of $\mathbf{z}$ is $\mathcal{Z} = \{z_i : z_i \neq \emptyset\}$. To prevent ambiguity about which sensor generated which observation in $\mathcal{Z}$, we assume that, for all i and j, the domains of z_i and z_j share only $\emptyset$.

Algorithm 3 `greedy-argmaxM`(Q, V, I, k)

```
$\mathfrak{a}^G \leftarrow \emptyset$.
for $m = 1$ to $k + 1$ do
    $\mathfrak{a}^G \leftarrow \mathfrak{a}^G \cup \{\arg\max_{i \notin \mathfrak{a}^G : \mathfrak{a}^G \cup \{i\} \in I} \Delta_Q(i|\mathfrak{a}^G)\}$
end for
return $\mathfrak{a}^G$
```

A simple and cheap greedy heuristic for maximizing Q subject to the partition matroid constraint is as described in algorithm 3. Algorithm 3 in each iteration adds either a sensor (only if fewer than k sensors have been selected) or a movement action (if none has been selected so far). Greedy PBVI can be modified accordingly to incorporate the greedy heuristic by modifying (4.2) with $\mathfrak{a}^G =$ `greedy-argmaxM`(Q, V, I, K). The greedy operator under matroid constraint can be defined as:

$$(\mathfrak{B}^M V_{t-1}^{\pi})(b) = \overset{M}{\max_{\mathfrak{a} \in I}}[\rho(b, \mathfrak{a}) + \sum_{\mathbf{z} \in \Omega} \Pr(\mathbf{z}|\mathfrak{a}, b) V_{t-1}^{\pi}(b_{\mathbf{z}}^{\mathfrak{a}})], \tag{4.52}$$

where $\max^M_{\mathfrak{a}}$ refers to the greedy maximization subject to matroid constraints.

Thus, we can extend our analysis from section 4.1.1 to mobile sensor active perception POMDP.

Theorem 10. *(Fisher et. al.) Let* $\mathbf{M} = \langle S, \mathcal{M}, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP where* $\mathcal{M} = (V, I)$ *is a matroid. Let* π *be a policy that maps beliefs to actions in* I*,* $\pi(b) = \mathfrak{a}$*. Let* $Q^\pi_t(b, \mathfrak{a})$ *be the* t*-step value function for following policy* π*, that is,* $Q^\pi_t(b, \mathfrak{a}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathfrak{a}) V^\pi_{t-1}(b^{\mathfrak{a}}_{\mathbf{z}})$*, where* V^π_{t-1} *is value function of policy* π *given recursively by:*

$$V^\pi_{t-1}(b) = [\rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \pi(b)) V(b^{\pi(b)}_{\mathbf{z}})], \tag{4.53}$$

and $V^\pi_0(b) = \rho(b)$*. If* $Q^\pi_t(b, \mathfrak{a})$ *is non-negative, monotone and submodular in* $\mathfrak{a}$*, then*

$$Q^\pi_t(b, \mathfrak{a}^G) \geq \frac{1}{2} \max_{\mathfrak{a}'\in I} Q^\pi_t(b, \mathfrak{a}'), \tag{4.54}$$

where $\mathfrak{a}^G = \texttt{greedy-argmaxM}(Q^\pi_t(b, \cdot), V, I, K)$*.*

Proof. Follows from Theorem 2.1 in Fisher et al. [1978]. □

Corollary 2. *Let* $\mathbf{M} = \langle S, \mathcal{M}, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP, where* $\mathcal{M} = (V, I)$ *is a matroid. Let* π *be a policy that maps beliefs to actions in* I*,* $\pi(b) = \mathfrak{a}$*. Let* $Q^\pi_t(b, \mathfrak{a})$ *be the* t*-step value function for following policy* π*, that is,* $Q^\pi_t(b, \mathfrak{a}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathfrak{a}) V^\pi_{t-1}(b^{\mathfrak{a}}_{\mathbf{z}})$*. Given any policy* π*, if* $Q^\pi_t(b, \mathfrak{a})$ *is non-negative, monotone, and submodular in* $\mathfrak{a}$*, then for all* b*,*

$$(\mathfrak{B}^M V^\pi_{t-1})(b) \geq \frac{1}{2} (\mathfrak{B}^* V^\pi_{t-1})(b). \tag{4.55}$$

Proof. Follows from Theorem 10. □

Lemma 6. *Let* $\mathbf{M} = \langle S, \mathcal{M}, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP, where* $\mathcal{M} = (V, I)$ *is a matroid. Let* π *be a policy that maps beliefs to actions in* I*,* $\pi(b) = \mathfrak{a}$*. Let* $Q^\pi_t(b, \mathfrak{a})$ *be the* t*-step value function for following policy* π*, that is,* $Q^\pi_t(b, \mathfrak{a}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathfrak{a}) V^\pi_{t-1}(b^{\mathfrak{a}}_{\mathbf{z}})$*. If for all* b*,* $\rho(b) \geq 0$*,*

$$V^\pi_t(b) \geq (1 - \epsilon) V^*_t(b), \tag{4.56}$$

and $Q^\pi_t(b, \mathfrak{a})$ *is non-negative, monotone, and submodular then, for* $\epsilon \in [0, 1]$*,*

$$(\mathfrak{B}^M V^\pi_t)(b) \geq (\frac{1}{2})(1 - \epsilon)(\mathfrak{B}^M V^*_t)(b). \tag{4.57}$$

Proof. Proof in Appendix. □

Theorem 11. *Let* $\mathbf{M} = \langle S, \mathcal{M}, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP where* $\mathcal{M}$ *is a matroid. Let* π *be a policy that maps beliefs to actions in* I*,* $\pi(b) = \mathfrak{a}$*. Let* $Q_t^\pi(b, \mathfrak{a})$ *be the* t*-step value function for following policy* π*, that is,* $Q_t^\pi(b, \mathfrak{a}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathfrak{a}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathfrak{a}})$*. If for all policies* π*,* $Q_t^\pi(b, \mathfrak{a})$ *is non-negative, monotone and submodular in* $\mathfrak{a}$*, then for all* b*,*

$$V_t^G(b) \geq (\frac{1}{2})^{2t} V_t^*(b). \tag{4.58}$$

Proof. Proof in Appendix. □

While it is not necessary that the action space of an active perception POMDP satisfies the matroid constraints, however, even for constraints that involves an intersection over p different matroid greedy heuristic still guarantees a constant factor approximation [Fisher et al., 1978].

In this section, we extend our analysis to active perception tasks involving mobile sensors. We show if the value function is submodular then, backups based on greedy maximization subject to matroid constraints is guaranteed to compute a value function which has bounded error with respect to the optimal value function.

4.3 Experiments

To empirically evaluate greedy PBVI, we tested it on the problem of tracking either one or multiple people using a multi-camera system. The problem was extracted from the shopping mall dataset described in the experiments section of Chapter 3.

To address the blow-up in the size of the state space for multi-person tracking, we use a variant of *transfer planning* [Oliehoek et al., 2013]. We divide the multi-person problem into several *source* problems, one for each person, and solve them independently to compute $V_t(b) = \sum V^i(b_i)$, where $V^i(b_i)$ is the value of the current belief b_i about the i-th person's location. Thus $V_t^i(b_i)$ only needs to be computed once, by solving POMDP of the same size as that in the single-person setting. During action selection, $V_t(b)$ is computed using the current b_i for each person.

To learn a POMDP model from the dataset, we divided the continuous space into 20 cells ($|S| = 21$: 20 cells plus an external state indicating the person has left the shopping mall). Using the data, we learned a maximum-likelihood tabular transition function. However, we did not have access to the ground truth of the observed tracks so we constructed them using the overlapping regions of the camera.

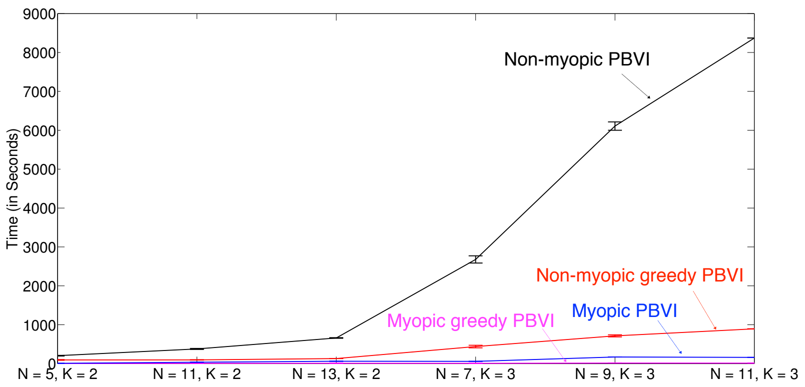


Figure 4.3: Runtimes for the different methods.

The reward function is described as a set of $|S|$ vectors, $\Gamma^{\rho} = \{\alpha_1 \ldots \alpha_{|S|}\}$, with $\alpha_i(s) = 1$ if $s = i$ and $\alpha_i(s) = 0$ otherwise. The initial belief is uniform across all states. We planned for horizon $h = 10$.

As baselines, we tested against regular PBVI and *myopic* versions of both greedy and regular PBVI that compute a policy assuming $h = 1$ and use it at each time step. Figure 4.3 shows runtimes under different values of n and k. Since multi-person tracking uses the value function obtained by solving a single-person POMDP, single and multi-person tracking have the same runtimes. These results demonstrate that greedy PBVI requires only a fraction of the computational cost of regular PBVI. In addition, the difference in the runtime grows quickly as the action space gets larger: for $n = 5$ and $k = 2$ greedy PBVI is twice as fast, while for $n = 11, k = 3$ it is approximately nine times as fast. Thus, greedy PBVI enables much better scalability in the action space. Figure 4.4, which shows the cumulative reward under different values of n and k for single-person (top) and multi-person (bottom) tracking, verifies that greedy PBVI's speed-up does not come at the expense of performance, as greedy PBVI accumulates nearly as much reward as regular PBVI. They also show that both PBVI and greedy PBVI benefit from non-myopic planning. While the performance advantage of non-myopic planning is relatively modest, it increases with the number of cameras and people, which suggests that non-myopic planning is important to making active perception scalable.

Furthermore, an analysis of the resulting policies showed that myopic and non-myopic policies differ qualitatively. A myopic policy, in order to minimize uncertainty in the next step, tends to look where it believes the person to be. By contrast, a non-myopic policy tends to pro-actively look where the person might go next, so as to more quickly detect her new location when she moves. Consequently, non-myopic policies exhibit less

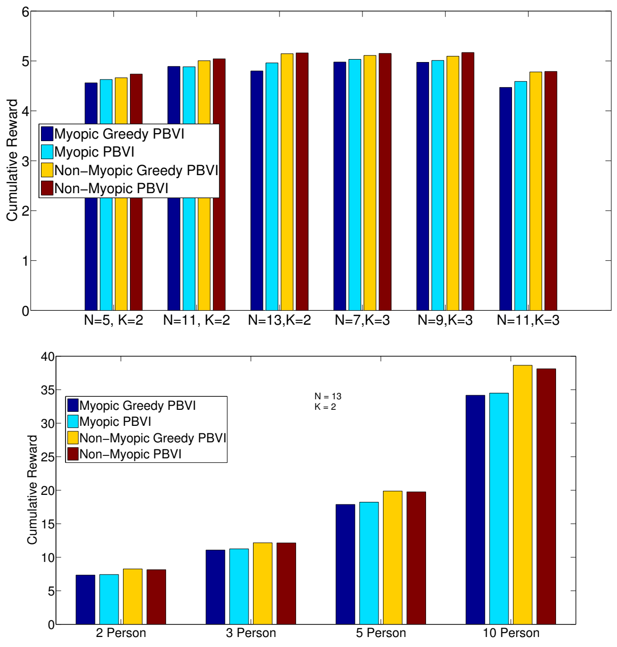


Figure 4.4: Cumulative reward for single-person (top) and multi-person (bottom) tracking.

fluctuation in belief and accumulate more reward, as illustrated in Figure 4.5. The blue lines mark when the agent chooses the camera that can observe the cell occupied by the person. The red line plots the max of the agent's belief. The difference in fluctuation in belief is evident, as the max of the belief often drops below 0.5 for the myopic policy but rarely does so for the non-myopic policy.

4.4 Conclusions & Future Work

In this chapter, we addressed the problem of scaling the POMDP planning in the combinatorial action space of the active perception POMDP. We modelled the action space of the active perception POMDP as selecting k out of n sensors, where k is the maximum number of sensors allowed by the resource constraints. Recent POMDP solvers enable scalability in the state space. However, for active perception, as the number of sensors grow, the action space grows exponentially. We proposed greedy PBVI, a POMDP planning method, that improves scalability in the action space of a POMDP. While we do not directly address the scaling in the observation space, we believe recent ideas on factorization of observation space [Veiga et al., 2014] can be combined with our approach

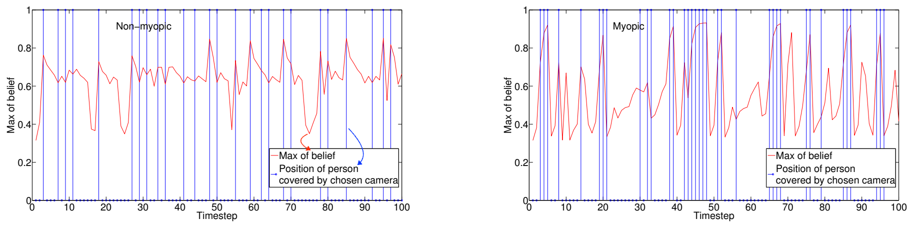


Figure 4.5: Behaviour of myopic vs. non-myopic policy.

to improve scalability in state, action and observation space to solve active perception POMDPs.

By exploiting the theory of submodularity, we showed that the value function computed by greedy PBVI is guaranteed to have bounded error. Specifically, we extend Nemhauser's result on greedy maximization of submodular functions to long-term planning. To apply these results to the active perception task, we showed that under certain conditions the value function of an active perception POMDP is submodular. One such condition requires that the future series of observations be independent of each other given the state. While this is a strong condition, it is only a sufficient condition and may not be a necessary one. Thus, one line of future work is to attempt to relax this condition for proving the submodularity of the value function. Finally, we showed that, even with a PWLC approximation to the true value function, which is submodular, the error in the value function computed by greedy PBVI remains bounded, thus enabling us to compute efficiently value functions for active perception POMDP.

Greedy PBVI is ideally suited for active perception POMDPs for which the value function is submodular. However, in real-life situations submodularity of the value function might not always hold. For example, in our setting when there is occlusion, it is possible for combinations of sensors that when selected together yield higher utility than the sum of their utilities when selected individually. Similar cases can arise when mobile robots are trying to sense the best point of view to observe a scene that is occluded. Thus in cases like this, greedy PBVI might not return the best solution.

Our empirical analysis established that the non-myopic policy beats the myopic one, the gain in certain cases is marginal. However, in cases involving mobile sensors and budgeted constraints, non-myopic policies become critically important. Finally, experiments on a real-world dataset showed that the performance of greedy PBVI is similar to the existing methods but requires only a fraction of the computational cost, leading to much better scalability for solving active perception tasks.

5 PAC Planning

In the previous chapter we proved that the value function of the active perception POMDP is submodular if the immediate belief-based reward is defined as the negative belief entropy. The experiments showed that if the state of the world is independent of the actions of the agent then the gain of the non-myopic planning is only marginal. Thus, in this chapter we focus on greedy maximization for submodular function maximization. We start with a small background on greedy maximization for submodular function maximization, followed by why the state-of-the-art methods are not directly applicable for large real world active perception problems, the main algorithm we propose, its analysis and finally the experiments.

The role of greedy maximization for maximizing submodular functions is pivotal. Greedy maximization has a lower computational cost than full maximization and in practice is observed to perform almost as well. However, for many large real-world problems, even greedy maximization can be too computationally expensive [Mirzasoleiman et al., 2015]. To accelerate greedy maximization Minoux [1978] proposed lazy greedy maximization which prunes elements whose marginal gains on the last iteration ensures that their marginal gains on the current iteration cannot be maximal. *Stochastic greedy maximization* [Mirzasoleiman et al., 2015] provides further speed-ups by evaluating the marginal gains only of a randomly sampled subset of elements at each iteration. Other variations [Wei et al., 2014, Badanidiyuru and Vondrák, 2014] also aim to minimize the number of marginal gain computations.

However, these methods assume it is computationally feasible to exactly compute the function being maximized and thus the marginal gain. In many settings, the exact computation of the submodular function is not possible; instead the algorithm has access only to the noisy estimates of the function being maximized [Singla et al., 2016, Satsangi et al., 2016b]. For example, in the sensor selection, the agent selects a subset of sensors such that it maximizes the negative conditional entropy over the position of a person in a shopping mall or airport. Computing conditional entropy involves an expectation over the entropy of the posterior beliefs about the hidden state. When surveilling large areas like shopping malls, exactly computing the entropy of a single posterior belief becomes infeasible, let alone an expectation over them. In fact, even maintaining an exact probability distribution over a person's position in an airport or shopping mall is not possible; thus, even computing the exact true entropies of the posterior beliefs is not possible in such settings.

Thus, in this chapter, we present a new algorithm called *probably approximately correct greedy maximization* [Satsangi et al., 2016a,b] for submodular function maximization. Rather than assuming access to a submodular function Q itself, we assume access only to confidence bounds on Q. In particular, we assume that these bounds are cheaper to compute than Q and are *anytime*, i.e., we can tighten them by spending more computation time, e.g., by generating additional samples. Inspired by lazy greedy maximization, our method uses confidence bounds to prune elements, thereby avoiding the need to further tighten their bounds. Furthermore, we provide a PAC analysis [Valiant, 2013] that shows that, with high probability, our method returns an approximately optimal set.

Given an unbiased estimator of Q, it is possible to use concentration inequalities like Hoeffding's inequality [Hoeffding, 1963] to obtain the confidence bounds needed by PAC greedy maximization. Unfortunately, many applications, such as sensor placement and decision tree induction, require information-theoretic definitions of Q such as information gain, negative conditional entropy, etc. These definitions depend on computing entropies over posterior beliefs, which are difficult to estimate in an unbiased way, especially without supplementary computations [Paninski, 2003]. The absence of an unbiased estimator renders Hoeffding's inequality inapplicable and makes it hard to obtain computationally *cheap* confidence bounds on conditional entropy [Nowozin, 2012a, Loh and Nowozin, 2013]. Therefore, we propose novel, cheap confidence bounds on conditional entropy.

Finally, we apply PAC greedy maximization with these new confidence bounds for the sensor selection to track people in a shopping mall. Our empirical results demonstrate that our approach performs comparably to greedy and stochastic greedy maximization, but at a fraction of the computational cost, leading to much better scalability.

5.1 Problem Setting

We consider a variation on submodular function maximization in which evaluating the submodular function Q, and therefore the marginal gain, is prohibitively expensive, rendering greedy and lazy greedy maximization (and other variants) inapplicable. Instead, we assume access to computationally cheap confidence bounds on Q stated in following assumption:

Assumption 1. *We assume access to anytime upper and lower confidence bounds on $Q(\mathcal{A})$ and a* `tighten`$(\mathcal{A}, t)$ *procedure that for all $\mathcal{A} \in \mathcal{A}^+$ that takes in as input arguments $\mathcal{A}$ and t (t is a positive integer) and returns $U_t(\mathcal{A})$ and $L_t(\mathcal{A})$ such that with probability $1 - \frac{\delta_l}{nt(t+1)}$, $L_t(\mathcal{A}) \leq Q(\mathcal{A})$ and with probability $1 - \frac{\delta_u}{nt(t+1)}$, $U_t(\mathcal{A}) \geq Q(\mathcal{A})$, for some fixed value of δ_l and δ_u. Also, we assume that the lower and upper confidence bounds L_t and U_t are monotonically increasing and decreasing respectively, that is, $L_t \leq L_{t'}$ and $U_t \geq U_{t'}$ for $t' > t$. (Here n is the size of $\mathcal{X} = \{1, 2, \ldots, n\}$ and $\mathcal{A}$ is a subset of $\mathcal{X}$ of size less than or equal to k.)*

These assumptions are satisfied in many settings where Q is too expensive to compute exactly. For example, if $Q(\mathcal{A}) = \mathbb{E}[X|\mathcal{A}]$ for some random variable X, then $\hat{Q}(\mathcal{A}) = \frac{1}{N}(\sum_{i=1}^{N} x_i)$, where the x_i's are i.i.d. samples of X, is an unbiased estimator of $\hat{Q}(\mathcal{A})$ for which U_t and L_t can easily be constructed using, e.g., Hoeffding's inequality. According to Hoeffding's inequality for $x_i \in [0, 1]$,

$$\Pr(|Q(\mathcal{A}) - \mathbb{E}[\hat{Q}(\mathcal{A})]| \geq \epsilon) \leq 2e^{(-2\epsilon^2 N)}. \tag{5.1}$$

Using Hoeffding's inequality, L_t and U_t can be constructed as: with probability $1 - \frac{\delta_l}{t(t+1)}$, $L_t(\mathcal{A}) = \hat{Q}(\mathcal{A}) - \sqrt{\frac{1}{2N}\log(\frac{2t(t+1)}{\delta_l})} \leq Q(\mathcal{A})$ is true and that with probability $1 - \frac{\delta_u}{t(t+1)}$, $U_t(\mathcal{A}) = \hat{Q}(\mathcal{A}) + \sqrt{\frac{1}{2N}\log(\frac{2t(t+1)}{\delta_u})} \geq Q(\mathcal{A})$. Furthermore, `tighten` procedure can tighten these lower and upper bounds by spending more computation, thereby, using more samples (higher N) to compute $\hat{Q}$. However, we specifically do *not* assume access to an unbiased estimator of Q. Instead, we seek an algorithm that performs submodular function maximization given only U_t, L_t, and `tighten`.

The absence of a computationally cheap unbiased estimator of Q arises in many settings, especially in the active perception tasks in which Q is defined using information-theoretic metrics such as *information gain* or *negative conditional entropy*. For example, in the sensor selection task, the agent must select a subset of sensors $\mathcal{A}$ of size k from a set of n available sensors, $\mathcal{X} = \{1, 2, \ldots, n\}$ such that it minimizes the uncertainty in the belief b of the agent. Formally, the submodular function Q in this case is the information gain (G) of selecting a subset of sensors:

$$G(\mathcal{A}) = H_b(s) - \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) H_{b_{\mathbf{z}}^{\mathcal{A}}}(s), \tag{5.2}$$

where Ω is the set of all possible values of observation $\mathbf{z}$ that can come from sensors present in the set $\mathcal{A}$ and $H_{b_{\mathbf{z}}^{\mathcal{A}}}(s)$ is the entropy of posterior belief obtained after selecting $\mathcal{A}$ and observing $\mathbf{z}$. Since $H_b(s)$ is independent of $\mathcal{A}$, we omit it for the rest of this chapter ($H_b(s)$ is necessary for G to be non-negative) and define $Q(\mathcal{A})$ as:

$$Q(\mathcal{A}) = -\sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A}) H_{b_{\mathbf{z}}^{\mathcal{A}}}(s). \tag{5.3}$$

Unfortunately, when there are many possible states and actions, maintaining an exact belief over s is intractable, which in turn, makes computing $H_{b_{\mathbf{z}}^{\mathcal{A}}}(s)$ intractable. Moreover, $H_{b_{\mathbf{z}}^{\mathcal{A}}}(s)$ is not just intractable but it is also difficult to efficiently estimate [Paninski, 2003, Nowozin, 2012a, Schürmann, 2004]. Since Q cannot be exactly computed (especially without paying an extra computational cost), greedy maximization is not applicable any more.

Therefore, in the next section we propose a new fundamentally different method that requires only U_t, L_t, and `tighten`. To solve sensor selection in particular, we also need cheap anytime implementations of U_t and L_t for conditional entropy, which we propose in Section 5.4.

5.2 PAC Greedy Maximization

In this section, we propose *probably approximately correct greedy maximization*, which enables an agent to perform submodular function maximization without ever computing Q exactly. The main idea is to use U and L to prune elements that with high probability do not maximize marginal gain.

Our approach is inspired by lazy greedy maximization. To see how, it is helpful to view lazy greedy maximization (also described in Chapter 2) as a pruning method that in each iteration maintains a priority queue of elements with their marginal gains on the previous iteration as their priorities and by terminating an iteration before the priority queue is empty it effectively prunes each element whose upper bound (given by its marginal gain on the previous iteration) is lower than the maximum lower bound (given by the best marginal gain found so far on the current iteration).

PAC greedy maximization generalizes this idea in two ways. First, it accepts arbitrary upper and lower bounds. This makes it possible to replace the bounds used by lazy greedy maximization, which rely on the exact computation of marginal gain, with cheaper ones. Second, it uses confidence bounds instead of hard bounds. By tolerating a small probability of error, our approach can prune more aggressively, enabling large speed-ups while maintaining a PAC bound.

Algorithm 4 `pac-greedy-max`$(\texttt{tighten}, \mathcal{X}, k, \epsilon_1)$

```
$\mathcal{A}^P \leftarrow \emptyset$
**for** $m = 1$ *to* $k$ **do**
    $\mathcal{A}^P \leftarrow \mathcal{A}^P \cup \texttt{pac-max}(\texttt{tighten}, \mathcal{X}, \mathcal{A}^P, \epsilon_1)$
**end for**
return $\mathcal{A}^P$
```

Algorithm 4 shows the main loop, which simply adds at each iteration the element selected by the `pac-max` subroutine. The role of `pac-max` is to return an element i^P such that it is ϵ-optimal with probability $1 - (\delta_l + \delta_u)$. Algorithm 5 shows this subroutine. In each iteration of the outer while loop, `pac-max` examines each of these elements and prunes it if its upper bound is not at least ϵ_1 greater than the max lower bound found so far. In addition, the element with the max lower bound is never pruned. If an element is not pruned, then its bounds are tightened. Algorithm 5 terminates when only one element remains.

Algorithm 5 is the closest to the algorithm Hoeffding's races, presented in Maron and Moore [1994, 1997] except that Maron and Moore [1994, 1997] propose explicitly to use Hoeffding's inequality to compute and tighten the upper and lower confidence bound[1]. Consequently, the analysis and convergence of the algorithms that they present are reliant on the application of Hoeffding's inequality and, thus, are applicable only for functions that can be estimated in an unbiased manner. This is in contrast to Algorithm 5 and its analysis that is given in the later section, both of which do not make any assumption on the way in which the upper and lower confidence bounds are generated or tightened.

5.3 PAC Bounds

In this section, we analyze PAC greedy maximization. With oracle access to Q, greedy maximization is guaranteed to find $\mathcal{A}^G$ such that $Q(\mathcal{A}^G) \geq (1 - e^{-1})Q(\mathcal{A}^*)$, if Q is monotone, non-negative and submodular [Nemhauser et al., 1978]. Since PAC greedy maximization does not assume oracle access to Q and instead works with cheap anytime

[1]The other *minute* differences between `pac-max` and Hoeffding's races are (a) the use of priority queue in `pac-max`, and (b) that Hoeffding's races do(es) not *explicitly* take into account the number of times `tighten` procedure was previously called as an input parameter to the `tighten` procedure.

Algorithm 5 `pac-max`$(\texttt{tighten}, \mathcal{X}, \mathcal{A}^P, \epsilon_1)$

```
1:  ▷ Input: pac-max takes as input access to tighten procedure; $\mathcal{X} = \{1, 2, \dots, n\}$
    original set of $n$ elements; $\mathcal{A}^P$ a subset of $\mathcal{X}$, in this case, $\mathcal{A}^P$ is the partial solution
    maintained by pac-greedy-max, $\epsilon$ a positive real number.
2:  $i^P \leftarrow 0$                                        ▷ element with max lower bound
3:  $\rho \leftarrow$ empty priority queue
4:  $t \leftarrow 0$                                          ▷ $t$ is the iteration number.
5:  $t = t + 1$
6:  for $i \in \mathcal{X} \setminus \mathcal{A}^P$ do
7:      $U_t(i), L_t(i) \leftarrow$ tighten$(\mathcal{A}^P \cup i, t)$          ▷ Here $U_t(i)$ and $L_t(i)$
8:                                                            denote the upper and lower
9:                                                            bound on $Q(\mathcal{A}^P \cup i)$.
10:                                               For conciseness, $U_t(i)$ and $L_t(i)$ denote
11:                                               $U_t(\mathcal{A}^P \cup i)$ and $L_t(\mathcal{A}^P \cup i)$ respectively.
12:     $\rho$.enqueue$(i, U_t(i))$                               ▷ initial priority
13:     $i^P \leftarrow \arg\max_{j \in \{i, i^P\}} L_t(j)$          ▷ Element with maximum lower bound
14: end for
15: while $(\rho$.length$() > 1)$ do
16:     $\rho' \leftarrow$ empty priority queue
17:     $t = t + 1$
18:     $i^P$-in-queue = True                          ▷ Set flag to check if $i^P$ is still in $\rho$
19:     while $\neg\rho$.empty$()$ do
20:         $i \leftarrow \rho$.dequeue$()$                     ▷ Element with maximum upper bound
21:         if $i = i^P$ then                                ▷ Check if $i^P$ is still in $\rho$
22:             $i^P$-in-queue = False
23:         end if
24:         if $(i = i^P) \vee (U_t(i) \geq L_t(i^P) + \epsilon_1)$ then
25:                                                           ▷ Prune $i$ that does not have
26:                                                upper bound that is $\epsilon$ greater than $L_t(i^P)$
27:                                                otherwise call tighten procedure for $i$.
28:             $U_t(i), L_t(i) \leftarrow$ tighten$(\mathcal{A}^P \cup i, t)$
29:             $i^P \leftarrow \arg\max_{j \in \{i, i^P\}} L_t(j)$
30:             $\rho'$.enqueue$(i, U_t(i))$
31:         else if $i^P$-in-queue = True then
32:             Continue
33:         else
34:             Break Inner While Loop
35:         end if
36:     end while
37:     $\rho \leftarrow \rho'$
38: end while
39: return $i^P$
```

confidence bounds on Q, we prove a PAC bound for PAC greedy maximization. In particular, we prove that under the same conditions, PAC greedy maximization finds a solution $\mathcal{A}^P$ such that, with high probability, $Q(\mathcal{A}^P)$ is close to $Q(\mathcal{A}^*)$.

We can now prove a lemma that shows that, with high probability, the marginal gain of the element picked by $\texttt{pac-max}(\texttt{tighten}, \mathcal{X}, \mathcal{A}^P, \epsilon_1)$ is ϵ-optimal with high probability.

Lemma 7. *Let* $\mathcal{X} = \{1, 2, \ldots, n\}$*,* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$*, and* $Q : 2^{\mathcal{X}} \to \mathbb{R}_+$*. If* $\texttt{pac-max}(\texttt{tighten}, \mathcal{X}, \mathcal{A}, \epsilon_1)$ *terminates and returns* i^P*, and if Assumption 1 holds, then with probability (at least)* $1 - \delta_1$*,*

$$Q(\mathcal{A}^P \cup i^P) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1, \tag{5.4}$$

where $\delta_1 = \delta_u + \delta_l$ *and* $i^* = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}^P} Q(\mathcal{A}^P \cup i)$ *and* $\mathcal{A}^P$ *is any set in* $\mathcal{A}^+$*.*

Proof. If $\texttt{pac-max}$ returns $i^P = i^*$, then the Lemma holds trivially, since

$$Q(\mathcal{A}^P \cup i^*) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{5.5}$$

For the case that $\texttt{pac-max}$ returns $i^P \neq i^*$, we provide the proof here.

Lets assume that $\texttt{pac-max}$ returns i^P after T (T not known or fixed) total iterations. That is, t goes from 0 to T.

We prove this Lemma in two parts:

- In part A, we show that if the assumed confidence intervals $U_t(i)$ and $L_t(i)$ hold for all t and i, then $\texttt{pac-max}$ returns an ϵ-optimal element. That is, if

$$U_t(i) \geq Q(\mathcal{A}^P \cup i) \tag{5.6}$$

  and

$$L_t(i) \leq Q(\mathcal{A}^P \cup i) \tag{5.7}$$

  is true for all $i \in \mathcal{X}$ and $t \in \{1, 2, \ldots, T\}$, then

$$Q(\mathcal{A}^P \cup i^P) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{5.8}$$

- In part B, we compute the probability that the confidence intervals hold for all i

and t, that is, the probability,

$$\begin{aligned}
\Pr\Big(&((U_t(i) \geq Q(\mathcal{A}^P \cup i) \wedge L_t(i) \leq Q(\mathcal{A}^P \cup i) \text{ for } i = 1, t = 1)\wedge \\
&((U_t(i) \geq Q(\mathcal{A}^P \cup i) \wedge L_t(i) \leq Q(\mathcal{A}^P \cup i) \text{ for } i = 1, t = 2)\wedge \\
&((U_t(i) \geq Q(\mathcal{A}^P \cup i) \wedge L_t(i) \leq Q(\mathcal{A}^P \cup i) \text{ for } i = 1, t = 3)\wedge \\
&\ldots \\
&((U_t(i) \geq Q(\mathcal{A}^P \cup i) \wedge L_t(i) \leq Q(\mathcal{A}^P \cup i) \text{ for } i = 2, t = 1)\wedge \\
&((U_t(i) \geq Q(\mathcal{A}^P \cup i) \wedge L_t(i) \leq Q(\mathcal{A}^P \cup i) \text{ for } i = 2, t = 2)\wedge \\
&\ldots \\
&\ldots \\
&((U_t(i) \geq Q(\mathcal{A}^P \cup i) \wedge L_t(i) \leq Q(\mathcal{A}^P \cup i) \text{ for } i = n, t = T)\Big)
\end{aligned} \tag{5.9}$$

We show that this probability is lower bounded by $1 - \delta_1$ if Assumption 1 holds. Here $\delta_1 = \delta_l + \delta_u$ is the probability that the confidence intervals (UCI or LCI) are not true at least once in T iterations for at least one i.

Part A: To show, if for all i and t, confidence intervals hold, that is, $U_t(i) \geq Q(\mathcal{A}^P \cup i) \geq L_t(i)$ is true for all i, t, then

$$Q(\mathcal{A}^P \cup i^P) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{5.10}$$

At any iteration $t \in \{1, 2, \ldots, T\}$ `pac-max` maintains the element with max lower bound. Lets denote the element with max lower bound at the end of iteration t by i_t^P. Since i^* was eliminated ($i^P \neq i^*$), thus at some iteration t', its upper bound was lower than the maximum lower bound + ϵ_1 (lets say of element $i_{t'}^P$). Let $L_{t'}(i)$ denote the lower bound (and $U_{t'}(i)$ denote the upper bound) at iteration t' of element i, then, the lower bound of element $i_{t'}^P$ is greater than the upper bound of i^* minus ϵ at some iteration t':

$$L_{t'}(i_{t'}^P) \geq U_{t'}(i^*) - \epsilon_1. \tag{5.11}$$

Since (a) we have assumed that L_t is monotonically increasing, and (b) `pac-max` returns i^P, this implies on termination the element with maximum lower bound is i^P, and this lower bound on i^P has to be greater than $L_{t'}(i_{t'}^P)$, since i^P was able to replace $i_{t'}^P$ at an iteration $t > t'$.

$$L_T(i^P) \geq L_{t'}(i_{t'}^P) \tag{5.12}$$

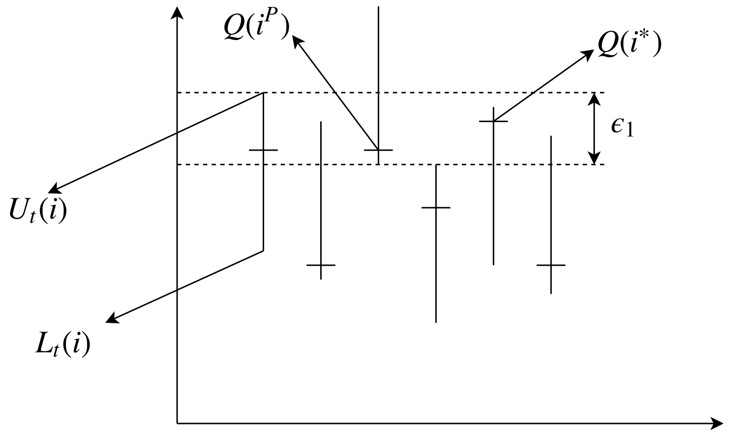


Figure 5.1: Figure illustrating the terminating condition for `pac-max`. `pac-max` returns i^P, the element with max lower bound, after T (not known or fixed) total iterations. In the figure notations $\mathcal{A}^P$ is omitted for clarity. The small horizontal lines represent the true value of an element and the ends of the vertical lines represent the upper and lower bounds on an element. The figure illustrates that if that for all i and for all $t \in \{1, 2, \dots T\}$, the confidence intervals $U_t(\mathcal{A}^P \cup i) \geq Q(\mathcal{A}^P \cup i) \geq L_t(\mathcal{A}^P \cup i)$ hold then $Q(\mathcal{A}^P \cup i^P) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1$.

Combining (5.11), and (5.12), we get,

$$L_T(i^P) \geq L_{t'}(i^P_{t'}) \geq U_{t'}(i^*) - \epsilon_1 \tag{5.13}$$

If confidence interval hold for all t and i, then $U_{t'}(i^*) \geq Q(\mathcal{A}^P \cup i^*)$ and $Q(\mathcal{A}^P \cup i^P) \geq L_T(i^P)$, this implies,

$$Q(\mathcal{A}^P \cup i^P) \geq L_T(i^P) \geq U_{t'}(i^*) - \epsilon_1 \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{5.14}$$

Figure 5.1 shows an example of terminating condition for `pac-max` and that $Q(\mathcal{A}^P \cup i^P)$ is within ϵ range of $Q(\mathcal{A}^P \cup i^*)$ if the confidence intervals hold.

Thus, if confidence intervals $U_t(i)$ and $L_t(i)$ hold for all t and i, then,

$$Q(\mathcal{A}^P \cup i^P) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon. \tag{5.15}$$

Part B: In part B, we compute the probability that the upper and lower confidence intervals hold for all t and i. The reasoning in this part follows from the proof presented in Maron and Moore [1994, 1997].

We will be using extensively the union bound during this part of the proof. According to union bound the probability of the union of events $A_1, A_2, \ldots, A_l$ is bounded by the sum of their individual probabilities:

$$\Pr(A_1 \vee A_2 \vee \cdots \vee A_l) = \Pr(\bigcup_l A_l) \leq \sum_l \Pr(A_l) \tag{5.16}$$

To compute the probability that the confidence intervals hold for all i for all t, we observe that this probability is equal to 1 - the probability that the confidence intervals do not hold for at least one i during at least one iteration t. So we want to compute the probability:

$$\begin{aligned}\Pr(&\text{upper confidence interval (UCI) OR lower confidence interval (LCI)}\\ &\text{do not hold for at least one } i \text{ for at least one value of } t\ ).\end{aligned} \tag{5.17}$$

To compute this probability, lets start with the probability of the confidence interval to NOT hold for one particular $i = i'$ at one particular iteration $t = t'$. We have assumed that at iteration t, $\texttt{tighten}(\mathcal{A}^P \cup i, t)$ returns $U_t(i)$ (and $L_t(i)$) such that with probability $1 - \frac{\delta_u}{nt(t+1)}, U_t(i) \geq Q(\mathcal{A}^P \cup i)$ (this condition means that upper confidence interval holds) is true. This implies that for a particular $i = i'$ at iteration $t = t'$, the probability of upper confidence interval to not hold is (less than) $\frac{\delta_u}{nt'(t'+1)}$ and the probability that lower confidence interval ($L_t(i) \leq Q(\mathcal{A}^P \cup i)$) does not hold is (less than) $\frac{\delta_l}{nt'(t'+1)}$.

Then,

$$\begin{aligned}\Pr\Big(\text{UCI OR LCI is not true for a particular } i(=i') \text{ at a particular iteration } t(=t')\Big) \leq\\ \Big(\Pr(\text{UCI is not true for } i' \text{ at iteration } t'\ )+\\ \Pr(\text{LCI is not true for } i' \text{ at iteration } t')\Big)\end{aligned} \tag{5.18}$$

By Assumption 1,

$$\Pr(\text{UCI is not true for } i' \text{ at iteration } t'\ ) \leq \frac{\delta_u}{nt'(t'+1)}, \tag{5.19}$$

and

$$\Pr(\text{LCI is not true for } i' \text{ at iteration } t'\ ) \leq \frac{\delta_l}{nt'(t'+1)}, \tag{5.20}$$

Thus, using union bound,

$$\Pr\Big(\text{ UCI OR LCI is not true for } i' \text{ at iteration } t' \Big) \\ \leq \frac{\delta_u + \delta_l}{nt'(t'+1)}. \tag{5.21}$$

Again using union bound, probability that confidence intervals do NOT hold for i' at $t = 1$ OR $t = 2$ OR $t = 3$ OR $\ldots$ OR $t = T$ is bounded by sum of individual (probability that confidence intervals do NOT hold for i' for $t = 1$) + (probability that confidence intervals do NOT hold for i' for $t = 2$) $+ \ldots$ (series ends at $t = T$).

From equation (5.21), we know the probability that confidence intervals do not hold for i' at iteration t' is less than $\frac{\delta_u+\delta_l}{nt'(t'+1)}$.

$$\Pr((\text{UCI or LCI is not true for } i' \text{ at least once in } t \in \{1, 2, \ldots, T\}) \\ \leq \sum_{t=1}^{T} \frac{\delta_l + \delta_u}{nt(t+1)} \tag{5.22}$$

The sum over t of the series $\frac{1}{t(t+1)}$, that is $S_T = \sum_{t=1}^{T} \frac{1}{t(t+1)}$ is bounded by $(1 - \frac{1}{T+1})$ for a finite T and even as $\lim T \to \infty$, $\lim_{T\to\infty} \sum_{t=1}^{T} \frac{1}{t(t+1)}$ is bounded by 1[2].

Thus, using union bound,

$$\Pr((\text{UCI or LCI is not true for } i' \text{ at least once in } t \in \{1, 2, \ldots, T\}) \\ \leq \frac{\delta_u + \delta_l}{n} \tag{5.25}$$

Again we can use union bound to show that the probability that the confidence intervals do not hold for $i = 1$ OR $i = 2$ OR $\ldots$ OR $i = n$, at least once in $t \in \{1, 2, \ldots, T\}$ is bounded by the (probability that the confidence interval do not hold for $i = 1$ at least once in $t \in \{1, 2, \ldots, T\}$) + (probability that the confidence interval do not hold for $i = 2$ at least once in $t \in \{1, 2, \ldots, T\}$) + $\ldots$ + (probability that the confidence interval do not hold for $i = n$ at least once in $t \in \{1, 2, \ldots, T\}$) .

Since for each i the probability that the confidence interval do not hold for i at least

[2]$\sum_{t=1}^{T} \frac{1}{t(t+1)}$ can be expressed as:

$$\sum_{t=1}^{T} \frac{1}{t(t+1)} = \sum_{t=1}^{T} [\frac{1}{t} - \frac{1}{t+1}] = [1 - \frac{1}{2} + \frac{1}{2} - \frac{1}{3} + \frac{1}{3} - \cdots - \frac{1}{T+1}] \tag{5.23}$$

$$= [1 - \frac{1}{T+1}]. \tag{5.24}$$

once in $t \in \{1, 2, \ldots, T\}$ is bounded by $\frac{\delta_u + \delta_l}{n}$. Taking the sum over n terms yields:

$$\Pr(\text{UCI or LCI is not true for at least one } i \text{ at least once in } t \in \{1, 2, \ldots, T\}) \\ \leq \delta_u + \delta_l. \tag{5.26}$$

Finally, since probability that confidence intervals hold for all i for all $t \in \{1, 2, \ldots, T\}$ = 1 - probability confidence intervals do not hold at least for one $t \in \{1, 2, \ldots, T\}$ for at least one i, we can write that with probability $1 - \delta_1$,

$$Q(\mathcal{A}^P \cup i^P) \geq Q(\mathcal{A}^P \cup i^*) - \epsilon_1. \tag{5.27}$$

□

Comparison to best-arm identification algorithms

As noted before, pac-max accomplishes a similar goal as the existing best-arm identification algorithms in the multi-armed bandit literature. The fundamental difference between our approach and the existing algorithms such as UCB-E [Audibert and Bubeck, 2010], LUCB [Kalyanakrishnan et al., 2012], Successive Elimination [Even-Dar et al., 2006], Exponential-Gap [Karnin et al., 2013], Bernstein's races [Mnih et al., 2008] and many others is in the treatment of the reward by the analysis of the respective algorithm. To the best of our knowledge, the convergence analysis and/or the sample complexity of all these algorithms necessarily assume(s) the expected reward associated with each arm to be a sum/mean (expected mean) of N (where N is a finite integer) i.i.d (independent and identically distributed) random variables, that makes the application of concentration inequalities like Hoeffding's inequality or Chernoff bound or (empirical) Bernstein's bound possible[3]. For example, Jamieson et al. [2014] presents lil'UCB that identifies the arm with the largest mean in a multi-armed bandit problem. However, note that the exploration term that lil'UCB proposes is quite specific and depends on Chernoff's bound for its exact derivation, and the proof for sample complexity of lil'UCB makes generous use of Hoeffding's inequality. In the same paper the authors compare their work with other popular best-arm identification algorithms such as LUCB [Kalyanakrishnan et al., 2012], Successive Elimination [Even-Dar et al., 2006], and Exponential-Gap Elimination [Karnin et al., 2013]. LUCB relies on Hoeffding's inequality for its convergence and sample complexity analysis (proof of Theorem 1 in [Kalyanakrishnan et al., 2012]).

[3]We made the same point before (5.1) where we noted that the mean of N i.i.d random variables is guaranteed to be an unbiased estimator. However, we do not treat the rewards or function that we want to maximize as a sum of N i.i.d random variables. This is mainly because for entropy estimation such an assumption does not hold.

Even-Dar et al. [2006] present two versions of successive elimination algorithms, one where the expected reward of each arm (a bandit arm is a element in $\mathcal{X} = \{1, 2, \ldots, n\}$ is our setting) is known and the other version when the expected reward is not known (they call their algorithms as one with known biases and unknown biases, but this is not be confused with the bias of an estimator), however, Hoeffding's inequality is central to the sample complexity and parameters of both of these algorithms. Analysis and terminating condition of Bernstein's races [Mnih et al., 2008] depends on the application of empirical Bernstein's bound that, in turn, is applicable to sum of N i.i.d random variables.

The analysis of pac-max makes no such assumption about the rewards or functions it takes as input. This is mainly because we seek an algorithm (and analysis) that can be applied to functions that cannot be estimated in an unbiased manner, for example, entropy, and thus do not necessarily satisfy the assumption that they can be expressed as a sum of N i.i.d random variables. For the sake of argument one can assume entropy to be sum of N i.i.d random variable but clearly that is not true. In short, if the estimator is biased then the analysis and the associated results with algorithms such as UCB-E, LUCB, Successive elimination, etc. cannot be directly used.

The cost of abstract upper and lower confidence bounds (and not tightening them with one of the concentration inequalities) is that it is difficult to comment on (a) the sample complexity of pac-max (b) the convergence of pac-max. In the current state, it is possible that for very low values of ϵ and for problems where the upper and lower bounds are such that they do not tighten beyond a certain point that pac-max may not converge. On the other hand, this was not a problem in practice as pac-max was observed to converge every time after a sufficient amount of parameter tuning was performed that involved running pac-max for multiple values of ϵ. In principle it might be possible to have a set of terminating conditions for pac-max that can guarantee its convergence. For example, pac-max can also be viewed as a probabilistic version of branch and bound approaches [Boyd and Mattingley, 2007]. As a starting point Boyd and Mattingley [2007] propose a number of conditions on the upper and lower bounds to show convergence of branch and bound approaches. However, since our motivation is real-time sensor selection and that we were able to observe the useful convergence of pac-max in practice, we leave the setting of the assumptions and conditions required for analysing the sample complexity and convergence of pac-max as future work.

Next, we show that, if in each iteration of greedy maximization an ϵ-optimal element is returned with probability $1 - \delta$, then greedy maximization returns a set that is $k\epsilon$-optimal with probability $1 - k\delta$, where k is the number of iteration greedy maximization is run for.

Theorem 12. *Let $\mathcal{X} = \{1, 2, \dots n\}$, $\mathcal{A}^+ : \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$, and $Q : 2^{\mathcal{X}} \to \mathbb{R}_+$ be a non-negative, monotone and submodular in $\mathcal{X}$. if Assumption 1 holds and if* `pac-max` *terminates every time it is called then, with probability $1 - \delta$,*

$$Q(\mathcal{A}^P) \geq (1 - e^{-1})Q(\mathcal{A}^*) - \epsilon, \tag{5.28}$$

where $\mathcal{A}^P = \texttt{pac-greedy-max}(\texttt{tighten}, \mathcal{X}, k, \epsilon_1)$, $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} Q(\mathcal{A})$, $\delta = k\delta_1$, and $\epsilon = k\epsilon_1$. (Here $\delta_1 = \delta_l + \delta_u$, and δ_l and δ_u are defined in Assumption 1.)

Proof. Let $\mathcal{A}^P_m$ denote the subset returned by `pac-greedy-max` after m iterations, that is, $\mathcal{A}^P_m = \texttt{pac-greedy-max}(\texttt{tighten}, \mathcal{X}, m, \epsilon_1)$ and let $\{i^*_1, i^*_2, \dots i^*_k\}$ (arbitrary order), be the k elements of $\mathcal{A}^*$. We denote the marginal gain of adding i to a subset $\mathcal{A}$ as:

$$\Delta_Q(i|\mathcal{A}) = Q(\mathcal{A} \cup i) - Q(\mathcal{A}). \tag{5.29}$$

To prove Theorem 12, we first prove an intermediate result that we will use later in the proof: Starting with the statement of Lemma 7, with probability $1 - \delta_1$,

$$Q(\mathcal{A}^P_m \cup i^P) \geq Q(\mathcal{A}^P_m \cup i^*) - \epsilon_1, \tag{5.30}$$

where $i^* = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}^P_m} Q(\mathcal{A}^P_m \cup i)$ and $i^P = \texttt{pac-max}(\texttt{tighten}, \mathcal{X}, \mathcal{A}^P_m, \epsilon_1)$.

This implies the following set of inequalities the explanation of which is provided after them: with probability $1 - \delta_1$,

$$Q(\mathcal{A}^P_m \cup i^P) \geq Q(\mathcal{A}^P_m \cup i^*) - \epsilon_1 \tag{5.31}$$

$$Q(\mathcal{A}^P_m \cup i^P) - Q(\mathcal{A}^P_m) \geq Q(\mathcal{A}^P_m \cup i^*) - Q(\mathcal{A}^P_m) - \epsilon_1 \tag{5.32}$$

$$\Delta_Q(i^P|\mathcal{A}^P_m) \geq \Delta_Q(i^*|\mathcal{A}^P_m) - \epsilon_1 \tag{5.33}$$

$$\Delta_Q(i^P|\mathcal{A}^P_m) \geq \frac{1}{|\mathcal{A}^* \setminus \mathcal{A}^P_m|} \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}^P_m} \Delta_Q(i|\mathcal{A}^P_m) - \epsilon_1 \tag{5.34}$$

$$\Delta_Q(i^P|\mathcal{A}^P_m) \geq \frac{1}{k} \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}^P_m} \Delta_Q(i|\mathcal{A}^P_m) - \epsilon_1. \tag{5.35}$$

Eq. (5.31) follows from Lemma 7, Eq. (5.32) is simple subtraction of $Q(\mathcal{A}^P_m)$ from both sides of inequality, Eq. (5.33) is re-writing (5.32) by using the definition of marginal gain as given in (5.29) (and in Chapter 2), Eq. (5.34) is true because $\Delta_Q(i^*|\mathcal{A}^P_m)$ is the maximum value of $\Delta_Q(i|\mathcal{A}^P_m)$ for all $i \in \mathcal{X} \setminus \mathcal{A}^P_m$. This implies it is definitely bigger than the average value of $\Delta_Q(i|\mathcal{A}^P_m)$ taken over $\mathcal{A}^* \setminus \mathcal{A}^P_m$ where $\mathcal{A}^*$ is a subset of $\mathcal{X}$. Eq.

(5.35) is true because $|\mathcal{A}^*| \leq k$.

The rest of the proof follows the same logic as Theorem 1 in Chapter 2, which is the same as the proof presented in [Krause and Golovin, 2014] for Nemhauser's original result on greedy maximization of submodular functions.

We present the following sets of inequalities and then provide the explanations for them below:

$$Q(\mathcal{A}^*) \leq Q(\mathcal{A}^* \cup \mathcal{A}_m^P) \tag{5.36}$$

$$= Q(\mathcal{A}_m^P) + \sum_{j=1}^{k} \Delta_Q(i_j^*|\mathcal{A}_m^P \cup \{i_1^*, i_2^*, \ldots, i_{j-1}^*\}) \tag{5.37}$$

$$\leq Q(\mathcal{A}_m^P) + \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_m^P} \Delta_Q(i|\mathcal{A}_m^P) \tag{5.38}$$

Equation (5.36) follows from monotonicity of Q, Eq. (5.37) is a straightforward telescopic sum, Eq. (5.38) is true because Q is submodular.

Eq. (5.35) says that with probability $1 - \delta_1$,

$$k(\Delta_Q(i^P|\mathcal{A}_m^P) + \epsilon_1) \geq \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_m^P} \Delta_Q(i|\mathcal{A}_m^P). \tag{5.39}$$

Using (5.39), (5.38) can be written as:

With probability $1 - \delta_1$,

$$Q(\mathcal{A}^*) \leq Q(\mathcal{A}_m^P) + k(\Delta_Q(i^P|\mathcal{A}_m^P) + \epsilon_1) \tag{5.40}$$

$$\leq Q(\mathcal{A}_m^P) + k(Q(\mathcal{A}_m^P \cup i^P) - Q(\mathcal{A}_m^P) + \epsilon_1) \tag{5.41}$$

$$\leq Q(\mathcal{A}_m^P) + k(Q(\mathcal{A}_{m+1}^P) - Q(\mathcal{A}_m^P) + \epsilon_1) \tag{5.42}$$

Eq. (5.40) follows from (5.38), (we just replaced $\sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_m^P} \Delta_Q(i|\mathcal{A}_m^P)$ with a greater quantity $k(\Delta_Q(i^P|\mathcal{A}_m^P) + \epsilon_1)$). Eq (5.41) follows from the definition of marginal gain in (5.29) and Eq. (5.42) is true because $\mathcal{A}_m^P \cup i^P$ is $\mathcal{A}_{m+1}^P$ by definition of $\mathcal{A}_m^P$ and i^P at the start of the proof.

Lets define $\beta_m = Q(\mathcal{A}^*) - Q(\mathcal{A}_m^P)$, then (5.42) can be written as: with probability

$1 - \delta_1$,

$$Q(\mathcal{A}^*) - Q(\mathcal{A}^P_m) \leq k[Q(\mathcal{A}^*) - Q(\mathcal{A}^P_m) - (Q(\mathcal{A}^*) - Q(\mathcal{A}^P_{m+1}) + \epsilon_1] \tag{5.43}$$

$$\beta_m \leq k[\beta_m - \beta_{m+1} + \epsilon_1] \tag{5.44}$$

$$\beta_{m+1} \leq \beta_m(1 - \frac{1}{k}) + \epsilon_1. \tag{5.45}$$

Substituting $m = 0$ in (5.45) gives, with probability $1 - \delta_1$,

$$\beta_1 \leq (1 - \frac{1}{k})\beta_0 + \epsilon_1 \tag{5.46}$$

Substituting $m = 1$ in (5.45) again gives, with probability $1 - \delta_1$,

$$\beta_2 \leq (1 - \frac{1}{k})\beta_1 + \epsilon_1 \tag{5.47}$$

Now combining (5.46) and (5.47) and using union bound as presented in equation (5.16) (explained below): with probability $1 - 2\delta_1$,

$$\beta_2 \leq (1 - \frac{1}{k})\Big[(1 - \frac{1}{k})\beta_0 + \epsilon_1\Big] + \epsilon_1 \tag{5.48}$$

$$\leq (1 - \frac{1}{k})^2\beta_0 + (2 - \frac{1}{k})\epsilon_1 \tag{5.49}$$

$$\leq (1 - \frac{1}{k})^2\beta_0 + 2\epsilon_1. \tag{5.50}$$

The logic behind using union bound here is that the inequality in (5.46) can fail with probability δ_1 and the inequality in (5.47) can fail with probability δ_1. If both the inequalities in (5.46) and (5.47) do not fail then (5.48) (and consequently (5.50) is definitely true. The probability that either of inequality in (5.46) or (5.47) fails is bounded by their sum of the probabilities that either inequality fails individually: $\delta_1 + \delta_1$. The probability of both inequality in (5.46) and (5.47) is true is $1 - 2\delta_1$.

Substituting $m = 2$ in (5.45) again gives, with probability $1 - \delta_1$,

$$\beta_3 \leq (1 - \frac{1}{k})\beta_2 + \epsilon_1 \tag{5.51}$$

Combining (5.51) with (5.50), and using union bound (we just explained how to use union bound here in the paragraph above (5.51)), with probability $1 - 3\delta_1$,

$$\beta_3 \leq (1 - \frac{1}{k})\beta_2 + \epsilon_1 \tag{5.52}$$

$$\leq (1 - \frac{1}{k})\Big[(1 - \frac{1}{k})^2\beta_0 + 2\epsilon_1\Big] + \epsilon_1 \tag{5.53}$$

$$\leq (1 - \frac{1}{k})^3\beta_0 + 3\epsilon_1. \tag{5.54}$$

Continuing like this for $m = 0$ to $k - 1$, we get, with probability $1 - k\delta_1$,

$$\beta_k \leq (1 - \frac{1}{k})^k\beta_0 + k\epsilon_1 \tag{5.55}$$

Now using the inequality that $1 - x \leq e^{-x}$ for all $x \in \mathbb{R}$, we get $1 - \frac{1}{k} \leq e^{\frac{-1}{k}}$, which implies, with probability $1 - k\delta_1$,

$$\beta_k \leq e^{\frac{-k}{k}}\beta_0 + k\epsilon_1 \tag{5.56}$$

Using definition of $\beta_k = Q(\mathcal{A}^*) - Q(\mathcal{A}^P)$ and $\beta_0 = Q(\mathcal{A}^*) - Q(\mathcal{A}_0^P)$, with probability $1 - k\delta_1$,

$$Q(\mathcal{A}^*) - Q(\mathcal{A}^P) \leq (e^{-1})[Q(\mathcal{A}^*) - Q(\mathcal{A}_0^P)] + k\epsilon_1 \tag{5.57}$$

Since $Q(\mathcal{A}_0^P) > 0$, with probability $1 - k\delta_1$,

$$Q(\mathcal{A}^*) - Q(\mathcal{A}^P) \leq (e^{-1})[Q(\mathcal{A}^*)] + k\epsilon_1 \tag{5.58}$$

$$Q(\mathcal{A}^P) \geq (1 - e^{-1})Q(\mathcal{A}^*) - k\epsilon_1 \tag{5.59}$$

$\square$

Theorem 12 proves that PAC greedy maximization, while assuming access only to anytime confidence bounds on Q, computes $\mathcal{A}^P$ such that with high probability $Q(\mathcal{A}^P)$ has bounded error with respect to $Q(\mathcal{A}^*)$. As PAC greedy maximization requires access to cheap upper and lower confidence bounds, in the next section, we propose such bounds for conditional entropy.

5.4 Conditional Entropy Bounds

In many settings, U_t and L_t can easily be constructed using, e.g., Hoeffding's inequality [Hoeffding, 1963] and `tighten` need only fold more samples into an estimate of Q.

However, Hoeffding's inequality only bounds the error between the estimate and the expected value of the estimator. This in turn bounds the error between the estimate and the true value only if the estimator is unbiased, i.e., the expected value of the estimator equals the true value.

We are interested in settings such as sensor selection, where Q is based on conditional entropy, that is computed by approximating the entropy of the posterior beliefs. However, a *naive* estimator of entropy of a discrete distribution is known to be unbiased especially in the absence of enough samples. Therefore, in this section, we propose novel, cheap confidence bounds on conditional entropy that work with a *naive* estimator of posterior entropies.

We start by defining the naive or maximum likelihood estimate of entropy. Given M samples, $\{s^1, s^2 \dots s^M\}$ from a discrete distribution $b(s)$, the *maximum likelihood estimator* (MLE) of $b(s)$ is:

$$\hat{b}(s) = \frac{1}{M} \sum_{j=1}^{M} \mathbb{1}(s^j, s), \tag{5.60}$$

where $\mathbb{1}(s^j, s)$ is an indicator function that is 1 if $s^j = s$ and 0 otherwise. The MLE of entropy is:

$$H_{\hat{b}}(s) \triangleq \sum_{s} \hat{b}(s) \log(\hat{b}(s)). \tag{5.61}$$

The naive estimator $H_{\hat{b}}(s)$ is known to be biased. This is because $H_{\hat{b}}(s)$ is a sum of estimates of $b(s)\log(b(s))$, i.e., it substitutes the quantity $b(s)\log(b(s))$ with an estimate $\hat{b}(s)\log(\hat{b}(s))$, the expected value of which is less than its true value (application of Jensen's inequality on the function $f(x) = -x\log(x)$ [Nowozin, 2012b]). Though $H_{\hat{b}}(s)$ is biased, some useful properties of the MLE estimator are [Antos and Kontoyiannis, 2001, Paninski, 2003]:

Theorem 13.

$$(a)\ \Pr(|H_{\hat{b}}(s) - \mathbb{E}[H_{\hat{b}}(s) \mid b]| \geq \eta) \leq \delta_\eta, \tag{5.62}$$

where $\delta_\eta = 2e^{\frac{-M}{2}\eta^2(\log(M))^{-2}}$.

$$(b)\ \mu_M(b) \leq \mathbb{E}[H_{\hat{b}}(s) \mid b] - H_b(s) \leq 0, \tag{5.63}$$

where $\mu_M(b) = -\log(1 + \frac{|supp(b)|-1}{M})$ *and* $|supp(b)|$ *is the size of support of* b.

Hence (5.62) bounds the variance of $H_{\hat{b}}(s)$ and (5.63) bounds its bias, which is always negative.

5.4.1 Lower Confidence Bound

Let $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ be defined as:

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) \triangleq \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{\hat{b}_{\mathbf{z}}^{\mathcal{A}}}(s), \tag{5.64}$$

where $H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s)$ is the MLE of the entropy of the posterior distribution $\hat{b}_{\mathbf{z}}^{\mathcal{A}}(s)$.

Lemma 8. *With probability* $1-\delta_l'$,

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) \leq H_{b}^{\mathcal{A}}(s|\mathbf{z}) + \eta, \tag{5.65}$$

where $\delta_l' = |\Omega| 2e^{\frac{-M}{2}\eta^2(\log(M))^{-2}}$, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) = \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{\hat{b}_{\mathbf{z}}^{\mathcal{A}}}(s)$ *and* $H_{b}^{\mathcal{A}}(s|\mathbf{z}) = \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{b_{\mathbf{z}}^{\mathcal{A}}}(s)$.

Proof. Starting from (5.63) for a fixed $\mathcal{A}$, for a particular $\mathbf{z}_i \in \Omega$ and $b = b_{\mathbf{z}_i}^{\mathcal{A}}$,

$$\mathbb{E}[H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}} | \hat{b}_{\mathbf{z}_i}^{\mathcal{A}}](s) \leq H_{b_{\mathbf{z}_i}^{\mathcal{A}}}(s). \tag{5.66}$$

Using (5.62), we know for a particular $\mathbf{z}_i$, with probability $1-\delta_\eta$,

$$H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}}(s) \leq \mathbb{E}[H_{\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}} | \hat{b}_{\mathbf{z}_i}^{\mathcal{A}}](s) + \eta. \tag{5.67}$$

$$\leq H_{b_{\mathbf{z}_i}^{\mathcal{A}}}(s) + \eta. \tag{5.68}$$

Eq. (5.68) follows directly from (5.66). Now, since (5.68) is true for all $\mathbf{z}$, then taking a simple expectation over $\mathbf{z}$ gives,

$$\sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{\hat{b}_{\mathbf{z}}^{\mathcal{A}}}(s) \leq \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathcal{A}) H_{b_{\mathbf{z}}^{\mathcal{A}}}(s) + \eta. \tag{5.69}$$

$$H_{\hat{b}}(s|\mathbf{z}) \leq H_b(s|\mathbf{z}) + \eta. \tag{5.70}$$

The probability of the event that the (5.68) is true for all $\mathbf{z} \in \Omega$, is $1-$ probability that at least for one particular value of $\mathbf{z} \in \Omega$ (5.68) is not true. Using union bound, this probability is bounded by sum over $\mathbf{z}$ over probability that for a particular value of $\mathbf{z}$ (5.68) is not true. Since there can be only $|\Omega|$ different values of $\mathbf{z}$, the probability that at least for one particular value of $\mathbf{z} \in \Omega$ (5.68) is not true is bounded by $|\Omega|\delta_\eta$.

Thus, with probability $1 - \delta'_l$,

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) \leq H_b^{\mathcal{A}}(s|\mathbf{z}) + \eta, \tag{5.71}$$

where $\delta'_l = |\Omega|\delta_\eta$. □

Using Lemma 8, L_t on $H_{b_{\mathbf{z}}^{\mathcal{A}}}(s)$ after t calls to `tighten`$(\mathcal{A}, t)$ can be constructed as: with probability $1 - \frac{\delta_l}{nt(t+1)}$, $L_t(\mathcal{A}) = H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) - \sqrt{\frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_l})} \leq H_b^{\mathcal{A}}(s|\mathbf{z})$. This expression can be obtained by equating the desired confidence $1 - \frac{\delta_l}{nt(t+1)}$ to $1 - |\Omega|\delta_\eta$ and solving for η:

$$|\Omega|\delta_\eta = \frac{\delta_l}{nt(t+1)} \tag{5.72}$$

$$\text{Substituting definition of } \delta_\eta = 2e^{-\frac{M}{2}\eta^2(\log(M))^{-2}}, \tag{5.73}$$

$$|\Omega|2e^{-\frac{M}{2}\eta^2(\log(M))^{-2}} = \frac{\delta_l}{nt(t+1)} \tag{5.74}$$

$$\text{Dividing both sides by } 2|\Omega| \tag{5.75}$$

$$e^{-\frac{M}{2}\eta^2(\log(M))^{-2}} = \frac{\delta_l}{2n|\Omega|t(t+1)} \tag{5.76}$$

$$\text{Taking log on both sides,} \tag{5.77}$$

$$-\frac{M}{2}\eta^2(\log(M))^{-2} = \log(\frac{\delta_l}{2n|\Omega|t(t+1)}) \tag{5.78}$$

$$\text{Multiplying } \frac{-2(\log(M))^2}{M} \text{ on both sides} \tag{5.79}$$

$$\eta^2 = \frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_l}) \tag{5.80}$$

$$\eta = \sqrt{\frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_l})} \tag{5.81}$$

Typically, the bottleneck in computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ is performing the belief update to find $\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}$ for each $\mathbf{z}_i$. In practice, we approximate these using *particle belief updates* [Doucet et al., 2001], which, for a given $\mathbf{z}_i$, generate a sample s^j from $\hat{b}(s)$ and then an observation $\mathbf{z}'$ from $\Pr(\mathbf{z}|s^j, \mathcal{A})$. If $\mathbf{z}_i = \mathbf{z}'$, then s^j is added to the set of samples approximating $\hat{b}_{\mathbf{z}_i}^{\mathcal{A}}$. Consequently, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ can be tightened by increasing M, the number of samples used to estimate $\hat{b}_{\mathbf{z}_i^{\mathcal{A}}}$. However, tightening $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$ by using larger values of M is not practical as computing it involves new posterior belief updates (with a larger value of M) and hence increases the computational cost of tightening $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z})$.

5.4.2 Upper Confidence Bound

Since $H_{\hat{b}}(s)$ is negatively biased, finding an upper confidence bound is more difficult. A key insight is that such a bound can nonetheless be obtained by estimating posterior entropy using an artificially "coarsened" observation function. That is, we group all possible observations into a set Φ of clusters and then pretend that, instead of observing $\mathbf{z}$, the agent only observes what cluster $\mathbf{z}$ is in. Since the observation now contains less information, the conditional entropy will be higher, yielding an upper bound. Furthermore, since the agent only has to reason about $|\Phi|$ clusters instead of $|\Omega|$ observations, it is also cheaper to compute. Any generic clustering approach, e.g., ignoring certain observation features can be used, though in some cases domain expertise may be exploited to select the clustering that yields the tightest bounds.

Let $\mathbf{r} = \langle r_1 \dots r_n \rangle$ represent a crude approximation of $\mathbf{z}$. That is, for every i, r_i is obtained from z_i by $r_i = f(z_i, d)$, where f clusters z_i into d clusters deterministically and r_i denotes the cluster to which z_i belongs. Also, if $z_i = \emptyset$, then $r_i = \emptyset$. Note that $H_b(\mathbf{r}|\mathbf{z}) = 0$ and the domain of r_i and r_j share only $\emptyset$ for all i and j.

Lemma 9. *Let* $\mathbf{r}$ *be an approximation of* $\mathbf{z}$*, that is,* $\mathbf{r}$ *is obtained after clustering (or processing)* $\mathbf{z}$ *deterministically such that it contains not extra information about* s*, then*

$$H_b^{\mathcal{A}}(s|\mathbf{z}) \leq H_b^{\mathcal{A}}(s|\mathbf{r}). \tag{5.82}$$

Proof. To prove this Lemma we use the chain rule for entropy that states that for two random variables A and B, the joint entropy can be expressed as:

$$H(A, B) = H(A|B) + H(B) = H(B|A) + H(A). \tag{5.83}$$

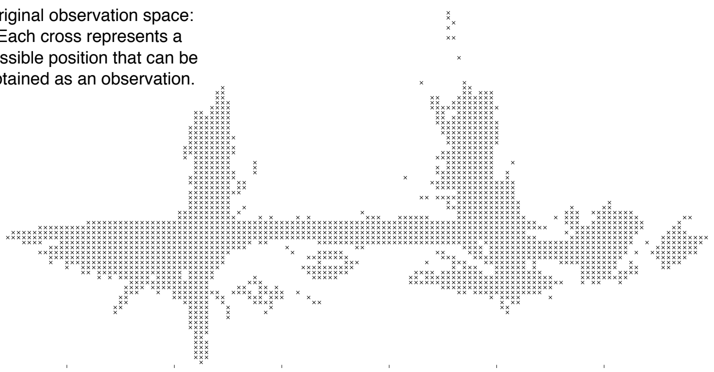
Using the chain rule for entropy on $H_b^{\mathcal{A}}(s, \mathbf{z}|\mathbf{r})$

$$H_b^{\mathcal{A}}(s, \mathbf{z}|\mathbf{r}) = H_b^{\mathcal{A}}(s|\mathbf{z}, \mathbf{r}) + H_b^{\mathcal{A}}(\mathbf{z}|\mathbf{r}) = H_b^{\mathcal{A}}(\mathbf{z}|s, \mathbf{r}) + H_b^{\mathcal{A}}(s|\mathbf{r}). \tag{5.84}$$

Since $\mathbf{r}$ contains no additional information and can be determined with full certainty given $\mathbf{z}$, $H_b^{\mathcal{A}}(s|\mathbf{z}, \mathbf{r}) = H_b^{\mathcal{A}}(s|\mathbf{z})$,

$$H_b^{\mathcal{A}}(s|\mathbf{z}) + H_b^{\mathcal{A}}(\mathbf{z}|\mathbf{r}) = H_b^{\mathcal{A}}(\mathbf{z}|s, \mathbf{r}) + H_b^{\mathcal{A}}(s|\mathbf{r}). \tag{5.85}$$

Since conditioning can never increase entropy [Cover and Thomas, 1991], $H_b^{\mathcal{A}}(\mathbf{z}|s, \mathbf{r}) \leq$



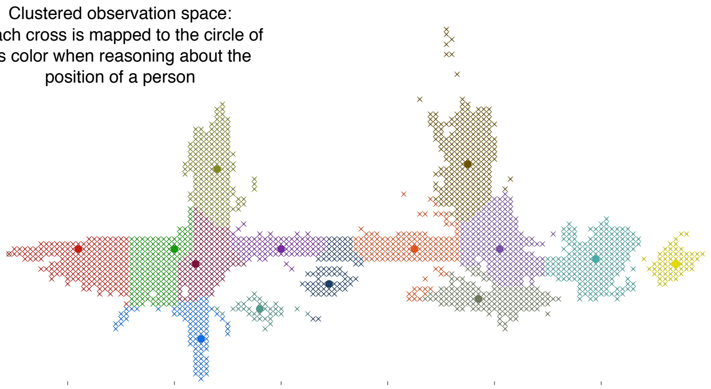


Figure 5.2: An example clustering of the original finely discertized observation space into 14 clusters with each cluster shown in a different color. The big color filled circles indicate the approximate center point of that particular cluster. By reasoning about the position of the person in the scene with this coarse observation space, the agent can compute a lower bound on the information gained from observing a particular region in the space. Best viewed in colour.

$H_b^{\mathcal{A}}(\mathbf{z}|\mathbf{r})$. Thus, for the equality in (5.85) to hold, it is necessary that

$$H_b^{\mathcal{A}}(s|\mathbf{r}) \geq H_b^{\mathcal{A}}(s|\mathbf{z}) \tag{5.86}$$

$\square$

$H_b^{\mathcal{A}}(s|\mathbf{r})$ is cheaper to compute than $H_b^{\mathcal{A}}(s|\mathbf{z})$ because it requires only $|\Phi|$ belief updates instead of $|\Omega|$. Starting with a small Φ, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r})$ can be tightened by increasing the number of clusters and thus $|\Phi|$. Figure 5.2 shows an example of clustering finely discretized observation space (top) into 14 clusters (bottom). Figure 5.3 shows the bias in entropy estimation as the number of samples are increased. It also shows that if the state space is clustered and a new probability distribution is computed over the clustered state space is leads to a reduction in the true entropy and bias of the entropy estimator.

Note that computing $H_b^{\mathcal{A}}(s|\mathbf{r})$ requires $\Pr(\mathbf{r}|s,\mathcal{A})$, which can be obtained by marginalizing $\mathbf{z}$ out from $\Pr(\mathbf{z}|s,\mathcal{A})$, a computationally expensive operation. However, this marginalization only needs to be done once and can be reused when performing greedy maximization for various $b(s)$. This occurs naturally in, e.g., sensor selection, where the hidden state that the agent wants to track evolves over time. At every time step, $b(s)$ changes and a new set $\mathcal{A}^P$ must be selected.

However, computing $H_{b_{\mathbf{r}}^{\mathcal{A}}}(s)$ still requires iterating across all values of s. Thus, to lower the computational cost further, we use estimates of entropy, as with the lower bound:

$$H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) = \sum_{\mathbf{r}\in\Phi} \Pr(\mathbf{r}|b,\mathcal{A}) H_{\hat{b}_{\mathbf{r}}^{\mathcal{A}}}(s). \tag{5.87}$$

Computing $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r})$ is cheaper than $H_b^{\mathcal{A}}(s|\mathbf{r})$ but is not guaranteed to be greater than $H_b^{\mathcal{A}}(s|\mathbf{z})$ since the entropy estimates have negative bias. However, we can still obtain an upper confidence bound.

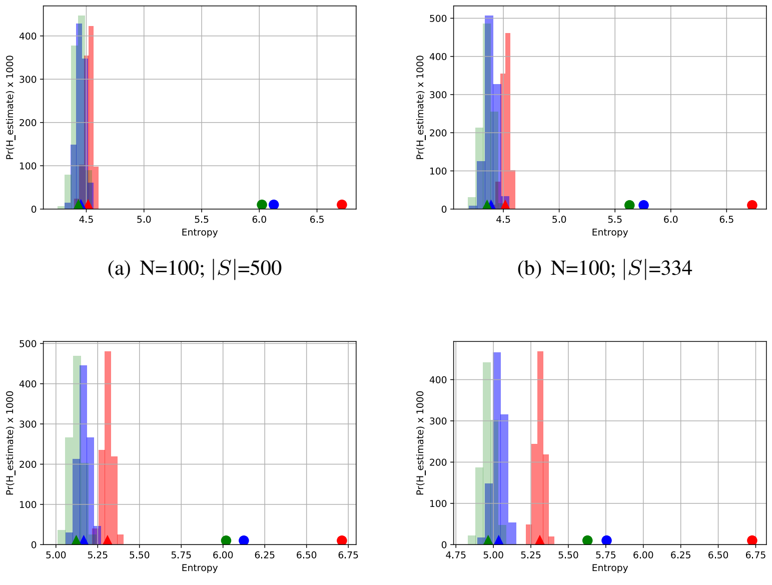
Lemma 10. *With probability* $1-\delta'_u$

$$H_b^{\mathcal{A}}(s|\mathbf{z}) \leq H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) + \eta - \mu_M(b), \tag{5.88}$$

where $\delta'_u = |\Phi|2e^{\frac{-M}{2}\eta^2(\log(M))^{-2}}$, $H_b^{\mathcal{A}}(s|\mathbf{z}) = \sum_{\mathbf{z}\in\Omega}\Pr(\mathbf{z}|b,\mathcal{A})H_{b_{\mathbf{z}}^{\mathcal{A}}}(s)$, $H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) = \sum_{\mathbf{r}\in\Phi}\Pr(\mathbf{r}|b,\mathcal{A})H_{\hat{b}_{\mathbf{r}}^{\mathcal{A}}}(s)$ *and* $\mu_M(b) = -\log(1+\frac{|supp(b)|-1}{M})$ *is assumed to be constant for some large value of* $|supp(b)| = C$.

Proof. (5.62) implies that for a given $\mathcal{A}$ and for a particular $\mathbf{r}_i$, with probability $1-\delta_\eta$,

$$\mathbb{E}[H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s) \mid b_{\mathbf{r}_i}^{\mathcal{A}}] \leq H_{\hat{b}_{\mathbf{r}_i}^{\mathcal{A}}}(s) + \eta. \tag{5.89}$$



(a) N=100; $|S|$=500

(b) N=100; $|S|$=334

(c) N=250; $|S|$=500

(d) N=250; $|S|$=334

Figure 5.3: Figure illustrating bias in entropy estimation in a simulated setting. In this experiment, a probability distribution over a hidden variable s that can take values in the set $S = \{1, 2, \ldots, 1000\}$ was considered. The red dot in the plots above shows the true entropy of the distribution and the red triangle and red histogram show the mean and the distribution of estimating entropy with N samples. To contrast, the values in the set S were clustered into 500 and 334 clusters such that in the resulting distribution the probability mass on a resulting cluster is the sum of the probability mass (in the original distribution) on the values that are in the cluster. The blue dot represents the entropy of the resulting distribution over the clustered values and blue triangle and histogram show the mean and distribution of estimated entropy respectively when the clusters are generated randomly. The other clustering strategy that was tried was to cluster values with high probability mass together. This is represented by the green dot, triangle and histogram. The figure shows that as the ratio of the number of samples used to estimate the entropy to the size of the support of the true distribution increases, the bias (relative to the the true entropy) in the estimation of entropy decreases. While here we cluster the state space, clustering the observation space (effectively smaller observation space) when computing information gain leads to a similar effect as a smaller number of observations allows for a higher number of samples to estimate posterior belief entropies leading to a lesser bias in the entropy estimation.

Since (5.89) is true for all possible values of $\mathbf{r}_i$ with probability $1-\delta_\eta$, taking an expectation over $\mathbf{r}_i$ and using union bound (explained below) gives, with probability $1-|\Phi|\delta_\eta$,

$$\mathbb{E}_{\mathbf{r}}[\mathbb{E}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}}}(s) \mid b^{\mathcal{A}}_{\mathbf{r}}]] \leq \mathbb{E}_{\mathbf{r}}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}}}(s) + \eta]. \tag{5.90}$$

The logic behind using union bound here is that the probability that (5.89) does not hold for at least one particular value of $\mathbf{r}$ is bounded by sum over all values of $\mathbf{r}$ of probabilities that (5.89) does not hold for that particular value of $\mathbf{r}_i$, which is $|\Phi|\delta_\eta$. Here $|\Phi|$ is the size of set containing all possible values of $\mathbf{r}$. Thus, the probability of the event that (5.89) holds for all values of $\mathbf{r} \in \Phi$ is $1-|\Phi|\delta_\eta$. Thus, the following sets of inequalities hold (explanation below the equations), with probability $1-\delta'_u$,

$$\mathbb{E}_{\mathbf{r}|b,\mathcal{A}}[\mathbb{E}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}}}(s) \mid b^{\mathcal{A}}_{\mathbf{r}}]] \leq \mathbb{E}_{\mathbf{r}|b,\mathcal{A}}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}}}(s) + \eta] \tag{5.91}$$

$$\mathbb{E}_{\mathbf{r}|b,\mathcal{A}}[H_{b^{\mathcal{A}}_{\mathbf{r}}}(s) + \mu_M(b^{\mathcal{A}}_{\mathbf{r}})] \leq \mathbb{E}_{\mathbf{r}|b,\mathcal{A}}[H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}}}(s) + \eta] \tag{5.92}$$

$$\mathbb{E}_{\mathbf{r}|b,\mathcal{A}}H_{b^{\mathcal{A}}_{\mathbf{r}}}(s) + \mu_M(b) \leq \mathbb{E}_{\mathbf{r}|b,\mathcal{A}}H_{\hat{b}^{\mathcal{A}}_{\mathbf{r}}}(s) + \eta \tag{5.93}$$

$$H^{\mathcal{A}}_b(s|\mathbf{r}) + \mu_M(b) \leq H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \eta \tag{5.94}$$

$$H^{\mathcal{A}}_b(s|\mathbf{z}) + \mu_M(b) \leq H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \eta \tag{5.95}$$

$$H^{\mathcal{A}}_b(s|\mathbf{z}) \leq H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \eta - \mu_M(b). \tag{5.96}$$

Eq (5.92) follows from (5.63) that bounds the bias in the entropy estimator by $\mu_M(b)$, Eq. (5.93) is simple separation of expectations applied to separate terms, we have assumed η and $\mu_M(b)$ to be constant (in the statement of the Lemma), Eq. (5.94) is replacing the conditional entropy $H_b(s|\mathbf{r})$ and $H_{\hat{b}}(s|\mathbf{r})$ by their definitions given in the statement of the Lemma, Eq. (5.95) is obtained after using Lemma 9 to obtain $H_b(s|\mathbf{z}) \leq H_b(s|\mathbf{r})$, Eq. (5.96) is obtained by subtracting $\mu_M(b)$ from both sides. □

With Lemma 10, U_t can be constructed as: $U_t(\mathcal{A}) = [H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r}) + \sqrt{\frac{2\log(M)^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_u})} + \log(1+\frac{1}{M}(|supp(b)|-1))] \geq H_b(s|\mathbf{z})$ (using $|\Phi| \leq |\Omega|$). In practice, we use a larger value of M when computing $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r})$ than $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$. Doing so is critical for reducing the negative bias in $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$. Furthermore, doing so does not lead to intractability because choosing a small $|\Phi|$ ensures that few belief updates will be performed.

Thus, when computing $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{z})$, we set M low but perform many belief updates; when computing $H^{\mathcal{A}}_{\hat{b}}(s|\mathbf{r})$ we set M high but perform few belief updates. This yields cheap upper and lower confidence bound for conditional entropy.

The following theorem ties together all the results we presented. Note that, since Q

is defined as *negative* conditional entropy, L_t is defined using our *upper* bound and U_t using our *lower* bound.

Theorem 14. *Let* $Q(\mathcal{A}) = H_b(s) - H_b^{\mathcal{A}}(s|\mathbf{z})$. *Let* `tighten` *be defined such that* $\texttt{tighten}(\mathcal{A}, \texttt{t})$ *returns*
$U_t(\mathcal{A}) = H_b(s) - H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{z}) + \sqrt{\frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_u})}$ *and*
$L_t(\mathcal{A}) = H_b(s) - [H_{\hat{b}}^{\mathcal{A}}(s|\mathbf{r}) + \sqrt{\frac{2(\log(M))^2}{M}\log(\frac{2n|\Omega|t(t+1)}{\delta_l})} + \log(1 + \frac{1}{M}(|supp(b)| - 1))]$. *Let* $\mathcal{A}^P = \texttt{pac-greedy-max}(\texttt{tighten}, \mathcal{X}, k, \epsilon_1)$ *and* $\mathcal{A}^* = \arg\max_{\mathcal{A}\in\mathcal{A}^+} Q(\mathcal{A})$, *where* $\mathcal{X} = \{1, 2, \ldots, n\}$ *and* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$. *If* $\mathbf{z}$ *is conditionally independent given* s *then, with probability* $1 - \delta$,

$$Q(\mathcal{A}^P) \geq (1 - e^{-1})Q(\mathcal{A}^*) - \epsilon, \tag{5.97}$$

where $\delta = k(\delta_l + \delta_u)$, $\epsilon = k\epsilon_1$.

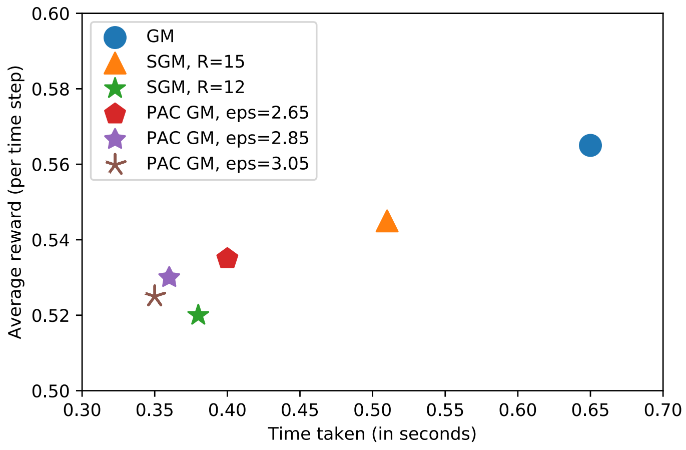
Proof. We showed that with probability $1 - \frac{\delta_l}{nt(t+1)}$, $L_t(\mathcal{A}) \leq Q(\mathcal{A})$ and with probability $1 - \frac{\delta_u}{tn(t+1)}$, $U_t(\mathcal{A}) \geq Q(\mathcal{A})$. Krause and Guestrin [2005b] showed that Q is non-negative, monotone and submodular if $\mathbf{z}$ is conditionally independent given s. The `tighten` procedure can be designed by tightening the upper and lower bounds by either increasing M or by changing the clusters used to estimate $H_{\hat{b}}(s|\mathbf{r})$. Thus, Theorem 12 with $\epsilon = k\epsilon_1$ and $\delta_1 = \delta_u + \delta_l$ implies the stated result. ☐

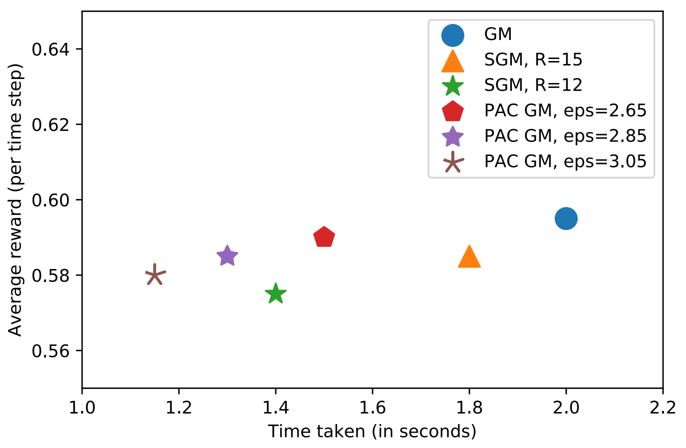
5.5 Experiments

We evaluated PAC greedy maximization on the problem of tracking people on the shopping mall dataset. The field of view of a few cameras were divided into two or three separate regions and each region was treated as an independent camera, so as to enable more challenging experiments with as many as $n = 20$ cameras.

We first consider tracking a single person. The hidden state s is modelled as the position and velocity of the person and described by the tuple $\langle x, y, v_x, v_y\rangle$, where x and y describe the position and v_x and v_y describe his velocity in the x and y directions. Both x and y are integers in $\{0, \ldots, 150\}$. The surveillance area can be observed with $n = 20$ cameras and, if selected, each camera produces an observation $\langle z^x, z^y\rangle$ containing an estimate of the person's x-y position.

We assume a person's motion in the x direction is independent of his motion in the y direction. Given the current position x_{curr}, the future position x_{next} is a deterministic function of x_{curr} and the current velocity in x-direction v_x^{curr}, i.e., $x_{next} =$





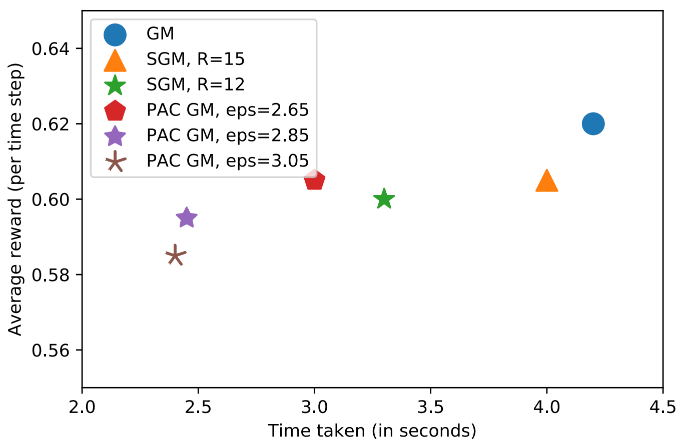


Figure 5.4: Multi-person tracking for $n = 20$ and (top) $k = 1$; (middle) $k = 2$; (bottom) $k = 3$.

$x_{curr} + v_x^{curr}$. The same is true for the y position. The future velocity v_{next} is modelled as a Gaussian distribution with the current velocity as the mean and the standard deviation, which depends on the current x-y position, learnt from the data, i.e., $v_x^{next} \sim \mathcal{N}(v_x^{curr}, \sigma^x)$ and $v_y^{next} \sim \mathcal{N}(v_y^{curr}, \sigma^y)$. The observations are assumed to be conditionally independent given the state and are generated from a Gaussian distribution with the true position as the mean and a randomly generated standard deviation. Since ground truth data about people's locations is not available, learning the standard deviation is not possible. A belief $b(s)$ about the person's location was maintained using an unweighed particle filter with 200 particles. Given a subset of the sensors and the observations they generated, $b(s)$ is updated using a Monte-Carlo belief update [Doucet et al., 2001]

To evaluate a given algorithm, a trajectory was sampled randomly. At each time step in the trajectory, a subset of k cameras out of $n = 20$ were selected by the algorithm. Using the resulting observations, the person was tracked using an unweighed particle filter [Doucet et al., 2001], starting from a random initial belief. At each time step, a prediction $\arg\max_s b(s)$ about the person's location was compared to the person's true location. Performance is the average number of correct predictions made per trajectory. For multi-person tracking, the best subsets of cameras for each person were computed independently of each other and then the subset with the highest value of Q was selected.

We conducted experiments with different values of n and k. As a baseline, we use greedy maximization and stochastic greedy maximization. Since we cannot compute Q exactly, greedy maximization simply uses an approximation, based on MLE estimates of conditional entropy, ignoring the resulting bias and making no attempt to reason about confidence bounds. Stochastic greedy maximization, in each iteration, samples a subset of size R from $\mathcal{X}$ and selects from that subset the element that maximizes the estimated marginal gain. Neither greedy nor stochastic greedy maximization employ lazy evaluations, i.e., pruning elements via a priority queue as in lazy greedy maximization, because the reliance on approximation of Q means pruning is no longer justified. In addition, since lazy greedy maximization's pruning is based on marginal gain instead of Q, the bias is exacerbated by the presence of two entropy approximations instead of one. On average the length of each trajectory sampled was 25 time steps and the experiments were performed on 30 trajectories with 5 independent runs. To avoid clutter, we show results for only the two best performing parameter (parameters tuned for all three algorithms were: number of samples used to estimate the posterior entropy, number of observations sampled to reason about future beliefs, clustering of the observations space for PAC GM, value of ϵ for PAC GM, size of $\mathcal{R}$ for SGM, etc) settings of each algorithm.

Figure 5.4 shows the number of correct predictions (y-axis) against the runtime (x-

axis) of each method. Thus, the top left is the most desirable region. In general, PAC greedy maximization performs nearly as well as the best-performing algorithm but does so at lower computational cost. As k increases PAC greedy maximization scales better than greedy maximization and stochastic greedy maximization and hence is more suited for real-life problems.

5.6 Conclusions & Future Work

In this chapter, we proposed PAC greedy maximization, a new algorithm for submodular function maximization when evaluating the function exactly is not feasible. PAC greedy maximization in each iteration calls `pac-max` that uses cheap and tunable upper and lower confidence bounds on information gain to find the best sensor to add to a partial solution. `pac-max` is closely related to the algorithms for finding the best arm in a multi-armed bandit [Audibert and Bubeck, 2010]. Thus, an exciting line of future work is to apply the ideas from the existing literature on best arm identification to PAC greedy maximization.

To complement PAC greedy maximization we propose cheap confidence bounds on information gain. Estimation of entropy is a popular topic in the field of statistics and machine learning. However, the bounds we propose are computationally cheaper than the existing methods which makes them suitable for the sensor selection task. By exploiting submodularity we showed that PAC greedy maximization when applied with our proposed confidence bounds is guaranteed to return a solution that with high probability has bounded error. A key factor for the performance of PAC greedy maximization can be the way by which the observations are clustered to get the upper confidence bound. Thus, studying good strategies for clustering for getting upper confidence bounds on entropy can be an interesting avenue for future work.

While in this chapter we focussed on reducing entropy of the belief of the agent at the next immediate time step, employing PAC greedy maximization for long-term planning is fairly simple, as greedy maximization in greedy PBVI can be simply replaced by PAC greedy maximization. It can even be combined with Monte Carlo value iteration [Bai et al., 2010], to yield a scalable offline planner. Moreover, if offline planning is not tractable, PAC greedy maximization can also be easily integrated with bandit-based planning [Kocsis and Szepesvári, 2006, Silver and Veness, 2010] for online planning.

A key bottleneck in the computation of information gain is that the computational cost of information gain grows exponentially with the size of the subset of sensors. This is because the number of observations that can be generated from a subset of sensors grows exponentially with its size. Thus, a key challenge to make sensor selection truly scalable is to either circumvent or overcome the challenge of expensive evaluation functions with algorithms that quickly identify the suboptimal choices.

Finally, the real-life applications of sensor selection algorithms can involve not just tens or hundreds of sensors, but thousands of sensors. While PAC greedy maximization is able to select 3 sensors out of 20 relatively quickly as compared to other methods, scaling PAC greedy maximization to even larger scenes can prove to be tough. In the next chapter we present a fundamentally different approach to scale greedy maximization to scenes where thousands of sensors are involved. By proposing a new utility function that is an approximation of a traditional utility function we are able to propose a new algorithm that can identify the most informative sensors out of approximately 10000 sensors in milliseconds.

6
Real-Time Online Planning

The previous chapters focused on sensor selection as an active perception task. While offline planning and PAC planning scale to problems of reasonably large size, for many active perception problems it is simply not possible to scale these methods because of the sheer size of the problem. For example, when tracking people in ultra high resolution images it is required to apply a trained person detector on the many possible pixel boxes (a rectangle that potentially contains a person or an object of interest, shown in Figure 6.1) in the image to detect people. Many tracking systems struggle to perform in real-time because of the high computational cost of detecting people in the ultra high resolution images. Thus, in this chapter we focus on real-time resource allocation for tracking systems. We formulate this problem as selecting k out of the n pixel boxes to apply a person detector on to track people in high resolution images. We introduce a new algorithm PartiMax that is quickly able to identify the most relevant pixel boxes in an image to apply a person detector on to track people, without adding any significant computational overhead. PartiMax exploits information in the previous beliefs to select k of the n candidate pixel boxes in the image. We prove that PartiMax is guaranteed to make a near-optimal selection with error bounds that are independent of the problem size. Furthermore, empirical results on a real-life dataset show that our system runs in real-time by processing only 10% of the pixel boxes in the image while still retaining 80% of the original tracking performance achieved when processing all the pixel boxes.

In the rest of this chapter we describe the motivation and background for tracking people in high resolution images, followed by description of our algorithm PartiMax and its analysis. Finally, we present our experiments that show that PartiMax runs in real-time by processing only 40 out of 7200 pixel boxes.

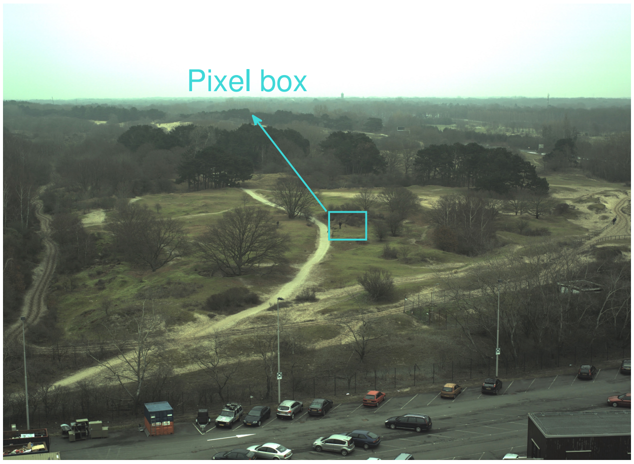


Figure 6.1: A wide-view scene recorded by a rooftop camera; the cyan rectangle shows an example pixel box.

6.1 Automated Tracking

Automated tracking is a key component of countless computer vision applications such as maintaining surveillance, studying traffic flows, and counting the number of people in a scene [Hu et al., 2012, Smeulders et al., 2014]. Consequently, in recent years many tracking systems have been proposed that make it possible to track people in a variety of challenging settings [La Cascia et al., 2000, Benfold and Reid, 2011, Smeulders et al., 2014]. However, these approaches still cannot perform real-time tracking on ultra high resolution videos (e.g., 5000×4000 pixels). In particular, the *detection* stage, i.e., identifying an object in a scene, is the main computational bottleneck for systems that work on the *tracking-by-detection* principle [Benfold and Reid, 2011]. For example, Figure 6.1 shows a wide-view scene recorded by a camera mounted on top of a building [Schutte et al., 2016]. Successful tracking depends on detecting the person in the image by applying a trained detector to many *pixel boxes*. Since the scene records a wide landscape, the pixel boxes must be relatively small (e.g., 180×180), yielding approximately 7200 pixel boxes per image. Consequently, performing a *brute force detection* (BD) that applies the person detector to all 7200 pixel boxes is extremely computationally intensive and prohibitive to do in real time.

In this chapter we propose a new algorithm that greatly reduces the cost of detection and thus enables real-time tracking on systems with ultra high resolution images or many cameras. The main idea is to perform *selective detection* (SD), i.e., apply the person

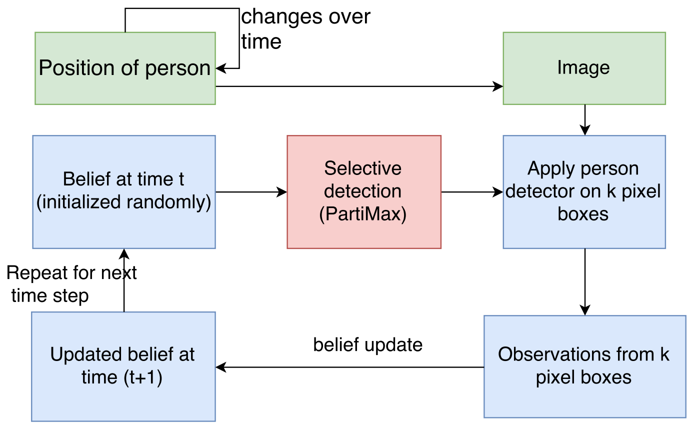

changes over time
Position of person
Image
Belief at time t (initialized randomly)
Selective detection (PartiMax)
Apply person detector on k pixel boxes
Repeat for next time step
Updated belief at time (t+1)
belief update
Observations from k pixel boxes

Figure 6.2: Proposed tracking system with PartiMax, our proposed selective detection method, highlighted in red. Blue boxes indicate other parts of the tracking system. At every time step, a belief about the position of the person in the image is maintained. PartiMax selects k pixel boxes in the image to which to apply a person detector. The observations only from the k selected pixel boxes are the used to update the belief about the position of the person in the image. The updated belief serves as the prior belief for the next time step and the system repeats the process.

detector not on all n pixel boxes, but only a carefully selected subset of k pixel boxes, while retaining performance guarantees, as shown in Figure 6.2. To do so, we build on existing techniques for *sensor selection*, which select the k out of n sensors with the highest *utility* in a multi-sensor network. By modelling each of the n pixel boxes as a separate sensor we are able to exploit existing methods for sensor selection to perform selective detection. The main challenge with planning online in real-time is that there are $\binom{n}{k}$ ways to perform the selection, and computing the best one would use up the same scarce computational resources we aim to intelligently allocate. Fortunately, when the utility function is submodular, greedy maximization which evaluates the utility function only $\mathcal{O}(nk)$, instead of $\binom{n}{k}$, times can find a near-optimal solution. In addition, variants of greedy maximization like stochastic greedy maximization [Mirzasoleiman et al., 2015] or PAC greedy maximization can further reduce the computational cost of planning.

However, for selective detection in real-time, even PAC/stochastic greedy maximization is too expensive because computing typical utility functions such as *information gain* or *expected coverage* requires marginalizing out the observation that each candidate sensor generates. In fact, in real-life settings evaluating information gain or expected coverage even once can be prohibitively expensive.

Thus, we propose a new utility function for selective detection called *particle cover-*

age and show it approximates expected coverage under certain conditions, but is much faster to compute. Then, we describe *PartiMax*. Unlike (stochastic) greedy maximization, which treats utility evaluation as a black-box, PartiMax maintains and updates the particle coverage of each pixel box in every iteration of greedy maximization, leading to large computational savings, as the particle coverage of each pixel box is not evaluated from scratch in each iteration. Furthermore, instead of eliminating a subset of pixel boxes *randomly* in every iteration like stochastic greedy maximization, PartiMax samples pixel boxes with high particle coverage, minimizing the chance of eliminating good pixel boxes and yielding superior tracking performance.

Since sampling pixel boxes with high particle coverage without computing the particle coverage is not trivial, we propose a sampling algorithm that can sample pixel boxes with high particle coverage in constant time. It does so by employing *tile coding*, a popular representation in reinforcement learning that discretizes continuous spaces. Furthermore, we prove that this algorithm is guaranteed to sample a pixel box with probability directly proportional to its particle coverage.

We show that, given access to a sampling algorithm like the one we propose, PartiMax is guaranteed to return a solution with tight error bounds that are independent of the problem size, i.e., independent of both n and k. Although PartiMax is designed for the particle coverage function, our bound applies generally to maximization over a set function.

Finally, we use PartiMax for selective detection to build a real-time tracking system, which we apply to a real-life dataset. Our results show that our tracking system retains 80% of its performance despite processing only 10% of each image and running in real time.

In the rest of the chapter, we give a quick recap of the notation (same as the previous chapters) followed by an introduction to particle filters for tracking. Next, we describe the particle coverage utility function for selective detection, followed by PartiMax, the algorithm we propose and its analysis and finally the experiments.

6.2 Background

Basic Setup

Let $\mathcal{X} = \{1, 2 \dots n\}$ denote the set of all pixel boxes and i denote a single pixel box in $\mathcal{X}$. $\mathcal{A}^+$ denotes the set of all possible subsets of $\mathcal{X}$ of size less than or equal to k,

$\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$. s is a hidden variable denoting the true location of a person[1] in the scene and S is the set of all possible values s can take. The observation vector $\mathbf{z} = \langle z_1, z_2 \ldots z_n \rangle$ denotes the result of applying the detector to each pixel box, i.e., an estimate of whether a person in s appears in that box. If a pixel box i is not selected for detection, then $z_i = \emptyset$. Ω is the set of all possible values $\mathbf{z}$ can take. The *belief* $b(s)$ is a probability distribution over s. Given $\mathcal{A}$ and $\mathbf{z}$, $b(s)$ can be updated using Bayes rule.

When there are many possible states, it is not possible to maintain $b(s)$ exactly. Thus, we use particle filters, described below, to maintain and update belief $b(s)$.

Particle Filters

When there are many possible states, it is infeasible to update $b(s)$ exactly. Instead, we can use *particle filters* [Doucet et al., 2001], sequential Monte Carlo algorithms for approximate inference in partially observable scenarios that are commonly used to track people in complex situations. The true belief $b(s)$ is approximated with a *particle belief* $\mathcal{B}$, a collection of m samples from $b(s)$, called particles: $\mathcal{B} = \{s_1, s_2 \ldots s_m\}$. Although weighted particle filters are often used for tracking, we use an unweighed particle filter since it can be efficiently implemented with a black-box simulator without the need to explicitly model the accuracy of the person detector or the motion dynamics [Silver and Veness, 2010].

Given a subset $\mathcal{A}$ and observation $\mathbf{z}$, particle beliefs can be updated using a *Monte Carlo belief update*. For each particle $s_l \in \mathcal{B}$, the next state s'_l is sampled from $\Pr(s'|s)$ (under the Markov assumption) to form $\mathcal{B}' = \{s'_l : s'_l \sim \Pr(s'|s_l) \wedge s_l \in \mathcal{B}\}$. For each $s'_l \in \mathcal{B}'$, the corresponding $\mathbf{z}_l$ is drawn from $\Pr(\mathbf{z}|s'_l, \mathcal{A})$. If $\mathbf{z}_l$ matches $\mathbf{z}$ according to some metric, or if $\mathbf{z}_l = \mathbf{z}$, then s'_l is added to the updated belief $\mathcal{B}^{\mathcal{A}}_{\mathbf{z}}$. Otherwise, the particle is discarded. To avoid particle degeneracy, a common problem with particle filters, we combine the belief update with new particles introduced by adding random artificial particles to the existing particle set.

Utility Functions

For tracking tasks, the utility function is often defined as information gain as described in Chapters 3 and 4 [Cover and Thomas, 1991, Krause and Guestrin, 2005b, Tham and Han, 2013]. It can also be defined as *expected coverage* [Spaan and Lima, 2009]. For particle belief expected coverage can be formally defined as: Let $\mathcal{I}^j_{\mathcal{B}'}$ be the set of particles in $\mathcal{B}'$

[1]For simplicity, we sometimes assume there is only one person in the scene. However, our methods and theoretical results extend easily to multiple people by assuming that any two person's motion and detections are independent. Furthermore, in the Experiments section, we present empirical results on tracking with multiple people.

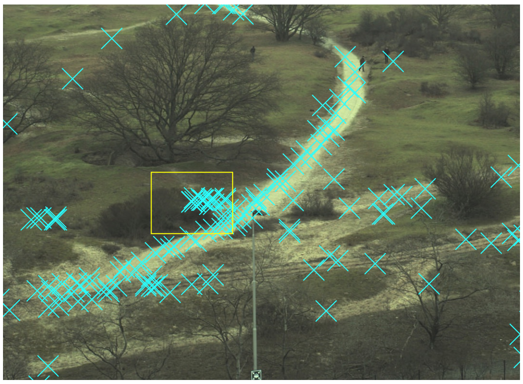


Figure 6.3: Particle belief: the yellow rectangle shows a pixel box and the particles it covers.

that are *covered* by pixel box j, $\mathcal{I}^j_{\mathcal{B}'} = \{s' \in \mathcal{B}' : j \text{ covers } s'\}$. A pixel box j covers s' if a person in state s' is visible in pixel box j. The expected coverage is defined as:

$$F_{\mathcal{B}'}(\mathcal{A}) = \sum_{\mathbf{z}} \Pr(\mathbf{z}|\mathcal{B}', \mathcal{A}) f_{\mathcal{B}^{\mathcal{A}}_{\mathbf{z}}}(\mathcal{A}), \tag{6.1}$$

where $f_{\mathcal{B}}(\mathcal{A}) = |\cup_{j \in \mathcal{A}} \mathcal{I}^j_{\mathcal{B}}|$ and $\Pr(\mathbf{z}|\mathcal{B}', \mathcal{A}) = \sum_{s' \in B'} \frac{1}{|\mathcal{B}'|} \Pr(\mathbf{z}|s', \mathcal{A}) f_{\mathcal{B}^{\mathcal{A}}_{\mathbf{z}}}(\mathcal{A})$.

Expected coverage is appropriate for sensor selection or selective detection because it rewards selecting pixel boxes that have the highest probability of detecting a target. The underlying assumption is that the observations generated by the person detector are informative enough to detect a person correctly inside a pixel box. This is barely a restrictive assumption, as most useful person detectors satisfy it.

6.3 Particle Coverage Utility Function

The utility functions described above are too expensive to compute in many practical settings, as they require marginalizing out observations, which is infeasible for real-time systems. In this section, we propose the *particle coverage function* (PCF) for selective detection, which does not require computing $\mathcal{B}^{\mathcal{A}}_{\mathbf{z}}$ and approximates expected coverage. PCF is defined as follows:

$$PCF_{\mathcal{B}'}(\mathcal{A}) = f_{\mathcal{B}'}(\mathcal{A}) = |\cup_{j \in \mathcal{A}} \mathcal{I}^j_{\mathcal{B}'}|. \tag{6.2}$$

$PCF_{\mathcal{B}'}(\mathcal{A})$ is simply the number of particles in $\mathcal{B}'$ that are covered by $\mathcal{A}$. In Figure 6.3, the particle coverage is the number of cyan particles that fall in the yellow pixel box. As opposed to expected coverage $F_{\mathcal{B}'}$, PCF does not involve an expectation over $\mathbf{z}$ nor does it require computing the resulting beliefs $\mathcal{B}_{\mathbf{z}}^{\mathcal{A}}$. PCF equals expected coverage under certain conditions, including the following.

Assumption 2. *For every $s' \in S$, $\mathcal{A} \subseteq \mathcal{X}$, there exist $\mathbf{z}_{s',\mathcal{A}}$ and $\bar{\mathbf{z}}_{s',\mathcal{A}}$ in Ω such that if s' is covered by $\mathcal{A}$, $\Pr(\mathbf{z} = \mathbf{z}_{s',\mathcal{A}}|s',\mathcal{A}) = 1$ and if s' is not covered by $\mathcal{A}$, then $\Pr(\mathbf{z} = \bar{\mathbf{z}}_{s',\mathcal{A}}|s',\mathcal{A}) = 1$.*

This is a simple assumption that says that for each combination of $s' \in S$ and $\mathcal{A} \subseteq \mathcal{X}$, there exists instances of observation $\mathbf{z}$ in the set Ω, where Ω is the set of all values $\mathbf{z}$ can take, $\bar{\mathbf{z}}_{s',\mathcal{A}}$ and $\mathbf{z}_{s',\mathcal{A}}$, such that they are received with full certainty depending on whether s' is covered by $\mathcal{A}$ or not. This assumption implies that any partial observability is due to perceptual aliasing, not noise in the sensors. Given Assumption 2, it is straightforward to show that expected coverage is equal to the particle coverage in the absence of sensor noise.

Theorem 15. *If Assumption 2 holds for a given $\mathcal{A}$, then $F_{\mathcal{B}'}(\mathcal{A}) = PCF_{\mathcal{B}'}(\mathcal{A})$, where $F_{\mathcal{B}'}(\mathcal{A}) = \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathcal{B}',\mathcal{A}) f_{\mathcal{B}_{\mathbf{z}}^{\mathcal{A}}}(\mathcal{A})$, $f_{\mathcal{B}'}(\mathcal{A}) = |\cup_{j\in\mathcal{A}} \mathcal{I}_{\mathcal{B}'}^{j}|$ and $PCF_{\mathcal{B}'}(\mathcal{A}) = f_{\mathcal{B}'}(\mathcal{A}) = |\cup_{j\in\mathcal{A}} \mathcal{I}_{\mathcal{B}'}^{j}|$. Here $\mathcal{I}_{\mathcal{B}'}^{j}$ denotes the set of particles in $\mathcal{B}'$ that are covered by the pixel box j .*

Proof. Expected coverage can be expressed as

$$F_{\mathcal{B}'}(\mathcal{A}) = \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|\mathcal{B}',\mathcal{A}) f_{\mathcal{B}_{\mathbf{z}}^{\mathcal{A}}}(\mathcal{A}) \tag{6.3}$$

$$= \sum_{\mathbf{z}\in\Omega} \sum_{s'\in\mathcal{B}'} \frac{1}{|\mathcal{B}'|} \Pr(\mathbf{z}|s',\mathcal{A}) f_{\mathcal{B}_{\mathbf{z}}^{\mathcal{A}}}(\mathcal{A}). \tag{6.4}$$

Let $\mathcal{S}'_{\mathcal{A}} \subseteq \mathcal{B}'$ denote the set of particles in $\mathcal{B}'$ that are covered by $\mathcal{A}$ and let $\bar{\mathcal{S}}'_{\mathcal{A}}$ be the set of particles in $\mathcal{B}'$ that are not covered by $\mathcal{A}$. Then, since for each combination of $s' \in \mathcal{B}', \mathcal{A}$, there are two observations possible: $\mathbf{z}_{s',\mathcal{A}}$ and $\bar{\mathbf{z}}_{s',\mathcal{A}}$. Thus, $F_{\mathcal{B}'}(\mathcal{A})$ can be written as:

$$F_{\mathcal{B}'}(\mathcal{A}) = \sum_{s'\in\mathcal{S}'_{\mathcal{A}}} \frac{1}{|\mathcal{B}'|} \Pr(\mathbf{z}_{s',\mathcal{A}}|s',\mathcal{A}) f_{\mathcal{B}_{\mathbf{z}_{s',\mathcal{A}}}^{\mathcal{A}}}(\mathcal{A}) + \sum_{s'\in\bar{\mathcal{S}}'_{\mathcal{A}}} \frac{1}{|\mathcal{B}'|} \Pr(\bar{\mathbf{z}}_{s',\mathcal{A}}|s',\mathcal{A}) \cancelto{0}{f_{\mathcal{B}_{\bar{\mathbf{z}}_{s',\mathcal{A}}}^{\mathcal{A}}}(\mathcal{A})} \tag{6.5}$$

$f_{\mathcal{B}_{\bar{\mathbf{z}}_{s',\mathcal{A}}}^{\mathcal{A}}}(\mathcal{A})$ is 0 for each $s' \in \bar{\mathcal{S}}'_{\mathcal{A}}$ because according to the particle belief update we described $\mathcal{B}_{\bar{\mathbf{z}}_{s',\mathcal{A}}}^{\mathcal{A}}$ will only accept particles from $\mathcal{B}'$ that are not covered by $\mathcal{A}$ since only those particle can lead to the observation $\bar{\mathbf{z}}_{s',\mathcal{A}}$. $f_{\mathcal{B}^{\mathcal{A}}_{\bar{\mathbf{z}}_{s',\mathcal{A}}}}(\mathcal{A})$ is the number of particles in $\mathcal{B}^{\mathcal{A}}_{\bar{\mathbf{z}}_{s',\mathcal{A}}}$ that are covered by $\mathcal{A}$ which is zero.

Thus,

$$F_{\mathcal{B}'}(\mathcal{A}) = \sum_{s' \in \mathcal{S}'_{\mathcal{A}}} \frac{1}{|\mathcal{B}'|} \Pr(\mathbf{z}_{s',\mathcal{A}}|s',\mathcal{A}) \cancel{f_{\mathcal{B}^{\mathcal{A}}_{\mathbf{z}_{s',\mathcal{A}}}}(\mathcal{A})}^{|\mathcal{B}'|} \tag{6.6}$$

$$\tag{6.7}$$

$f_{\mathcal{B}^{\mathcal{A}}_{\mathbf{z}_{s',\mathcal{A}}}}(\mathcal{A})$ is the number of particles covered by $\mathcal{A}$ in $\mathcal{B}^{\mathcal{A}}_{\mathbf{z}_{s',\mathcal{A}}}$ if $\mathbf{z}_{s',\mathcal{A}}$ is observed. $\mathcal{B}^{\mathcal{A}}_{\mathbf{z}_{s',\mathcal{A}}}$ will only accept particles from $\mathcal{B}'$ that are covered by $\mathcal{A}$ because only those particles can lead to the observation $\mathbf{z}_{s',\mathcal{A}}$.

$$F_{\mathcal{B}'}(\mathcal{A}) = \sum_{s' \in \mathcal{S}'_{\mathcal{A}}} \frac{1}{|\mathcal{B}'|} \Pr(\mathbf{z}_{s',\mathcal{A}}|s',\mathcal{A})|\mathcal{B}'| \tag{6.8}$$

The sum $\sum_{s' \in \mathcal{S}'_{\mathcal{A}}} \Pr(\mathbf{z}_{s',\mathcal{A}}|s',\mathcal{A})$ is the sum of the number of particles in $\mathcal{B}'$ that are covered by $\mathcal{A}$ times 1 ($\Pr(\mathbf{z}_{s',\mathcal{A}}|s',\mathcal{A}) = 1$, according to Assumption 2), which is exactly the same as $PCF_{\mathcal{B}'}(\mathcal{A})$.

□

In cases where Assumption 2 does not hold, particle coverage can be considered an approximation to expected coverage. Its key advantage is that computing $f_{\mathcal{B}'}$ does not require hypothetical belief updates, as one can iterate over the particle belief and simply count the number of particles that are covered by $\mathcal{A}$, making it practical for real-time applications. Moreover, it is a member of a class of coverage functions that are known to be submodular [Krause and Golovin, 2014, Takamura and Okumura, 2009] so we can employ greedy maximization to approximately maximize $f_{\mathcal{B}'}$. Our experiments show that $f_{\mathcal{B}'}$ is a good choice of utility function for selective detection in real time, leading to excellent tracking performance at a fraction of the computational cost.

Note that we formulate Assumption 2 merely for analysis purposes: to describe a set of cases in which particle coverage and expected coverage are identical. Assumption 2 is not a restrictive condition for applying PartiMax, described below. On the contrary, in Section 6.6 we present excellent results for PartiMax on a real-life dataset for which Assumption 2 does not hold.

Furthermore, while we define particle coverage for the case of an unweighed particle filter, the concept is more general. In essence, the particle coverage of a pixel box is the cumulative probability mass concentrated on the states that are covered by the pixel box.

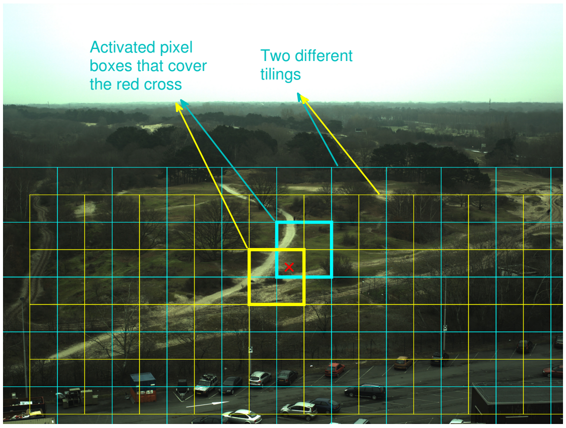


Figure 6.4: An example tile coding with two tilings. The highlighted tiles show the two pixel boxes that cover the red cross.

Thus, any method that approximates a belief can be used to compute particle coverage by simply computing the probability mass concentrated on a set of states. For example, for a weighted particle filter, the particle coverage of a pixel box is just the sum of the weights of the particles covered by the pixel box.

6.4 PartiMax

In this section, we propose *PartiMax*, which combines the complementary benefits of PCF and stochastic greedy maximization for selective detection. Moreover, rather than merely naively applying them together, we exploit the unique structure of PCF to develop a better approach for sampling pixel boxes that is guaranteed to sample pixel boxes with high coverage, thus offering a further increase in performance. PartiMax is based on the key insight that sampling pixel boxes with a probability that is directly proportional to their particle coverage leads to strong theoretical guarantees on the expected utility. Thus, we prove error bounds for PartiMax that are independent of the number of available pixel boxes n, the number of particles in the particle filter m, or the number of pixel boxes to be selected k.

Greedy maximization and stochastic greedy maximization assume oracle access to the utility function and thus compute the marginal gain for every pixel box in every iteration. Generally, computing particle coverage function given a pixel box requires iterating over the particles to count how many fall in the space covered by the pixel box. Unlike greedy maximization, PartiMax does not explicitly compute particle coverage for each pixel box on the fly but instead maintains the particle coverage by updating it in every iteration. Using an approach inspired by *tile coding*, a popular reinforcement learning technique for coding continuous state spaces, PartiMax is able to compute and maintain the particle coverage of every pixel box without having to visit n pixel boxes or m particles in every iteration.

A tile coding consists of many *tilings*. Each tiling is a set of *tiles*, which in our setting are pixel boxes. The pixel boxes in a tiling partition the state space S, i.e., they are disjoint and completely cover S. For example, Figure 6.4 shows two tilings in blue and yellow. Typically, different tilings have the same size pixel boxes but start at a fixed offset from each other, as in the figure. Since the pixel boxes in a given tiling form a partition, there is exactly one pixel box in each tiling that covers a given state s'. If we represent each tiling as an array, locating the pixel box that covers a given state s' requires only simple arithmetic involving the size of the pixel boxes and the offset between the tilings. Figure 6.4 highlights the two pixel boxes, one in each tiling, that cover a given state (red cross). Thus, by representing the entire space of pixel boxes as multiple tilings, the set of pixel boxes $\mathcal{T}_{s'}$ that covers a given state s' can be identified in constant time.

In reinforcement learning, tile codings are used to discretize continuous state spaces in order to approximate a value function. Here, we use it differently, just as a scheme for dividing an image into overlapping pixel boxes. The benefit of this approach is that it enables PartiMax to maintain Δ_f efficiently, by providing constant-time access to the set $\mathcal{T}_{s'}$ of all pixel boxes that cover a given state s', i.e., $\mathcal{T}_{s'} = \{i \in \mathcal{X} : i \text{ covers } s'\}$.

Algorithm 6 `PartiMax`$(\mathcal{B}', \mathcal{X}, k)$

```
1: $\langle \Phi, \Delta_f \rangle \leftarrow$ initialize$(\mathcal{B}', \mathcal{X})$
2: $\mathcal{A}^S \leftarrow \emptyset$.
3: for $l = 1$ to $k$ do
4:     $\mathcal{R} \leftarrow$ sampleP$(r, \mathcal{B}', \mathcal{X}, \mathcal{A}^S)$
5:     $i' \leftarrow \arg\max_{i \in \mathcal{R}} \Delta_f(i|\mathcal{A}^S)$
6:     $\mathcal{A}^S \leftarrow \mathcal{A}^S \cup i'$
7:     $\langle \Delta_f, \Phi \rangle \leftarrow$ update$(\Delta_f, \Phi, i', \mathcal{A}^S, \mathcal{X})$
8: end for
9: return $\mathcal{A}^S$
```

Algorithm 6 shows pseudocode for PartiMax. It starts by calling `initialize` (Algorithm 7), which returns two data structures, Φ and Δ_f. $\Phi(i|\emptyset)$ stores for each i the set of particles in $\mathcal{B}'$ that i covers; and $\Delta_f(i|\emptyset)$ is the number of particles that are covered by i. For each particle $s' \in \mathcal{B}'$, `initialize` calls `covers`, which uses the tile coding to find the set of pixel boxes $\mathcal{T}_{s'}$ that cover that particle. For every activated pixel box, $i \in \mathcal{T}_{s'}$, $\Delta_f(i|\emptyset)$ is incremented and s' is added to the set of particles $\Phi(i|\emptyset)$.

Algorithm 7 `initialize`$(\mathcal{B}', \mathcal{X})$

```
1: $\Phi(i|\emptyset) \leftarrow \emptyset \, \forall \, i \in \mathcal{X}$
2: $\Delta_f(i|\emptyset) \leftarrow 0 \, \forall \, i \in \mathcal{X}$
3: for $s' \in \mathcal{B}'$ do
4:     $\mathcal{T}_{s'} \leftarrow \texttt{covers}(s')$
5:     $\Phi(i|\emptyset) \leftarrow \Phi(i|\emptyset) \cup \{s'\} \, \forall \, i \in \mathcal{T}_{s'}$
6:     $\Delta_f(i|\emptyset) = \Delta_f(i|\emptyset) + 1 \, \forall \, i \in \mathcal{T}_{s'}$
7: end for
8: return $\langle \Phi, \Delta_f \rangle$
```

Once Φ and Δ_f are returned by `initialize`, PartiMax proceeds like stochastic greedy maximization, adding in each iteration the pixel box i' that maximizes the marginal gain from $\mathcal{R}$, a subset of $\mathcal{X}$ of size r. Since going over all pixel boxes is too expensive, PartiMax calls Algorithm 8 to obtain $\mathcal{R}$, a subset of $\mathcal{X}$ of size r ($r << n$). However, unlike stochastic greedy maximization, $\mathcal{R}$ is not sampled uniformly randomly but instead Algorithm 8 samples from a distribution such that the probability that i is included in $\mathcal{R}$ is directly proportional to the particle coverage of i.

Algorithm 8 `sampleP`$(r, \mathcal{B}', \mathcal{X}, \mathcal{A}^S)$

```
1: $\mathcal{R} \leftarrow \emptyset$
2: while $|\mathcal{R}| < r$ do
3:     $s' \sim \text{Unif}(\Psi(\mathcal{B}'|\mathcal{A}^S))$          ▷ $\Psi(\mathcal{B}'|\mathcal{A}^S)$ denotes the set of particles
4:                                  in $\mathcal{B}'$ that are not covered by any pixel box in $\mathcal{A}^S$
5:     $\mathcal{T}_{s'} \leftarrow \texttt{covers}(s')$
6:     $i \sim \text{Unif}(\mathcal{T}_{s'})$                    ▷ uniformly random sample from $\mathcal{T}_{s'}$
7:     $\mathcal{R} \leftarrow \mathcal{R} \cup i$
8: end while
9: return $\mathcal{R}$
```

In general, sampling from such a distribution would be difficult, but with PCF we can do this efficiently. Algorithm 8 first uniformly randomly samples a particle from the belief. If the particle is not covered by $\mathcal{A}^S$, then it uses tile coding to find the set of pixel boxes $\mathcal{T}_{s'}$ that cover s' and adds a pixel box uniformly randomly from $\mathcal{T}_{s'}$. This is repeated until r pixel boxes are added to $\mathcal{R}$.

At the end of each iteration, PartiMax calls `update` (Algorithm 9), which updates $\Delta_f(i|\mathcal{A}^S)$ and $\Phi(i|\mathcal{A}^S)$ for every $i \in \cup_{s' \in \Phi(i'|\mathcal{A}^S)} \mathcal{T}_{s'}$. It starts by iterating over the particles s' in $\Phi(i'|\mathcal{A}^S)$ and for each particle uses the tile coding to find $\mathcal{T}_{s'}$. For every pixel box $i \in \mathcal{T}_{s'}$, $\Delta_f(i|\mathcal{A}^S)$ is decremented and s' is removed from $\Phi(i|\mathcal{A}^S)$, to account for the fact that i' now covers s' and thus the marginal gain of i is reduced. The marginal gain of every other i remains unchanged. Similarly, $\Phi(i|\mathcal{A}^S)$ is updated by subtracting

s' from $\Phi(i|\mathcal{A}^S)$ for every i in $\mathcal{T}_{s'}$.

Algorithm 9 `update`$(\Delta_f, \Phi, i', \mathcal{A}^S, \mathcal{X})$

```
1: for $s' \in \Phi(i')$ do
2:     $\mathcal{T}_{s'} \leftarrow$ covers$(s')$
3:     $\Delta_f(i|\mathcal{A}^S) = \Delta_f(i|\mathcal{A}^S) - 1 \; \forall \, i \in \mathcal{T}_{s'}$
4:     $\Phi(i|\mathcal{A}^S) \leftarrow \Phi(i|\mathcal{A}^S) \setminus s' \; \forall \; i \in \mathcal{T}_{s'}$
5: end for
6: return $\langle \Delta_f, \Phi \rangle$
```

6.5 Analysis

We now establish bounds on the cumulative error of PartiMax that are independent of the problem size. We start with a lemma that shows that the probability of adding i to $\mathcal{R}$ via Algorithm 8 is directly proportional to the marginal gain of i.

Lemma 11. *Let $i =$ `sampleP`$(1, \mathcal{B}', \mathcal{X}, \mathcal{A}^S)$ then $\Pr_{\mathcal{A}^S}(i = i_j) = \frac{1}{c}\Delta(i_j|\mathcal{A}^S)$, where $c = tm'$ is a constant, $\mathcal{X} = \{1, 2, \ldots, n\}$ is the set of n pixel boxes, $\mathcal{A}^S$ is any set in $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$, m' is the set of particles in $\mathcal{B}'$ that are not covered by any pixel box in $\mathcal{A}^S$, $\Delta_f(i_j|\mathcal{A}^S) = f_{\mathcal{B}'}(\mathcal{A}^S \cup i_j) - f_{\mathcal{B}'}(\mathcal{A}^S)$ is the number of particles that are covered by i_j but not covered by any pixel box in $\mathcal{A}^S$. Here, $f_{\mathcal{B}'}(\mathcal{A}) = |\cup_{j \in \mathcal{A}} \mathcal{I}^j_{\mathcal{B}'}|$ is the number of particles in $\mathcal{B}'$ that are covered by the pixel boxes in $\mathcal{A}$.*

Proof. The probability that `sampleP`$(1, \mathcal{B}', \mathcal{X}, \mathcal{A}^S)$ returns i_j is product of probability that i_j is included in the set $\mathcal{T}_{s'}$ on line 5 in Algorithm 8, times the probability that i_j is sampled from $\mathcal{T}_{s'}$.

$$\Pr_{\mathcal{A}^S}(i = i_j) = \Pr(i_j \in \mathcal{T}_{s'}) \Pr(i_j \text{ is sampled from } \mathcal{T}_{s'}) \tag{6.9}$$

The $\Pr(i_j \in \mathcal{T}_{s'})$ is the number of ways i_j can be sampled in $\mathcal{T}_{s'}$ on line 5 in Algorithm 8. i_j can be sampled in $\mathcal{T}_{s'}$ only if i_j covers s'. Thus, the number of ways i_j can be sampled in $\mathcal{T}_{s'}$ is the number of particles that i_j covers but $\mathcal{A}^S$ does not cover, that is, every particle in $\Psi(\mathcal{B}|\mathcal{A}^S)$. The total number of particles that can be sampled is m.

Thus,

$$\Pr(i_j \in \mathcal{T}_{s'}) = \frac{\text{number of particle } i_j \text{ covers but } \mathcal{A}^S \text{ does not}}{m'} \tag{6.10}$$

$$= \frac{f(\mathcal{A}^S \cup i_j) - f(\mathcal{A}^S)}{m'} \tag{6.11}$$

$$= \frac{\Delta_f(i|\mathcal{A}^S)}{m'}, \tag{6.12}$$

where $\Delta_f(i|\mathcal{A}^S) = f(\mathcal{A}^S \cup i) - f(\mathcal{A}^S)$ is the marginal gain of adding i_j to $\mathcal{A}^S$, that is the increase in the number of particles covered by $\mathcal{A}^S \cup i_j$ caused by the addition of i_j to $\mathcal{A}^S$. (Here $f(\mathcal{A}^S)$ is the number of particles covered by $\mathcal{A}^S$.)

The probability that i_j is sampled from $\mathcal{T}_{s'}$ (uniformly randomly) is $\frac{1}{|\mathcal{T}_{s'}|}$. Lets say there are t tilling in total, so that, $|\mathcal{T}_s'| = t$. Thus, using (6.9):

$$\Pr_{\mathcal{A}^S}(i = i_j) = \frac{1}{tm'}\Delta_f(i_j|\mathcal{A}^S) \tag{6.13}$$

□

Next, we show that PartiMax is guaranteed to be near-optimal.

Theorem 16. *Let F be a non-negative set function over a collection of sets $\mathcal{A}^+ = \{\mathcal{A}_1, \mathcal{A}_2 \ldots \mathcal{A}_v\}$ and let $\mathcal{A}^* = \arg\max_{\mathcal{A}\in\mathcal{A}^+} F(\mathcal{A})$, let $\mathcal{A}' = \arg\max_{\mathcal{A}\in\mathcal{R}} F(\mathcal{A})$, such that $\mathcal{R}$ is a set of r sets sampled independently from a probability distribution over $\mathcal{A}^+$ such that probability that a given set $\mathcal{A} \in \mathcal{A}^+$ is sampled is $\Pr(\mathcal{A}) = \frac{1}{c}F(\mathcal{A})$ for all $\mathcal{A} \in \mathcal{A}^+$, where c is a scalar constant, such that, $\frac{c}{F(\mathcal{A}^*)} - 1 \leq r$. Then,*

$$F(\mathcal{A}^*) - \mathbb{E}F(\mathcal{A}') \leq (\frac{r}{1+r})^r F(\mathcal{A}^*). \tag{6.14}$$

Proof. Let $p_1, p_2 \ldots p_v$ denote $\Pr(\mathcal{A}_1), \Pr(\mathcal{A}_2) \ldots \Pr(\mathcal{A}_v)$ respectively. Also without loss of generality, we assume $p_1 \geq p_2 \geq \ldots p_v$. Consequently, it follows, $F(\mathcal{A}_1) \geq F(\mathcal{A}_2) \geq \cdots \geq F(\mathcal{A}_v)$. Note $\mathcal{A}^* = \mathcal{A}_1$.

The probability that every time a sample is drawn, it is not $\mathcal{A}_1$ is $(1 - p_1)$. The probability that $\mathcal{A}_1$ is not drawn in r independent samples drawn from $\mathcal{P}$ is $(1 - p_1) \times (1 - p_1) \times \ldots$(r times).

$$\Pr(\mathcal{A}_1 \notin \mathcal{R} : |\mathcal{R}| = r) = (1 - p_1)^r \tag{6.15}$$

Since,

$$\Pr(\mathcal{A}_1 \in \mathcal{R}) = 1 - \Pr(\mathcal{A} \notin \mathcal{R}), \tag{6.16}$$

implies,

$$\Pr(\mathcal{A}_1 \in \mathcal{R}) = 1 - (1 - p_1)^r. \tag{6.17}$$

The expected value of $F(\mathcal{A}')$ is:

$$\begin{aligned}
\mathbb{E}[F(\mathcal{A}')] &= \Pr(\mathcal{A}_1 \in \mathcal{R}) \max_{\mathcal{A}'' \in \mathcal{R}} F(\mathcal{A}'') + \Pr(\mathcal{A}_1 \notin \mathcal{R}) \max_{\mathcal{A}'' \in \mathcal{R}} F(\mathcal{A}'') && (6.18)\\
&= \Pr(\mathcal{A}_1 \in \mathcal{R}) F(\mathcal{A}_1) + \Pr(\mathcal{A}_1 \notin \mathcal{R}) \max_{\mathcal{A}'' \in \mathcal{R}} F(\mathcal{A}'') && (6.19)\\
&\geq \Pr(\mathcal{A}_1 \in \mathcal{R}) F(\mathcal{A}_1) && (6.20)\\
&= (1 - (1 - p_1)^r) F(\mathcal{A}_1) && (6.21)
\end{aligned}$$

Eq. (6.18) follows from the definition since $F(\mathcal{A}') = \max_{\mathcal{A} \in \mathcal{R}} F(\mathcal{A})$, Eq. (6.19) follows from the assumption that let $\mathcal{A}_1 = \mathcal{A}^*$, thus if $\mathcal{A}_1 \in \mathcal{R}$ it the element that maximizes F, Eq. (6.20) is true because we are removing a non-negative quantity from (6.19), Eq. (6.21) is simple substitution of $\Pr(\mathcal{A}_1 \in \mathcal{R})$ as computed before.

This implies,

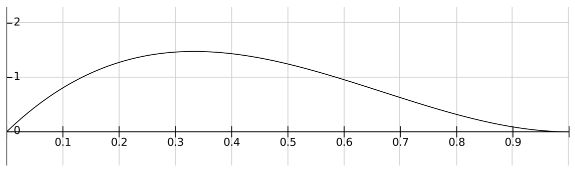
$$\begin{aligned}
\mathbb{E}[F(\mathcal{A}')] &\geq (1 - (1 - p_1)^r) F(\mathcal{A}_1) && (6.22)\\
-\mathbb{E}F(\mathcal{A}') &\leq -(1 - (1 - p_1)^r F(\mathcal{A}_1) && (6.23)\\
F(\mathcal{A}_1) - \mathbb{E}F(\mathcal{A}') &\leq F(\mathcal{A}_1) - (1 - (1 - p_1)^r) F(\mathcal{A}_1) && (6.24)\\
&= cp_1 - (1 - (1 - p_1)^r) cp_1 && (6.25)\\
&= (1 - p_1)^r cp_1. && (6.26)
\end{aligned}$$

Eq. (6.22) follows from (6.21), Eq. (6.23) is obtained by multiplying both sides by -1 and inverting the inequality, Eq. (6.23) follows from adding $F(\mathcal{A}_1)$ (non-negative) on both sides of inequality, Eq. (6.25) follows from the assumption that $c \Pr(A) = F(\mathcal{A})$ for all $\mathcal{A}$, thus, $p_1 = \Pr(\mathcal{A}_1) = \frac{F(\mathcal{A}_1)}{c}$.

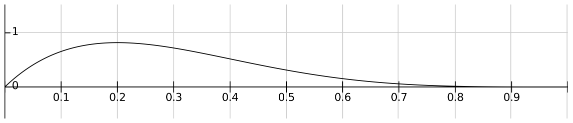
The maxima of the expression $(1 - p_1)^r cp_1$ occurs at $p_1 = \frac{1}{r+1}$:

$$\begin{aligned}
\frac{\partial((1 - p_1)^r cp_1)}{\partial p_1} &= \frac{\partial(1 - p_1)^r}{\partial p_1} cp_1 + \frac{\partial cp_1}{\partial p_1}(1 - p_1)^r && (6.27)\\
&= r(1 - p_1)^{r-1}(-1)cp_1 + c(1 - p_1)^r && (6.28)\\
&= (1 - p_1)^{r-1}[-crp_1 + c - cp_1] && (6.29)\\
&= (1 - p_1)^{r-1} c[-p_1(1 + r) + 1] && (6.30)
\end{aligned}$$

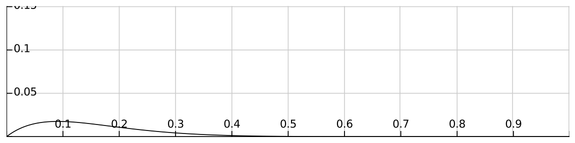
Equating (6.30) to zero, gives: $p_1 = \frac{1}{r+1}$ (unless $p_1 = 1$).



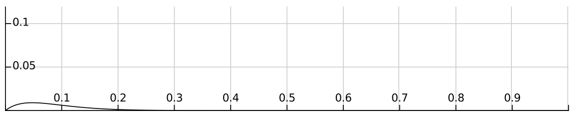
(a)



(b)



(c)



(d)

Figure 6.5: Figure illustrating local maxima of expression $((1-p_1)^r)cp_1$ on y-axis and p_1 varying from 0 to 1 on x-axis for different values of r and c: (a) r = 2, c=10; (b) r=4, c=10; (c) r=10, c=0.5; r=20, c=0.5

This maxima can be confirmed by the plots of $(1-p_1)^r cp_1$ for various values of r as shown in Figure 6.5.

Substituting $p_1 = \frac{1}{r+1}$ in (6.25),

$$F(\mathcal{A}_1) - \mathbb{E}[F(\mathcal{A}')] \leq ((1-\frac{1}{r+1})^r)c\frac{1}{r+1} \tag{6.31}$$

$$\tag{6.32}$$

Let $\frac{c}{F(\mathcal{A}_1)} - 1 \leq r \implies \frac{1}{p_1} \leq r+1 \implies p_1 \geq \frac{1}{r+1}$. This implies,

$$F(\mathcal{A}_1) - \mathbb{E}[F(\mathcal{A}')] \leq ((1-\frac{1}{r+1})^r)cp_1. \tag{6.33}$$

Replacing cp_1 by $F(\mathcal{A}_1)$ in (6.33) gives,

$$F(\mathcal{A}_1) - \mathbb{E}[F(\mathcal{A}')] \leq (\frac{r}{r+1})^r F(\mathcal{A}_1). \tag{6.34}$$

Since $F(\mathcal{A}_1)$ is $F(\mathcal{A}^*)$, this implies,

$$F(\mathcal{A}^*) - \mathbb{E}[F(\mathcal{A}')] \leq (\frac{r}{r+1})^r F(\mathcal{A}^*). \tag{6.35}$$

$\square$

The above theorem guarantees that, granted access to a probability distribution such that $\Pr(\mathcal{A}) = \frac{1}{c}F(\mathcal{A})$, there exists a tight theoretical guarantee for selecting $\mathcal{A}' = \arg\max_{\mathcal{A}\in\mathcal{R}} F(\mathcal{A})$ for $\frac{c}{F(\mathcal{A}^*)} - 1 \leq r$.

Directly applying Theorem 16 and Lemma 11 yields the following lemma, which shows that the marginal gain of PartiMax $\Delta_f(i'|\mathcal{A}^S)$ in each iteration is at least $(1 - (\frac{r}{r+1})^r)\Delta_f(i^*|\mathcal{A}^S)$, where $i^* = \arg\max_{i\in\mathcal{X}\setminus\mathcal{A}^S} \Delta(i|\mathcal{A}^S)$.

Lemma 12. *Let $\mathcal{X} = \{1, 2, \ldots, n\}$ be the set of n pixel boxes, $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$, and $\mathcal{A}^S$ be any set in $\mathcal{A}^+$. Let $\mathcal{B}'$ be a particle belief with m particles, $f_{\mathcal{B}'}(A)$ be the number of particles covered by $\mathcal{A}$ in $\mathcal{B}'$, $f_{\mathcal{B}'}(\mathcal{A}) = |\cup_{j\in\mathcal{A}} \mathcal{I}^j_{\mathcal{B}'}|$ and let $\Delta_f(i|\mathcal{A}) = f_{\mathcal{B}'}(\mathcal{A} \cup i) - f_{\mathcal{B}'}(\mathcal{A})$ denote the marginal gain of adding i to $\mathcal{A}$. Let $i^* = \arg\max_{i\in\mathcal{X}\setminus\mathcal{A}^S} \Delta_f(i|\mathcal{A}^S)$ and let $i' = \arg\max_{i\in\mathcal{R}} \Delta_f(i|\mathcal{A}^S)$, where $\mathcal{R} = \texttt{sampleP}(\texttt{r}, \mathcal{B}', \mathcal{X}, \mathcal{A}^\texttt{S})$ and $r \geq \frac{tm}{2} - 1$, where t is the number of tilings, $m = |\mathcal{B}'|$ is the number of particles in $\mathcal{B}'$. The expected marginal gain of PartiMax in an iteration, $\mathbb{E}[\Delta_f(i'|\mathcal{A}^S)|\mathcal{A}^S]$, is atleast, $(1 - (\frac{r}{r+1})^r)\Delta_f(i^*|\mathcal{A}^S)$, that is,*

$$\Delta_f(i^*|\mathcal{A}^S) - \mathbb{E}[\Delta_f(i'|\mathcal{A}^S)|\mathcal{A}^S] \leq (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}^S). \tag{6.36}$$

Proof. Follows from Theorem 16 with F as Δ_f, $\mathcal{X} \setminus \mathcal{A}^S$ as $\mathcal{A}^+$, i^* as $\mathcal{A}^*$, i' as $\mathcal{A}'$ for a given $\mathcal{A}^S$.

$$\Delta_f(i^*|\mathcal{A}^S) - \mathbb{E}[\Delta_f(i'|\mathcal{A}^S)|\mathcal{A}^S] \leq (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}^S). \tag{6.37}$$

Theorem 16 requires that $\mathcal{R}$ is made with r independent samples drawn from a distribution such that probability of sampling a pixel box i is $\Pr_{\mathcal{A}^S}(i) = \frac{1}{c}\Delta_f(i|\mathcal{A}^S)$, where c is a constant. Lemma 11 exactly shows this for $c = tm'$, where t is the number of tilings and m' is the number of samples in $\mathcal{B}'$ that are not covered by $\mathcal{A}^S$. This satisfies one condition of Theorem 16.

Theorem 16 also requires $r \geq \frac{tm'}{\Delta_f(i^*|\mathcal{A}^S)} - 1$. Now, m is the number of particles in $\mathcal{B}'$ which is naturally greater than number of particles in $\mathcal{B}'$ that are not covered by $\mathcal{A}^S$, m'. Thus, the value of $r \geq \frac{tm}{\Delta_f(i^*|\mathcal{A}^S)} - 1$ satisfies this condition.

Now, $\Delta_f(i^*|\mathcal{A}^S)$ is the number of particles covered by i^* in $\mathcal{B}'$ that are not covered by $\mathcal{A}^S$. For $\Delta_f(i^*|\mathcal{A}^S) = 0$ or $\Delta_f(i^*|\mathcal{A}^S) = 1$, Eq. (6.36) is trivially satisfied for all values of $r \geq 1$. This is because $i^* = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}^S} \Delta_f(i|\mathcal{A}^S)$ and if $\Delta_f(i^*|\mathcal{A}^S)$ is zero then $\Delta_f(i|\mathcal{A}^S)$ is zero for all i, which implies that $\mathbb{E}[\Delta_f(i'|\mathcal{A}^S)|\mathcal{A}^S]$ is also zero. Thus, (6.36) holds trivially for all integer values of $r \geq 1$

For the case when $\Delta_f(i^*|\mathcal{A}^S) = 1$, this can only happen if for all i, $\Delta_f(i|\mathcal{A}^S) = 1$ or 0. In this case, `sampleP` is designed such that it will only sample i in $\mathcal{R}$ for which $\Delta_f(i|\mathcal{A}^S) = 1$. Thus, $\mathbb{E}[\Delta_f(i'|\mathcal{A}^S)|\mathcal{A}^S] = 1$. Thus, again (6.36) holds trivially for all integer values of $r \geq 1$.

For the case when $\Delta_f(i^*|\mathcal{A}^S)$ is greater than 1, the minimum values it can have is 2. Thus, directly applying Theorem 16 as in (6.37) yields if $r \geq \frac{tm}{2} - 1$:

$$\Delta_f(i^*|\mathcal{A}^S) - \mathbb{E}[\Delta_f(i'|\mathcal{A}^S)|\mathcal{A}^S] \leq (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}^S). \tag{6.38}$$

□

Lemma 12 in turn yields the following theorem:

Theorem 17. *Let $f_{\mathcal{B}'}(A)$ be the number of particles covered by $\mathcal{A}$ in $\mathcal{B}'$, $f_{\mathcal{B}'}(\mathcal{A}) = |\cup_{j \in \mathcal{A}} \mathcal{I}^j_{\mathcal{B}'}|$ and let $\Delta_f(i|\mathcal{A}) = f_{\mathcal{B}'}(\mathcal{A} \cup i) - f_{\mathcal{B}'}(\mathcal{A})$ denote the marginal gain of adding i to $\mathcal{A}$. Let $\mathcal{A}^* = \arg\max_{\mathcal{A} \in \mathcal{A}^+} f_{\mathcal{B}'}(\mathcal{A})$ and $\mathcal{A}^S_l =$ `PartiMax`$(\mathcal{B}', \mathcal{X}, l)$ for $1 \leq l \leq k$ and $\mathcal{A}^S_k = \mathcal{A}^S$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \ldots, n\}$.*

If in every iteration `Partimax` calls `sampleP` with $r \geq \frac{tm}{2} - 1$ (lets say $r = \frac{tm}{2} - 1$), where t is the number of tilings and m is the number of samples in $\mathcal{B}'$, then,

$$\mathbb{E}[f(\mathcal{A}^S)] \geq (1 - e^{-1} - ((r/(r+1))^r) f(\mathcal{A}^*). \tag{6.39}$$

Proof. The proof follows similar logic as the proof presented for Theorem 1 in Chapter 2 except that we will introduce an expectation over $\mathcal{A}^S$ and bound the sum of *error terms* in each iteration of greedy maximization by $f(\mathcal{A}^*)$.

Let $\{i^*_1, i^*_2, \ldots, i^*_k\}$ (arbitrary order) be the k elements of $\mathcal{A}^*$. Let $\mathcal{A}^S_l$ denote the subset build by `PartiMax` if it was run for l iterations: $\mathcal{A}^S_l =$ `PartiMax`$(\mathcal{B}', \mathcal{X}, l)$.

To prove 17 we first prove an intermediary result that the expected gain of PartiMax in each iteration is atleast $\frac{1}{k}\sum_{i \in \mathcal{A}^* \setminus \mathcal{A}^S_l} \Delta_f(i|\mathcal{A}^S_l) - (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}^S_l)$. Let $i' = \arg\max_{i \in \mathcal{R}} \Delta_f(i|\mathcal{A}^S_l)$.

We start with the statement of Lemma 12, which requires r to be greater than $\frac{tm}{2} - 1$, then for a given $\mathcal{A}^S_l$

$$\mathbb{E}[\Delta_f(i'|\mathcal{A}_l^S)|\mathcal{A}_l^S] \geq \Delta_f(i^*|\mathcal{A}_l^S) - (\frac{r}{r+1})^r)\Delta_f(i^*|\mathcal{A}_l^S). \tag{6.40}$$

By definition $i^* = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}_l^S} \Delta(i|\mathcal{A}_l^S)$, thus, $\Delta_f(i^*|\mathcal{A}_l^S)$ is the maximum value of $\Delta_f(i|\mathcal{A}_l^S)$ for all $i \in \mathcal{X} \setminus \mathcal{A}_l^S$. Since $\mathcal{A}^* \subseteq \mathcal{X}$, thus $\Delta_f(i^*|\mathcal{A}_l^S)$ must be greater than the average value of $\Delta_f(i|\mathcal{A}_l^S)$ over $i \in \mathcal{A}^* \setminus \mathcal{A}_l^S$, that is,

$$\Delta_f(i^*|\mathcal{A}_l^S) \geq \frac{1}{|\mathcal{A}^* \setminus \mathcal{A}_l^S|} \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_l^S} \Delta_f(i|\mathcal{A}_l^S), \tag{6.41}$$

and since $|\mathcal{A}^* \setminus \mathcal{A}_l^S| \leq k$,

$$\Delta_f(i^*|\mathcal{A}_l^S) \geq \frac{1}{k} \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_l^S} \Delta_f(i|\mathcal{A}_l^S). \tag{6.42}$$

Using (6.42) and (6.40):

$$\mathbb{E}[\Delta_f(i'|\mathcal{A}_l^S)|\mathcal{A}_l^S] \geq \frac{1}{k} \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_l^S} \Delta_f(i|\mathcal{A}_l^S) - (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}_l^S) \tag{6.43}$$

Now we follow the same steps as in proof of Theorem 2 in the chapter Background :

$$f(\mathcal{A}^*) \leq f(\mathcal{A}^* \cup \mathcal{A}_l^S) \tag{6.44}$$

$$= f(\mathcal{A}_l^S) + \sum_{j=1}^{k} \Delta_f(i_j^*|\mathcal{A}_l^S \cup \{i_1^*, i_2^*, \dots, i_{j-1}^*\}) \tag{6.45}$$

$$\leq f(\mathcal{A}_l^S) + \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_l^S} \Delta_f(i|\mathcal{A}_l^S) \tag{6.46}$$

$$f(\mathcal{A}^*) - f(\mathcal{A}_l^S) \leq \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_l^S} \Delta_f(i|\mathcal{A}_l^S) \tag{6.47}$$

Eq (6.44) follows from monotonicity of f, Eq (6.45) is a straightforward telescopic sum, Eq (6.46) is true because f is submodular.

Eq. (6.43) showed that $k(\mathbb{E}[\Delta_f(i'|\mathcal{A}_l^S)|\mathcal{A}_l^S] - (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}_l^S)) \geq \sum_{i \in \mathcal{A}^* \setminus \mathcal{A}_l^S} \Delta_f(i|\mathcal{A}_l^S)$, using this with (6.47) gives,

$$f(\mathcal{A}^*) - f(\mathcal{A}_l^S) \leq k(\mathbb{E}[\Delta_f(i'|\mathcal{A}_l^S)|\mathcal{A}_l^S] - (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}_l^S)) \tag{6.48}$$

Using definition of i' and Δ_f, $\Delta_f(i'|\mathcal{A}_l^S) = f(\mathcal{A}_{l+1}^S) - f(\mathcal{A}_l^S)$,

$$f(\mathcal{A}^*) - f(\mathcal{A}_l^S) \leq k(\mathbb{E}[f(\mathcal{A}_{l+1}^S) - f(\mathcal{A}_l^S)|\mathcal{A}_l^S] - (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}_l^S)) \tag{6.49}$$

Taking expectation over $\mathcal{A}_l^S$,

$$\mathbb{E}[f(\mathcal{A}^*) - f(\mathcal{A}_l^S)] \leq k(\mathbb{E}[f(\mathcal{A}_{l+1}^S) - f(\mathcal{A}_l^S)] - \mathbb{E}[(\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}_l^S)]) \tag{6.50}$$

Now lets define $\beta_l = f(\mathcal{A}^*) - f(\mathcal{A}_l^S)$, and $\zeta_l = (\frac{r}{r+1})^r \Delta_f(i^*|\mathcal{A}_l^S)]$which gives:[2]

$$\mathbb{E}\beta_l \leq k(\mathbb{E}[\beta_l - \beta_{l+1} - \zeta_l] \tag{6.51}$$

$$\mathbb{E}\beta_l(1 - \frac{1}{k}) \leq \mathbb{E}\delta_{l+1} - \mathbb{E}\zeta_l \tag{6.52}$$

$$\mathbb{E}\beta_{l+1} \geq \mathbb{E}\beta_l(1 - \frac{1}{k}) + \mathbb{E}\zeta_l) \tag{6.53}$$

Let $l = 0$,

$$\mathbb{E}\beta_1 \leq (1 - \frac{1}{k})\mathbb{E}\beta_0 + \mathbb{E}\zeta_0. \tag{6.54}$$

Substituting $l = 1$ in (6.53),

$$\mathbb{E}\beta_2 \leq (1 - \frac{1}{k})\mathbb{E}\beta_1 + \mathbb{E}\zeta_1. \tag{6.55}$$

Combining (6.54) and (6.55),

$$\mathbb{E}\beta_2 \leq \mathbb{E}(1 - \frac{1}{k})^2\beta_0 + \mathbb{E}[\zeta_1 + \zeta_0] \tag{6.56}$$

Substituting $l = 2$ in (6.53),

$$\mathbb{E}\beta_3 \leq (1 - \frac{1}{k})\mathbb{E}\beta_2 + \mathbb{E}\zeta_2 \tag{6.57}$$

[2]Note that $i^* = \arg\max_{\imath \in \mathcal{X} \setminus \mathcal{A}_l^S} \Delta_f(i^*|\mathcal{A}_l^S)$ is implicitly dependent on l. This i^* is not to be confused with the elements of $\mathcal{A}^* = \{i_1^*, i_2^*, \ldots, i_k^*\}$.

Combining (6.56) and (6.57),

$$\mathbb{E}\beta_3 \leq \mathbb{E}(1-\frac{1}{k})^3\beta_0 + \mathbb{E}[\zeta_2+\zeta_1+\zeta_0] \tag{6.58}$$

Continuing like this for $l = k-1$,

$$\mathbb{E}\delta_k \leq (1-\frac{1}{k})^k\delta_0 + \sum_{l=0}^{k-1}\mathbb{E}\zeta_l \tag{6.59}$$

The sum $\sum_{l=0}^{k-1}\mathbb{E}\zeta_l = (\frac{r}{r+1})^r\sum_{l=0}^{k-1}\mathbb{E}[\Delta(i^*|\mathcal{A}_l^S)]$. Here, the sum $\sum_{l=0}^{k-1}\mathbb{E}[\Delta(i^*|\mathcal{A}_l^S)]$ is sum of expected values of k marginal gains which by definition is less than $f(\mathcal{A}^*)$, since $\mathcal{A}^* = \arg\max_{\mathcal{A}\in\mathcal{A}^+} f(\mathcal{A})$, this implies,

$$\mathbb{E}\delta_k \leq (1-\frac{1}{k})^k\delta_0 + (\frac{r}{r+1})^r f(\mathcal{A}^*) \tag{6.60}$$

Substituting $\delta_0 = f(\mathcal{A}^*) - f(\mathcal{A}_0^S)$, thus

$$\mathbb{E}\delta_k \leq (1-\frac{1}{k})^k f(\mathcal{A}^*) + (\frac{r}{r+1})^r f(\mathcal{A}^*) \tag{6.61}$$

Since $1-x \leq e^{-x}$ for all $x \in \mathbb{R}$,

$$\mathbb{E}\delta_k \leq e^{-1} f(\mathcal{A}^*) + (\frac{r}{r+1})^r f(\mathcal{A}^*) \tag{6.62}$$

Substituting $\delta_k = f(\mathcal{A}^*) - f(\mathcal{A}^S)$,

$$\mathbb{E}[f(\mathcal{A}^*) - f(\mathcal{A}^S)] \leq e^{-1} f(\mathcal{A}^*) + (\frac{r}{r+1})^r f(\mathcal{A}^*) \tag{6.63}$$

This implies,

$$\mathbb{E}[f(\mathcal{A}^S)] \geq (1 - e^{-1} - (\frac{r}{r+1})^r) f(\mathcal{A}^*) \tag{6.64}$$

$\square$

The above theorem thus establishes a bound on the error of PartiMax that is independent of the size of the problem. While we have shown the theorem under the condition that $r \geq \frac{tm}{2} - 1$, however, this condition is to bound the worst case when the maximum marginal gain a pixel box can have in a particular iteration of `PartiMax` is as low as 2, which rarely occurs in practice. Ideally, in iteration l of `PartiMax`, Theorem 16 would like r to be greater than $\frac{tm}{\Delta(i^*|\mathcal{A}_{l-1}^S)}$, which can be many times less than $\frac{tm}{2}$. Moreover, even this is not a hard condition in practice, as we showed in Figure 6.6 and Figure 6.7

that even for very low values of r, the idea of sampling a pixel boxes from a probability distribution that is directly proportional the utility of the pixel box leads to good empirical performance.

While we have shown these results for PartiMax for selective detection, Theorem 16 is applicable to any problem that involves maximization over a set function where we can sample from a probability distribution such that the probability of sampling a subset $\mathcal{A}$ is directly proportional to the value of that subset specified by the set function F.

6.6 Experiments

We evaluated PartiMax on a dataset [Schutte et al., 2016] containing approximately 2100 trajectories of people recorded by a camera taking 5120×3840 resolution images running at 6 frames per second. The trajectories were generated using the *ACF* detector [Dollár et al., 2014] and in-camera tracking [Schutte et al., 2016].

We model the state s as the person's position and velocity, $s = \langle x, y, v_x, v_y \rangle$, where x and y describe position and v_x and v_y describe velocity. Both x and y are integers in $\{0, \ldots, 5000\}$. We use a motion model that predicts the next position as:

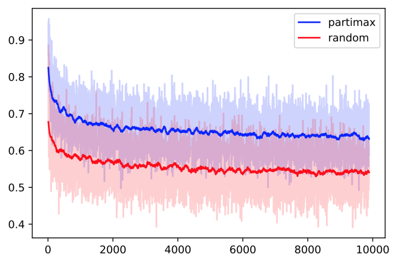
$$x_{next} = x_{curr} + v_x^{curr} + \mathcal{N}(0, \sigma_x), \tag{6.65}$$

for x and analogously for y. We use a maximum likelihood estimate of σ_x learned from the data.

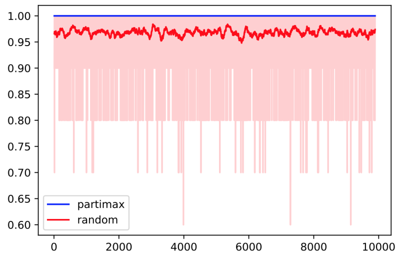
Each pixel box was 180×180 and each tiling had a 60×30 offset from the previous one. This offset was chosen because it is the size of the average *bounding box* required to bound a detected person in the scene. This setup yields approximately 7200 pixel boxes per image.

We assume access to a detector that determines with 90% accuracy whether a person is located within a given pixel box and gives a noisy observation about the location of the person if detected. Using the motion model and this detector, we maintain a particle belief $\mathcal{B}$ about the person's location using an unweighed particle filter with 250 particles. Multi-person tracking uses a separate particle filter for each person.

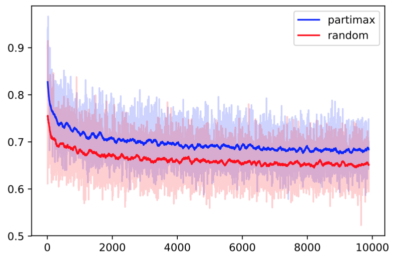
In our experiments, each algorithm selects k pixel boxes to which to apply the detector. To evaluate its performance, we sample a test trajectory from the dataset and try to track the person's movement, starting with a random belief $\mathcal{B}$ and updating it at each time step using the observations generated from the selected pixel boxes. At each time step, the agent is asked to predict the position of the person in the scene and gets a reward of +1 for correct predictions and 0 otherwise. Performance is quantified as the total



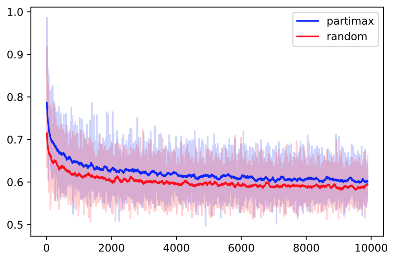
(a) Beta Distribution ($\alpha = 2; \beta = 5$)



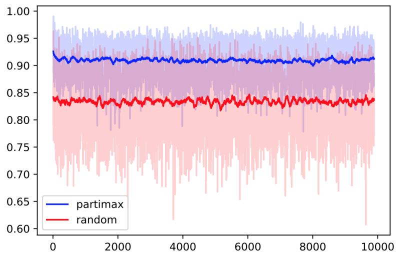
(b) Binomial Distribution ($p = 0.5$)



(c) Gaussian Distribution ($\mu = 0; \sigma = 1$)

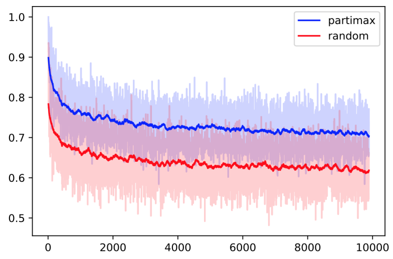


(d) Laplace Distribution ($\mu = 0; \lambda = 1$)

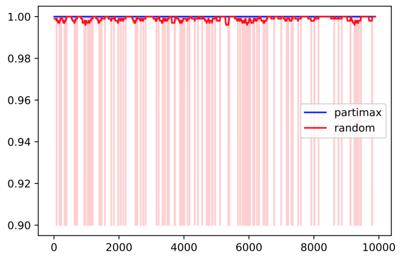


(e) Uniform Distribution ($a = 0; b = 1$)

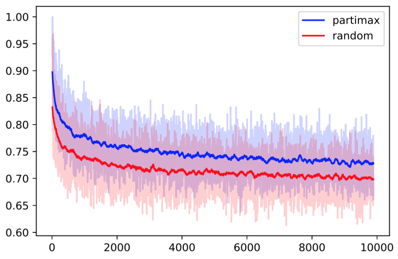
Figure 6.6: Figure showing the performance of PartiMax in a simulated setting. Here, the function F was generated randomly using the distribution mentioned above in the plots such that for $i \in \mathcal{X} = \{1, 2, \ldots, n\}$, $F(i)$ was sampled from the respective distribution. The function was later normalized to values between 0 and 1 so that 1 is the maximum value of F. Here we plot the $\max_{i \in \mathcal{R}} F(i)$ on $y-$axis, where $\mathcal{R}$ is formed (a) by sampling r (here $r = 5$) samples from a distribution such that the probability of including i in $\mathcal{R}$ is directly proportional $F(i)$ (called `partimax` in the plots; (b) by sampling r samples uniformly randomly from $\mathcal{X}$ (called `random` in the plots. The $x-$axis plots the increasing value of n. For each value of n 10 runs of the above experiment was performed and the average value is plotted in light blue and red. The solid blue and red line show the running average over 100 values of n on $x-$axis. Mainly, the plots show that even for values of r as small as 5 out of 10000, PartiMax is able to retain 70% of maximum value for all the distributions.



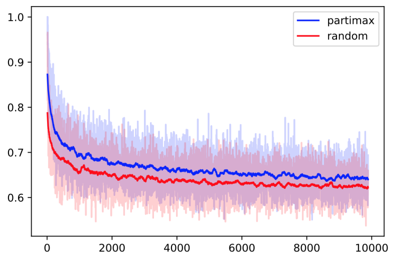
(a) Beta Distribution ($\alpha = 2; \beta = 5$)



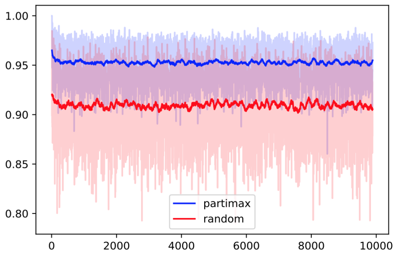
(b) Binomial Distribution ($p = 0.5$)



(c) Gaussian Distribution ($\mu = 0; \sigma = 1$)



(d) Laplace Distribution ($\mu = 0; \lambda = 1$)



(e) Uniform Distribution ($a = 0; b = 1$)

Figure 6.7: Figure showing the performance of PartiMax in a simulated setting when $r = 10$. The setting is exactly the same as Figure 6.6. The plots shows that even for small values of $r = 10$ PartiMax retains a large portion of utility.

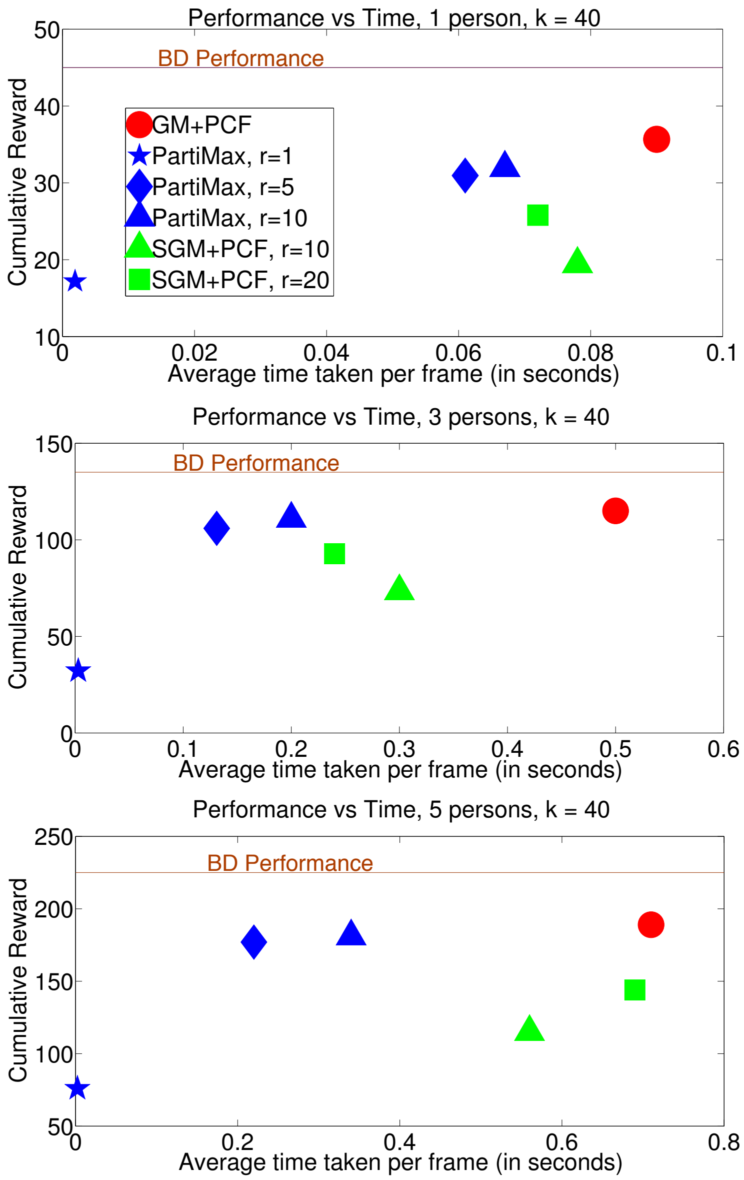


Figure 6.8: Total correct predictions vs. CPU time (seconds) for tracking one (top), three (middle) and five (bottom) people. The closer to the top-left corner, the better.

cumulative reward aggregated by the agent at the end of a trajectory over a series of 50 time steps.

The experiments were run for over 140 trajectories for 8 independent runs for 1 person tracking and 6 independent runs for 3 and 5 person tracking.

As a baseline, we compare against an efficient version of greedy maximization (GM+PCF) (in red in plots) that employs tile coding to maintain the particle coverage of each pixel box. GM+PCF is the same as PartiMax but, instead of selecting the pixel box with the highest particle coverage, in each iteration from $\mathcal{R}$, GM+PCF selects it from $\mathcal{X}$. A naive implementation of greedy maximization that computes the particle coverage of each pixel box in every iteration by going over the entire belief was too slow for a complete run and required 160 seconds to select $k = 40$ from $n = 7200$ for one person tracking. GM+PCF returns the same solution as greedy maximization but is faster.

We also compare to stochastic greedy maximization (in green in the plots) that randomly samples a subset $\mathcal{R}$ from $\mathcal{X}$ but employs tile coding to maintain the particle coverage of each pixel box. A naive implementation of stochastic greedy maximization that computes particle coverage of each pixel box from scratch takes around 0.83 seconds for $k = 40$ and $r = 10$ for one person tracking. The combination of SGM + PCF returns the same solution as stochastic greedy maximization, but faster.

Figure 6.8 shows a detailed comparison between PartiMax, greedy maximization, and stochastic greedy maximization when tracking 1, 3, or 5 people with $k = 40$. The y-axis shows the cumulative correct predictions averaged over multiple trajectories that the agent made using observations from each algorithm and the x-axis shows the time taken by each algorithm to select 40 out of 7200 pixel boxes. Thus, the top left corner indicates good tracking performance at a low computational cost. The brown line in the figure shows the tracking performance when the brute force detection is used, that is the person detector is applied to the entire image (except the part containing sky), which takes approximately 2.5 seconds.

The blue diamond and triangle at the top left corner of each plot show the superior performance and computational efficiency of PartiMax compared to the baselines. PartiMax not only matches the performance of greedy maximization, it does so extremely efficiently with a low value of r, thanks to the sampling scheme we propose. Stochastic greedy maximization's tracking performance suffers due to its random sampling, while the computational cost of GM+PCF increases with the number of people. PartiMax combines the best of both of these baselines and performs better both in terms of tracking performance and computational cost. In fact, as the number of people in the scene increases, PartiMax scales much better than any other algorithm. Overall, PartiMax is able to retain 80% percent of BD's tracking performance but is at least 10 times faster.

6.7 Conclusions & Future Work

In this chapter we proposed a new tracking system that selectively processes only a fraction of an image to track people in real time. By casting the problem of resource allocation in automated tracking systems as selection of k pixel boxes out of the n available we were able to exploit the existing methods for sensor selection to propose PartiMax. For real time resource allocation in tracking systems we proposed PCF, a simple and cheap coverage function, instead of using computationally expensive functions such as the expected coverage or information gain. By exploiting the structure in the problem we proposed a new sampling algorithm that can sample high utility pixel boxes without ever computing the utility of any pixel box. Given access to such an algorithm we showed that it is possible to obtain error bounds on maximizing any set function that is independent of the size of the problem. Finally our experiments showed that PartiMax is able to retain 80% of the original tracking performance while only processing 10% of the entire image.

The definition of PCF is such that it does not reason about the observation noise in the sensor/pixel box model; however, it still performs well even in the presence of noise. While there are many possible hypotheses for explaining this, an exciting line of future work is to study the relationship between PCF and the utility functions that reason about the noise model such as the expected coverage and information gain. Many applications like active learning and influence maximization involves maximization of information gain. Thus, exactly establishing the conditions under which a computationally cheap coverage function can closely approximate the information gain (one case is in absence of any noise) can lead to algorithms that scale easily to large problems as demonstrated by PartiMax.

Our formulation of the selective detection problem specifically when the utility function is PCF is closely related to the submodular set cover problem [Wolsey, 1982, Hochbaum, 1996]. Extending PartiMax to abstract settings for maximizing coverage functions is another useful avenue for future work. In particular, PartiMax is able to exploit the structure in the tracking problem with help of the tile coding representation of the pixel boxes. Thus, a key idea is to identify other problems where similar structure is present. The sampling scheme we propose samples sets that have high particle coverage and extending it to other interesting utility functions such as information gain is an exciting idea. The sampling algorithm, notably, is an instance of sampling from submodular point processes [Iyer and Bilmes, 2015]. Sampling from point processes is an interesting and growing field and its application for sensor/pixel box selection merits further investigation.

The methods presented in this and the previous chapters assume a model of the world to select the best actions to take to reduce uncertainty about the state of the world. Consequently, their performance is dependent on a precise and carefully designed model that may require domain expertise. In many cases such a model of the world is difficult to obtain. Moreover in real-life many of the several assumptions made to derive the model may break leading to bad performance. Thus, in the next chapter we present an model-free approach to active perception using deep reinforcement learning methods that enables an agent to learn the optimal policy to follow to reduce uncertainty in its belief.

7 Deep Active Perception

7.1 Introduction

The previous chapters focused on decision-theoretic methods for active perception that require a learned model of the world. Commonly referred to as *model-based* reinforcement learning or planning methods, these methods require the probabilities of the real-world events expressed by a model of the world [Sutton and Barto, 1998]. However, modelling real-life problems often requires expert knowledge of multiple domains and even then, in many cases the best models are limited in their representations and expressions of the real-world complexities. *Model-free* reinforcement learning methods like *Q-learning* aim to directly learn the optimal policy without explicitly modelling the world [Watkins, 1989]. *Deep reinforcement learning* [Mnih et al., 2015] combines reinforcement learning with *representation learning* [Bengio et al., 2013] to learn efficient representations of the world and the optimal policy directly from the raw sensor data and from the interaction of the agent with the world enabling an *end-to-end* approach. In this chapter we present a deep neural network architecture for active perception based on deep reinforcement learning that enables an agent to learn the optimal behaviour for reducing uncertainty in its belief. We present Deep Anticipatory Networks (DANs): a deep neural network architecture that enables an agent to take actions to reduce its uncertainty about its current and future states. DAN consists of two neural networks: a Q network that outputs the Q values of each available action for an agent and a model, M network, that predicts the state of the world based on some partial observations/glimpses of the world. The agent follows the policy that is greedy with respect to the Q values that are generated by the Q network. The main idea behind DAN is to train the Q network on a reward function such that the Q network is rewarded if and only if the M network correctly predicts the state of the world from the partial glimpses collected by the policy generated by the Q network. The M network is trained to predict the current state of the world based on the partial observations/glimpses collected by the Q network in a supervised fashion given some ground truth data.

Ideally, an end-to-end approach should require minimal supervision from the human user, even in terms of the reward function used to train the network, for example, when training an agent to play ATARI games it is possible to signal the agent if it won or lost without explicit human supervision [Mnih et al., 2015]. However, person tracking largely remains a supervised learning problem where an annotated dataset of true positions of people is important to cross-validate if the agent correctly predicted the true state or not. Consequently, for training DAN for sensor selection for tracking we assume a dataset of ground truth annotations is available. For the sensor selection problem such a dataset can be collected by investing a one time effort offline to process/transfer the images from all the sensors accurately and in high quality to yield the true positions of people in the entire scene. The result of this processing can be used as a ground truth dataset to train the DAN architecture to learn deep representations and the optimal policy for the sensor selection task. We assume access to such a dataset is available to train DAN in a similar fashion to Generative Adversarial Networks (GANs) [Goodfellow et al., 2014] by simultaneously training the Q and the M network on small mini-batches of data. By simultaneously training the Q and M network we hope that the Q and M network will simultaneously learn efficient representations that are useful to take actions and to correctly predict the state of the world respectively. Finally, we apply DAN to learn the optimal policy for the sensor selection task where an agent at each time step must select one of out of the n available sensors to best track people in a simulated problem.

In the rest of the chapter we give a small background of deep reinforcement learning methods. This is followed by the description of the DAN architecture and its training procedure and finally the experiments.

7.2 Deep Q Learning

Reinforcement learning methods enable an agent to directly learn π^* from the interaction of the agent with the environment. The basic idea behind reinforcement learning is to *estimate* the Q values of each action of the agent by using the Bellman equation as an iterative update. For example, Q-learning in each iteration m (Q_0 can be initialized randomly) performs the following update:

$$Q_{m+1}(b,\mathcal{A}) = [\overbrace{\rho(b) + \max_{\mathcal{A}'} Q_m(b^{\mathcal{A}}_{\mathbf{z}},\mathcal{A}')}^{target\ value} - Q_m(b,\mathcal{A})]. \tag{7.1}$$

After a number of iterations (for large m) the Q-estimates, in practice, converge to a close approximation of Q^* (for a fixed t and hence t is omitted from the notation).

Deep reinforcement learning methods like deep Q networks (DQN) (or deep Q learning) combine reinforcement learning with *deep neural networks* to learn hidden representations of the world and the control policy π^* simultaneously. Deep neural networks are composed of multiple layers of linear and non-linear operators parametrized by a set of weights denoted by θ. Given some training data in the form of inputs and outputs these networks can be trained to approximately learn the relationship between the inputs and outputs. For example, let $b^t = \langle \mathcal{A}^0, \mathbf{z}^1, \ldots, \mathcal{A}^{t-1}, \mathbf{z}^t \rangle$ denote the history of actions and observations that the agent has obtained until time step t. Given a dataset of $\langle b^t, \mathcal{A}^t \rangle$ pairs as inputs and the Q-value of each $\langle b^t, \mathcal{A}^t \rangle$ pair as the outputs a deep neural network can be trained to learn the underlying Q function approximately. DQN combines the training of deep neural networks with Q learning to learn the underlying Q function directly from the experience of the agent.

If θ denotes the parameters of the neural network then the Q function can be denoted by $Q(b^t, \mathcal{A}^t|\theta)$. Instead of updating the individual Q values as in Q learning, now the parameters θ can be updated iteratively based on a loss function:

$$L(b^t,\mathcal{A}^t|\theta_m) = \mathbb{E}_{b^t,\mathcal{A}^t}[\rho(b^t) + \max_{\mathcal{A}'^t} Q(b^{t+1},\mathcal{A}'^t|\theta_m) - Q(b^t,\mathcal{A}^t|\theta_m)], \tag{7.2}$$

$$\theta_{m+1} = \theta_m + \alpha \nabla_\theta L(\theta_m), \tag{7.3}$$

where $\nabla_\theta L(\theta_m)$ denotes the gradient of the loss function with respect to θ and α denotes the learning rate. The above described update of θ based on the expected value of gradient across all possible $\langle b^t, \mathcal{A}^t \rangle$ is called the gradient descent algorithm. Since it is computationally expensive to compute the expected gradient across all possible values of $\langle b^t, \mathcal{A}^t \rangle$ pairs, stochastic gradient descent approximates the expected gradient by evaluating it over a small set of data and is observed to perform well for training deep neural networks in practice.

Deep reinforcement learning removes the need to engineer complex features by combining representation learning with reinforcement learning algorithms. For example, DQNs are able to learn control policies for playing Atari games directly from the raw pixel data frames [Mnih et al., 2013]. DQN consists of many *convolutional* layers stacked on top of some *fully connected* layers that output the Q values of the actions available to the agent given as input a history of the last few data frames of the game. These networks can be trained end-to-end to learn efficient representations of the state and the optimal policy.

To *stabilize* training DQN uses the following three techniques. First, the experience of the agent is stored in an experience replay buffer in the form of an experience tuple $\langle b^t, \mathcal{A}^t, r^{t+1}, b^{t+1} \rangle$, where r^{t+1} is the immediate reward obtained by the agent for taking action $\mathcal{A}^t$ in belief b^t and transitioning to belief b^{t+1}. To update the parameters θ a random mini batch of experience tuples is drawn from the experience replay and θ is updated according to the equations 7.2 and 7.3. This is to remove the correlation between sequentially obtained samples as stochastic gradient descent works best when the data is i.i.d (independent and identically distributed). Second, a separate target Q network (separate instance of the main Q network) is maintained to provide the target values for updating θ, except that its parameters are updated to match the parameters of the main network every few iterations. This is done to avoid constant shifting of the target value and to provide a constant and stable target value to update θ, the parameters of the main Q network. Third, an adaptive rate of learning α is used instead of a fixed scalar value.

Deep Recurrent Q Networks

Deep recurrent Q networks (DRQN) [Hausknecht and Stone, 2015] extend DQN by adding a recurrent unit, specifically long-short term memory (LSTM) [Hochreiter and Schmidhuber, 1997], to the original architecture of DQN. The recurrent unit can be trained to summarize the history of observations and actions. Instead of a history of the last few data frames as input to the network, the input to DRQN is only the immediate past data frame. The agent effectively learns to maintain a hidden representation of the world via the recurrent unit and takes actions dependent on the output of the recurrent unit to maximize the reward.

In general, existing applications of deep reinforcement learning focus on tasks that express the goal of the agent via a state based reward function. For active perception the aim of the agent is to reduce uncertainty in its future belief as an end goal, thus the reward function is solely a function of the future beliefs of the agent, $\rho(b^t)$. The main forte of deep reinforcement learning methods such as DQN and DRQN is that they learn a deep hidden representation of the state of the world from the raw input. Since an explicit representation of the belief of the agent is not available in this case it is not possible to quantify the uncertainty in the belief of the agent. To tackle this problem we introduce DAN that addresses a broad class of problems where the aim of the agent is to take actions to best predict the current or future state of the world.

7.3 Deep Anticipatory Networks

In this section we introduce DANs: Deep Anticipatory Networks. DAN is named so because it enables an agent to take actions to anticipate the future state of the world accurately. Continuing from the earlier formulation, the goal of the agent is to take actions such that they maximize the information gain of the agent. However, without an explicit belief and model of the world it is not possible to compute the information gain of each action of the agent to find the one that has the maximum information gain. To tackle this challenge we propose DAN. DAN consists of two different networks (or agents): a Q network and a model (M) network. The role of the Q network is to take as input the history of previous observations and actions of the agent and output the Q values of all available actions of the agent. The role of the model network is to take as input the previous observation-action history of the agent and predict the future state of the world. The M network is trained in a supervised fashion with the data that is collected by the agent while following the policy π generated by the Q network. If the M network predicts the state of the world correctly then the Q network is rewarded +1 else if the model network predicts the state of the world incorrectly then the Q network is punished or rewarded 0. Finally, the input to the model network is guided by a policy that is greedy with respect to the Q function that is the output of the Q network.

In other words, the Q network is rewarded for learning a Q function that promotes actions that helps the model network to learn a model of the world that can from partial observations predict the true state of the world. Figure 7.1 illustrates an abstract DAN. At each time step, the history of observations and actions are fed to the Q network, the Q network in turn outputs the Q values of each action and the agent takes the action with the highest Q value and gets an observation. This action and observation along with the history of the previous actions and observations are fed into the M network that predicts the state of the world (or it can be any other variable) and if the prediction made by the M network is correct then the Q network receives a reward of +1 or else the Q network gets a negative (-1) or 0 reward. As in active perception tasks the aim of the agent is to reduce the uncertainty in its belief as an end goal, this process goes on forever.

Note that this formulation is closely related to POMDP-IR. Here the M network can be seen as the network that models the prediction actions in POMDP-IR. The M network in the above described DAN can be augmented with multiple prediction actions that give the agent a choice to commit or to not commit fully to a particular state and instead

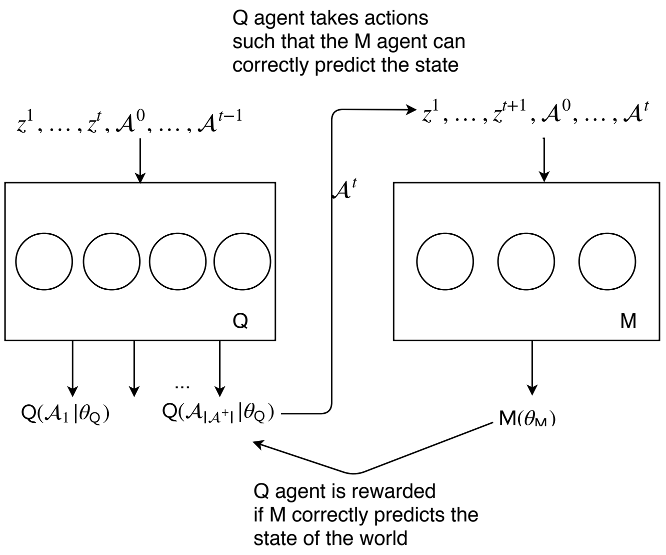

Q agent takes actions
such that the M agent can
correctly predict the state
$z^1, \dots, z^t, \mathcal{A}^0, \dots, \mathcal{A}^{t-1}$
$z^1, \dots, z^{t+1}, \mathcal{A}^0, \dots, \mathcal{A}^t$
$\mathcal{A}^t$
Q
M
...
$Q(\mathcal{A}_1 | \theta_Q)$
$Q(\mathcal{A}_{|\mathcal{A}^+|} | \theta_Q)$
$M(\theta_M)$
Q agent is rewarded
if M correctly predicts the
state of the world

Figure 7.1: An abstract model of DAN that consists of a Q network (or agent) and a M network (or agent). The Q network controls the input to the M network and the M network controls the reward Q network gets.

say ‘I do not know’ when it is not confident of its prediction. The Q network in that case can be rewarded a small positive reward. The Q network is the part of POMDP-IR that enables the agent to identify the normal sensory actions that best help the agent to minimize the uncertainty in the hidden belief of the M network. As shown in Chapter 3, such a POMDP-IR agent, in principal, approximately maximizes the negative conditional entropy. While we do not provide a formal proof that such a result also holds for DANs, this is the main intuition behind DANs and the claim that they are designed to maximize the information gain of a hidden abstract belief of the agent.

7.4 Training DAN

DAN is trained in a similar fashion to generative adversarial networks (GANs) as in both the Q and the M networks are trained simultaneously on small mini batches of data. Since one of the component in DAN is a DQN we additionally borrow the techniques used to train DQN to train DAN. Specifically, each belief-action pair that the agent encounters is stored in an experience buffer to be sampled later to train both the Q and the M network. Moreover, we also maintain two separate target networks for Q and M networks to get stable target values when updating the Q network. The exact algorithm is shown in Algorithm 10. Let θ_{Q} and θ_{M} denote the parameters of the Q network and the M network respectively.

Algorithm 10 `Train-DAN`

```
1:  Initialize $\theta_{\mathsf{Q}}$ and $\theta_{\mathsf{M}}$ randomly.
2:  Initialize experience replay.
3:  for $m = 1 \to \infty$ do
4:      for $t = 1$ to length of episode do
5:          Execute $\mathcal{A}^t = \max_{\mathcal{A} \in \mathcal{A}^+} Q(b^t, \mathcal{A}|\theta_{\mathsf{Q}})$ or take a random action with probability $\epsilon$.
6:          Observe $\mathbf{z}^{t+1}$ and update $b^{t+1}$
7:          Observe reward $r^{t+1} = f(target\mathsf{M}(b^{t+1}|\theta_{target\mathsf{M}}), s^{t+1})$.
8:          Add the experience tuple $\langle b^t, \mathcal{A}^t, r^{t+1}, b^{t+1}, s^{t+1} \rangle$ to the experience replay.
9:          Sample random mini-batch of data from the experience tuple.
10:         Update $\theta_{\mathsf{Q}}$ according to equations 7.2 and 7.3.
11:         Update $\theta_{\mathsf{M}}$ according to equations 7.4 and 7.6
12:     end for
13:     if update target = True then: // update target can be set to be True every few iterations
14:         update target Q parameters with $\theta_{\mathsf{Q}}$
15:         update target M parameters with $\theta_{\mathsf{M}}$
16:     end if
17: end for
```

For the purpose of training we assume access to a dataset with annotated true position of the person walking in the scene. In each iteration of the algorithm, for each episode, the agent follows the policy that is greedy with respect to the Q values that the Q network outputs. This experience accumulated by the agent is added to the experience buffer in the form of the tuple $\langle b^t, \mathcal{A}^t, r^{t+1}, b^{t+1}, s^{t+1} \rangle$ that is later used to train the Q network. The observations $\mathbf{z}^{t+1}$and the true state s^{t+1} are obtained from the dataset while the reward r^{t+1} is obtained from the target M network. r^{t+1} is defined as 1 if the target M network correctly predicts the true state else r^{t+1} is zero. At each time step, the agent samples random experience tuples from the experience buffer and updates θ_{Q} according to the Equations 7.2 and 7.3. Note that the target value in Equation 7.2 is obtained from the target M and target Q network and not the main Q and M network. Every few iterations the target Q and the target M network are updated with θ_{Q} and θ_{M} as parameters.

Once θ_{Q} is updated, θ_{M} is updated as well by performing a gradient descent update with cross entropy loss:

$$L = \texttt{cross-entropy}(\mathsf{M}(b^t|\theta_{\mathsf{M}}), s^{t+1}), \tag{7.4}$$

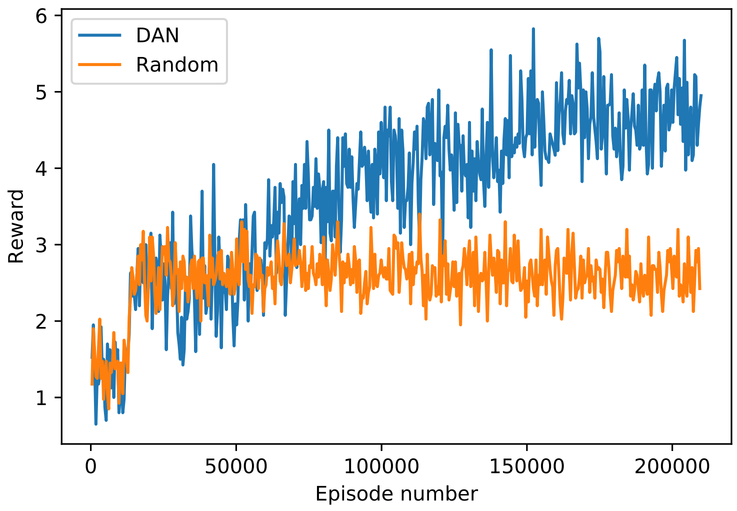


Figure 7.2: Performance comparison between randomly collecting partial observations to train the M network to predict the state of the world vs using a policy obtained by updating the Q network in the DAN architecture. The y-axis shows the mean reward over the last 440 episodes.

ideas from training DQN, DRQN and GANs so that the two separate networks in the DAN agent are trained separately but simultaneously. Experiments on a simulated setting demonstrate that its possible to train DAN such that the Q networks takes actions so that the M network is able to better predict the state of the world.

The results presented in this chapter are on a small simulated setting. While these results still demonstrate the potential possibilities of applying DAN to many tasks, they do not say anything about how useful DANs are when applied in real-life settings. Thus, the first and a natural task for future work is to train DAN on a real-life setting, for example, on the sensor selection task considered in other chapters of this thesis. Another enhancement that is quite clear from the DAN architecture described in the experiments is to share some of the representations and updates between the Q and M network. Since both Q and the M network almost share the same inputs, it is possible for them to share the same representations by sharing a big portion of the parameters θ_{Q} and θ_{M}. It is also possible to train the DAN architecture with shared parameters with shared updates to the parameters.

While it might be possible to train the DAN architecture by training the Q and M network together with a shared objective that can be expressed in terms of a loss function that is a weighted sum of individual loss function of the Q and the M network, we believe that the separation between the Q and M network leads to a better modularity. For example, the M network can well be a human that provides the reward to an agent as in the case of a human operator monitoring multiple feeds from a multi-camera network and the Q network can then be trained accordingly. The Q network in DAN can also be easily replaced with any other deep reinforcement learning methods such as actor-critic [Mnih et al., 2016] or policy gradients [Sutton et al., 2000, Silver et al., 2014, Ciosek and Whiteson, 2017]. Moreover, the DAN architecture introduces a general notion of co-operation between two agents to train themselves to achieve their own respective tasks that can be potentially applied in other settings such as active learning [Gal et al., 2017], meta-learning [Andrychowicz et al., 2016] or multi-agent settings [Foerster et al., 2017].

Finally, as mentioned in earlier chapters, one of the key challenges for active perception is large action spaces. While this chapter does not deal with large action space specifically, augmenting DANs to deal with it would be an important and may be an essential addition for practical problems. Dealing with large discrete action spaces remains a challenge in reinforcement learning and combining DANs with approximate maximization techniques such as greedy and stochastic greedy maximization is an interesting and non-trivial challenge.

8 Related Work

In this chapter we contrast the contributions of this thesis with the existing literature. We start with a broad overview of the existing methods for active perception and sensor selection. Next, we describe the existing POMDP planning methods and their limitations when applied to active perception POMDPs. This is followed by a brief account of the existing methods for submodular function maximization in the offline and online setting. Finally, we contrast our approach from recently developed deep visual attention models and deep reinforcement learning methods.

8.1 Active Perception and Sensor Selection

Active perception is a broad concept and encompasses a variety of problems, example, sensor alignment [Allen and Bajcsy, 1985], temporal or scene selection [Kelly, 1971, Ye and Tsotsos, 1995], active localization [Burgard et al., 1997], viewpoint selection [Wilkes and Tsotsos, 1992, Bruce and Tsotsos, 2009], control of memory [Oh et al., 2016] and visual attention [Borji and Itti, 2013, Mnih et al., 2014]. Bajcsy et al. [2016] provide an overview of the recent successes of active preception in robotics and computer vision. The common theme to many of the active perception problems is the efficient allocation of scarce resources to minimize uncertainty over a hidden variable. However, the application of submodular function maximization for active perception, to our knowledge, is limited. Hollinger et al. [2012] and Hollinger and Sukhatme [2013] exploit submodularity to propose policies for automatic underwater navigation for collecting oceanographic data and robot motion respectively. Taniguchi et al. [2015] perform greedy maximization of information gain for object recognition. Similarly, Le et al. [2008] employ greedy viewpoint selection for object detection for robots via interactive manipulation. However, a POMDP formulation that can reason about the long-term consequences, as we propose in Chapter 4, is missing from these approaches. The application of submodularity for long term POMDP planning is limited to our knowledge and we believe we are the first to combine long-term POMDP planning with submodularity for active perception.

This thesis focusses on sensor selection as an active perception task for tracking people or maintaining surveillance in large public spaces. Sensor selection has been studied in multiple contexts [Hero and Cochran, 2011, Monari and Kroschel, 2010, Tessens et al., 2014]. Earlier work focusses on either small simulated problems or myopic solutions, e.g., [Kreucher et al., 2005, Williams et al., 2007, Spaan and Lima, 2009, Joshi and Boyd, 2009, Monari and Kroschel, 2010]. In general, application of POMDP planning methods for the sensor selection task, especially for large real world problems is quite limited in the current literature. Kreucher et al. [2005] propose an *adaptive sampling technique* based on particle filters for sensor management. Williams et al. [2007] present an approach based on approximate dynamic programming for computing a policy for sensor scheduling. However, these approaches are limited to small simulated settings and do not attempt to learn the parameters of the model they propose or scale their methods to real world problems. The same is true for Joshi and Boyd [2009] that present a heuristic based on convex optimization for sensor selection to estimate a hidden variable that is a linear combination of the measurements received from the selected sensors. We make no such assumption. Other approaches based on information-theoretic heuristics [Gupta et al., 2007] also do not exploit the POMDP framework or submodularity as we do to scale greedy PBVI in the action space.

In contrast to these approaches, a POMDP-based approach leads to an informed policy as the agent can learn the model or policy from historical data and use the POMDP model to compute a non-myopic policy for the sensor selection. While a non-myopic approach might give only a marginal benefit for sensor selection; however, as the experiments in Chapter 4 showed that in a budget-based setting or in settings involving mobile robots, non-myopic planning is useful. Ji et al. [2007], Spaan [2008], Spaan and Lima [2009], Spaan et al. [2015] and Natarajan et al. [2012] consider a POMDP-based approach to active and cooperative active perception. However, they consider an objective function that conditions on the state and not on the belief, as belief-dependent rewards in POMDPs break the PWLC property of the value function. They use point-based methods [Spaan and Vlassis, 2005] for solving POMDPs. While recent point-based methods [Shani et al., 2012] for solving POMDPs scale reasonably in the state space of POMDPs, they do not address the scalability in the action and observation space of a POMDP as the methods described in this thesis do. Similarly, Krishnamurthy and Djonin [2007] consider a POMDP based approach for sensor selection where the reward is not linear in the belief, however, they do not address the issue of scalability in the action space of the POMDP they propose. Kumar and Zilberstein [2009] focus on a noiseless setting and a ‘detection’ based reward for sensor selection while we model the observation noise in sensors and reward the agent for collecting more information.

Traditionally, formulating the reward function in a POMDP as a function of belief was not possible. ρPOMDP and POMDP-IR are two frameworks that allow a belief-based formulation of the reward function in a POMDP. In Chapter 3, we establish the equivalence of these two frameworks and give a detailed empirical analysis of belief-based rewards in POMDPs which was missing until now in the literature. Previous work on belief-based rewards such as Eck and Soh [2012] provide an intuitive/empirical comparison between belief-based rewards and state based rewards for a simulated task of classifying mines. Araya-López et al. [2012] propose the use of belief-based rewards in POMDPs for active learning. However, none of this work provides the insights that we provide in Chapter 3, that ρPOMDP and POMDP-IR are equivalent frameworks and that the belief-based rewards do not cause any additional computational cost when compared to the same size POMDP with state-based rewards.

8.2 POMDP Planning

Exact POMDP planners such as proposed in Sondik [1971], Monahan [1982], Kaelbling et al. [1998] are limited to small POMDPs because of their computational cost. Point-based methods [Spaan and Vlassis, 2005, Pineau et al., 2006, Kurniawati et al., 2008, 2011, Shani et al., 2012] however have made it possible to solve POMDPs with large state spaces. The main idea here is to sample a subset of the possible beliefs and compute the optimal value function only for the beliefs that are in sampled subset of beliefs. However, existing point-based methods do not address the scalability in the action space of the POMDPs. By exploiting greedy maximization, greedy PBVI scales better in the action space of the active perception POMDP. Other POMDP planning methods such as those based on heuristic search [Smith and Simmons, 2004] do not necessarily target scalability in the action or observation space, instead they may involve an argmax over the action space to find the action that maximizes the upper/lower bound on the value function, which in our case is not possible. In general, point-based methods [Spaan and Vlassis, 2005, Shani et al., 2007, Poupart et al., 2011] differ in their belief collection method but perform an argmax over the entire action space which is not possible for the active perception POMDP.

Monte Carlo methods for offline planning such as Monte Carlo POMDPs [Thrun, 2000] and Monte Carlo value iteration [Bai et al., 2010] can handle large state and observation spaces in principle, maybe even large action spaces. However, their application on real-life problems with large POMDPs is still limited. Moreover, these methods are designed to address the state-based rewards. Their extension to belief-based rewards can be tricky especially for information-theoretic definitions since estimating information-theoretic rewards is not a straightforward task as shown in Chapter 5.

Online planning methods [Ross et al., 2008] like bandit-based Monte Carlo planning [Kocsis and Szepesvári, 2006, Silver and Veness, 2010] can also potentially handle large size POMDPs. However, existing methods do not explicitly address the combinatorial action space of the active perception POMDP. Our methods such as PartiMax and PAC greedy maximization can be easily integrated with Monte-Carlo planning to yield an even more efficient online planner.

The application of submodularity is gaining popularity in recent years for MDP planning in tasks other than sensor selection. Recently, David et al. [2016] and Kumar et al. [2017] proposed algorithms that exploit greedy maximization for MDP planning. David et al. [2016] propose greedy value iteration for one-shot recommendation systems. Kumar et al. [2017] use greedy maximization for decentralized planning for multi-agent systems with submodular rewards.

8.3 Submodular Function Maximization

Krause and Golovin [2014] provide a detailed survey on submodular function maximization. As submodularity is a natural and frequently occuring property, it applications are ubiquotous, for example, sensor deployment [Krause and Guestrin, 2005b], text summarization [Takamura and Okumura, 2009, Li et al., 2012], active learning [Chen and Krause, 2013], etc. Most work on submodular function maximization focuses on algorithms for approximate greedy maximization that minimize the number of evaluations of the submodular function Q [Minoux, 1978, Badanidiyuru and Vondrák, 2014, Mirzasoleiman et al., 2015]. In particular, Mirzasoleiman et al. [2015] sample a random subset of $\mathcal{X}$ on each iteration and selects the element from this subset that maximizes the marginal gain. Badanidiyuru and Vondrák [2014] select an element on each iteration whose marginal gain exceeds a certain threshold. Other proposed methods that maximize surrogate submodular functions [Wei et al., 2014, Chen and Krause, 2013] or address streaming [Krause and Gomes, 2010] or distributed settings [Mirzasoleiman et al., 2013], also assume access to the exact Q. Most of these works can be combined with greedy PBVI to extend them to long-term sequential decision making.

PAC greedy maximization, proposed in Chapter 5, is different from these methods as it does not assume that the submodular function Q can be computed exactly which is a prerequisite for most of the above mentioned methods. Streeter and Golovin [2009], Radlinski et al. [2008] and Yue and Guestrin [2011] propose conceptually related methods that also assume Q is never computed exactly. However, their *online* setting is fundamentally different in that the system must first select an entire subset $\mathcal{A} \in \mathcal{A}^+$ and only then receives an estimate of $Q(\mathcal{A})$, as well as estimates of the marginal gain of the elements in $\mathcal{A}$. Since the system learns over time how to maximize Q, it is a variation on the multi-armed bandit setting. By contrast, in Chapter 5, we assume that feedback about a given element's marginal gain is available (through tightening U and L) *before* committing to that element. Singla et al. [2016] also propose PAC maximization of submodular function. They propose an efficient algorithm that in each iteration of greedy maximization adds to the partial solution an element that maximizes the marginal gain with high probability. However, they assume access to an unbiased estimator of the function Q to get the upper and lower confidence bounds, we instead, propose new upper and lower confidence bounds, since in our setting such an estimator is not available.

PAC greedy maximization is closely related to *best arm identification* algorithms [Audibert and Bubeck, 2010]. However, such methods assume an unbiased estimator of Q is available and hence concentration inequalities like Hoeffding's inequality [Hoeffding, 1963] are applicable. An exception is the work of Loh and Nowozin [2013], which bounds the difference between an entropy estimate and that estimates expected value. However, since the entropy estimator is biased, this does not yield confidence bounds with respect to the true entropy. While they propose using their bounds for best arm identification, no guarantees are provided, and would be hard to obtain since the bias in estimating entropy has not been addressed. However, their bounds [Loh and Nowozin, 2013, Corollary 2] could be used in place of Theorem 13. While other work proposes more accurate estimators for entropy [Paninski, 2003, Schürmann, 2004, Ho et al., 2010, Nowozin, 2012a], they are not necessarily computationally efficient and thus not directly useful for PAC greedy maximization.

Finally, Conforti and Cornuéjols [1984] and Sharma et al. [2015] present alternate tight theoretical bounds on submodular maximization of entropy that explains the good performance of greedy maximization for maximizing information-theoretic quantities. While we have relied throughout on the popular result of Nemhauser et al. [1978], these bounds can be combined with the ideas that we presented to propose even tighter theoretical bounds for our methods. Fisher et al. [1978] extends the results of Nemhauser et al. [1978] to combinatorial structures such as matroids. When the POMDP action space is modelled as matroids greedy PBVI can be altered accordingly to extend these bounds to POMDP setting. We present one such case when we model the action space of the POMDP as a partition matroid in Chapter 4, section 4.2.

Adaptive submodularity [Golovin and Krause, 2011] is a recently developed extension that allows action selection to condition on previous observations [Chen et al., 2017]. However, it assumes a static state and thus cannot model the dynamics of a POMDP across time steps. Therefore, for sensor selection, adaptive submodularity is only applicable *within* a time step, during which state does not change but the agent can sequentially add sensors to a set. In principle, adaptive submodularity could enable this intra-time step sequential process to be adaptive, i.e., the choice of later sensors could condition on the observations generated by earlier sensors. However, this is not possible in our setting because (a) we assume that, due to computational costs, all sensors must be selected simultaneously; (b) information gain is not known to be adaptive submodular [Chen et al., 2015]. Consequently, our analysis for greedy PBVI, PAC greedy maximization and PartiMax considers only classic, non-adaptive submodularity.

Finally, greedy maximization is known to be *robust to noise* [Streeter and Golovin, 2009, Krause and Golovin, 2014]: if instead of selecting $i^G = \arg\max_{i \in \mathcal{X} \setminus \mathcal{A}^G} \Delta(i|\mathcal{A}^G)$, we selects i' such that $\Delta(i'|\mathcal{A}^G) \geq \Delta(i^G|\mathcal{A}^G) - \epsilon_1$, the total error is bounded by $\epsilon = k\epsilon_1$. We exploit this property in Chapter 5 but use confidence bounds to introduce a probabilistic element, such that with high probability $\Delta(i^P|\mathcal{A}^G) \geq \Delta(i^G|\mathcal{A}^G) - \epsilon_1$.

8.4 Visual Attention and Tracking

In general, most detection and tracking systems, e.g., Dalal and Triggs [2005], Felzenszwalb et al. [2010], Dollár et al. [2014], Benenson et al. [2014], including those based on convolutional neural networks [Girshick et al., 2014, Sermanet et al., 2014, Tian et al., 2015, Redmon et al., 2016], work by processing the entire image. However, visual attention models [Tsotsos and Shubina, 2010, Kootstra, 2010, Mnih et al., 2014] that selectively process only a part of an image are quickly becoming a popular feature of perception modules. For detection systems, existing work on visual attention or active vision identifies relevant regions of interest in an image [Kim et al., 2012], e.g., by generating proposals [Hosang et al., 2015] or saliency points [Bruce and Tsotsos, 2009, Shtrom et al., 2013]. These methods, however, are based on the properties (or low-level features) of the entire images. In contrast to these methods, PartiMax, proposed for tracking in Chapter 6, exploits the information in the beliefs from previous time steps to select sensors or pixel boxes for tracking people. Dellaert and Collins [1999] enable fast tracking algorithms by selective pixel integration that is similar to PartiMax. However, they go over the all possible subsets of pixels to maximize an information theoretic definition of the utility function. PartiMax avoids going over all possible pixel boxes with a smart sampling algorithm that exploits the coverage based definition of particle coverage.

Recently developed neural models of visual attention [Mnih et al., 2014, Denil et al., 2012] come close to the application we use PartiMax for. They are based on model-free deep reinforcement learning methods to identify relevant region to apply a trained detector on [Mnih et al., 2014]. In contrast to it PartiMax uses a learned model of the world to plan online to find the relevant regions to which to apply a trained detector.

Our work in Chapter 6 exploits the vast existing sensor selection literature for visual attention. Most work on sensor selection uses utility functions involving information gain [Wang et al., 2005, Roy and Earnest, 2006] and expected coverage [Spaan, 2008], which are too expensive for real-time systems. For real-time tracking we propose particle coverage that is simple to integrate with particle filters and suited for real-time applications because of its low computational cost. Moreover, we propose PartiMax that is able to very quickly maximize particle coverage resulting in an extremely efficient algorithm for visual attention for tracking.

8.5 Deep Reinforcement Learning

Recently, many methods based on deep reinforcement learning such as DQN, asynchronous actor critic (A3C) [Mnih et al., 2016] have been proposed that enable an agent to learn a control policy in an end-to-end manner from raw sensor data that maximizes the expected cumulative reward in an MDP. Most of the literature that exists on deep reinforcement learning assumes a state based reward that the agent receives at the end of the episode signalling whether the agent successfully completed the task or not. Moreover, most of these approaches consider an MDP formulation where the underlying state is fully observable (except DRQN [Hausknecht and Stone, 2015] and neural attention models [Mnih et al., 2014]). We, on the other hand, formulate the active perception task as reducing uncertainty as the end goal in a continuously evolving environment under partial observability. This is the main difference between our motivation and the existing work. DAN tackles the challenge associated with this formulation by separating the sensing actions and prediction actions giving the agent the choice to identify deep representations that are useful for predicting the hidden state even when only a little part of it is observable.

The closest work to DANs is the use of prediction-based rewards for intrinsic curiosity [Pathak et al., 2017] and as auxiliary variables [Jaderberg et al., 2016]. Again, the main difference here is that they use the prediction reward as a mean to train an agent to learn to play *Mario* and *vizdoom*. The performance of the policy is evaluated on an *extrinsic* state based 'end' reward instead of evaluating it in terms of number of correct predictions. A key difference in terms of the architecture and training algorithm for DANs and aforementioned approaches is that they train the agent as a whole entity with the resulting loss function being the weighted sum of multiple loss functions. DANs on the other hand 'factorize' this loss function into separate loss functions for the M network and Q network. This factorization provides better *modularity* when it comes to separating the representations that are useful to only Q or the M network and sharing representations that are useful to both.

We apply DANs for sensor selection task to track people. This application is closely related to attention models. Mnih et al. [2014] model the loss function as one conditioned on an 'end' state based reward where the agent after selecting a number of glimpses of an image of a digit must classify it correctly. Apart from this, there exists multiple subtle but key differences between sensor selection and existing literature on neural attention models. The current literature on attention is divided into models based on *soft* [Mnih et al., 2014] and *hard* [Xu et al., 2015] attention. Soft attention models use a differentiable function such as the sigmoid function to decide the probabilities over the region that an agent should attend and these probabilities vary smoothly over the image, where as hard attention models use non-differentiable switch like functions that change abruptly over the image to decide what part of an image the agent attends ignoring completely the rest. Sensor selection presents a 'harder-than-hard' attention case since the agent can only attend the hidden scene with glimpses that offer fixed points of attention and that cannot be resized. In contrast to this, for both soft attention and hard attention models the agent can choose the points where to attend in an image and it can also adjust the size and shape of the glimpse. In certain cases, the choice of the design of the attention glimpse may lie with the user of the system, even in this cases, the glimpses can be designed in favour of a tractable solution. However, for sensor selection the agent must select from the fixed point of views that are available through the already deployed set of sensors. This absence of the choice to select the point of attention or resize the glimpses available to the agent makes it harder for the agent to predict the state of the world from the partial observations. For certain multi-camera systems that involve pan-tilt-zoom cameras it might be possible for the agent to resize the glimpses or change the field of view of the cameras, however, the problem of selecting best operations out of pan, tilt and zoom poses its own unique challenges that we do not focus on in this thesis.

Ondruska and Posner [2016] present an approach for tracking people with deep neural networks, however, they do not employ an attention mechanism. Denil et al. [2012] try to learn where to look in an image to track people, however, they follow a bandit based approach instead of learning deep representations of the value function.

Many approaches [Le et al., 2008, Kostrikov et al., 2016] model active perception tasks with state based rewards that leads to an exploratory behaviour of the agent such that it is able to take actions that help the agent to classify an object or digit in a hidden image. These are again fundamentally different from our formulation.

DANs are modelled on a similar concept as GANs where two different networks are trained on each other's feedback. However, the GANs assume an adversarial relationship between the two networks leading to a min-max formulation of the final objective, while DANs lead to max-max formulation of the final objective.

Finally, DANs are related to learning in POMDPs/MDPs [James and Singh, 2009, Katt et al., 2017], however, DAN are designed to learn hidden representations of the world whereas previous approaches aim to learn the transition or observation function after assuming/designing the representation of the world.

9 Discussion and Future Work

9.1 Discussion

In this thesis we tackle the challenge of active perception for person tracking. We borrow from, and introduce new approaches in the sub fields of uncertainty approximation, POMDP planning, submodular function maximization, visual attention with particle filters and model-free deep reinforcement learning. The common aim of the methods we propose is active perception for tracking people. In summary, we present multiple methods that are tied, but not limited, by their application to active perception for tracking people in multi-camera networks. Here we discuss the limitations and usefulness of each method we propose. Table 9.1 provides a summary of the main strength and limitations of the methods proposed in this thesis.

Greedy PBVI is a principled and theoretically sound approach that is most suitable when the agent must compute the policy before executing the task and must plan long-term. Greedy PBVI scales in the action space of a POMDP, however, large observation spaces can prove to be a challenge for greedy PBVI. On the bright side for active perception POMDPs greedy PBVI performs almost optimally in practice (even if submodularity is violated in practice).

PAC greedy maximization can handle large state, action and observation spaces while working in real-time without loosing tracking performance. Ideally, PAC greedy maximization is most suited when there is little structure in the problem to exploit, that is, the multiple sensors that the agent has to choose from are all very distinct and may lead to multiple distinct possibilities in the future. For example, if each sensor that the agent has the choice of selecting from, covers a distinct region and has a distinct noise model then it is important to reason about all possibilities by simulating them. PAC greedy maximization with its pruning scheme can quickly eliminate sub-optimal choices and hence is suited for such scenes.

PartiMax exploits the structure in the tracking domain to boost the tracking performance and save the computation time. By dividing a single image into pixel boxes of equal sizes and similar noise models, PartiMax can afford to not reason about the noise models of these pixel boxes and still perform spectacularly even in the presence of noise. In fact, of all the methods we propose PartiMax results in lowest computation time with good tracking performance because it exploits the structure in the problem smartly. While PartiMax as an algorithm is customised to tracking people with particle filters, we give general theoretical guarantees for combinatorial optimization that are uniquely independent of the problem size.

Finally, if it is not possible to manually model the world and learn this model, DANs offer a model-free option that is directly deployable given enough computational resources and some ground truth data. DAN works on the principal of model-free deep reinforcement learning methods and thus does not require complex model to learn a policy that can select sensors to track people. While we show experimental results on DAN in a small simulated setting, given the current progress in the field of deep reinforcement learning, it will not be surprising if DANs performs well on real-life problems.

9.2 Future Work

Since this thesis ties together r methods from multiple sub fields of machine learning there are multiple directions for future research.

Entropy representation and approximation is a fundamental research topic in decision-making and probabilistic inference. It is also a common theme of this thesis that is encountered in almost every chapter. Building upon these methods for better entropy approximation is an exciting line of research for the future. For example, estimating entropy from samples is known to be a challenging problem [Paninski, 2003]. However, there are not many approaches in literature that deal with entropy estimation specifically for decision making. Specifically, estimating the difference in the entropy of two probabilities distribution as compared to estimating the entropies of each one of them individually is a different problem that has numerous applications.

Submodular function maximization is a popular topic in machine learning and has multiple applications [Krause and Golovin, 2014]. Information gathering and submodular function maximization tie together invariably. Recently many algorithms have been proposed for efficient submodular function maximization under various settings [Mirzasoleiman et al., 2017, Stan et al., 2017]. The setting that is most relevant to active per-

Table 9.1: Strengths and limitations of the methods presented in this thesis.

Method and description	Main strength	Main limitation
Greedy PBVI (Ch. 4) is an offline POMDP planner.	Greedy PBVI scales better in the action space of a POMDP and performs well in practice.	The performance of greedy PBVI may depend on the submodularity of the value function of the POMDP.
PAC greedy maximization (Ch. 5) of submodular functions.	PAC GM scales to large submodular maximization problems by quickly eliminating sub-optimal choices from the many available choices.	PAC GM may require a fair amount of parameter tuning to find the right balance between the computational cost and accuracy of the solution.
PartiMax (Ch. 6) is a tracking system built on top of particle filters.	PartiMax exploits structure in the tracking domain to run as a resource efficient tracker. In principal, PartiMax is guaranteed to perform well for all tracking problems irrespective of their size.	Application of PartiMax to other domains may require a few modifications to the original algorithm.
DAN (Ch. 7): a deep neural network architecture for model free active perception.	DANs do not require an explicit representation or model of the world for computing the policy for sensor selection.	Performance of DAN on real-life dataset/systems is yet to be seen.

ception is sequential maximization of submodular functions or optimizing submodular functions that are drawn from some unknown distribution [Stan et al., 2017]. Recently, this problem was formulated as a case of two-stage submodular function maximization where the ground set of items is reduced to a smaller set given training examples of the submodular functions. The problem is finally defined as selecting the reduced set that maximizes the expected value of the submodular function. Sensor selection is a fantastic application of such a method where the submodular functions are drawn from a probability distribution that is parametrised by the belief of the agent. However, assembling a dataset of submodular functions with full feedback (that is the value of each possible subset of the ground set) is not an easy task and in many settings bandit feedback is readily available instead of a complete feedback. Thus, extending these methods to the bandit feedback in an online setting is an exciting idea. Such an approach may tie closely with learning submodular functions, which are provably difficult to learn in traditional settings [Balcan and Harvey, 2011, Yue and Guestrin, 2011]. By exploiting the additional information in the previous beliefs in a sequential setting it might be possible to relax this problem. Finally, application of submodular reward functions for MDP/POMDP planning [David et al., 2016], including decentralised planning for multi-agent settings [Kumar et al., 2017], is gaining attention in current literature and naturally provides an interesting direction for future research.

For model free active perception we introduce DANs. While the current chapter on DANs lacks a thorough empirical analysis, conducting experiments that investigate the expressive power of DANs is a natural next step. Positive results can open multiple opportunities for future research direction, for example, improving the DAN architecture or improved training algorithms for DAN. Combining model-based and model-free RL [Gelly and Silver, 2007, Farquhar et al., 2017], both of which we present in this thesis, for active perception or for solving POMDPs with large action space is another attractive avenue.

On a different note an interesting direction for future research is human-in-the-loop RL [Mandel et al., 2017]. DANs especially can be used to directly learn a good policy that can integrate human feedback to optimize the performance of the system. By imagining one of the networks in the DAN architecture to be a human it is possible to learn the optimal behaviour of the other network directly through its experience with the human. It is common practice for security guards to monitor security cameras to maintain surveillance in large public spaces. These are natural settings for application of our methods that can be augmented with methods from human-in-the-loop RL.

The methods presented in this thesis have been shown to scale to problems of large sizes, thus, a natural application for them is real-time strategy games where an agent must select one of the many possible actions in real-time to accomplish certain objectives [Ontañón, 2017]. Also, methods presented in this thesis are directly applicable to model and data subset selection [Kirchhoff and Bilmes, 2014]. Search engines can also benefit from the submodular/combinatorial approach of this thesis [David et al., 2016] apart from the usual applications of submodular functions such as sensor placement, text summarization, active learning, influence maximization for marketing, etc.

10 Conclusions

This thesis tackles the question of 'how can an agent controlling a multi-camera network allocate its own resources?'. We focus on the task of maintaining surveillance in large public spaces with multi-camera networks and model this problem as an active perception POMDP where the agent at each time step must select k out of the n available cameras/sensors to allocate scarce resources to reduce uncertainty over the state of the world. Formulating uncertainty reduction as an end in itself is a challenging task, as it breaks the PWLC property of the value function, which is imperative for solving POMDPs efficiently. ρPOMDP and POMDP-IR are two frameworks that allow formulating uncertainty reduction as an end in itself and do not break the PWLC property. We show that ρPOMDP and POMDP-IR are two equivalent frameworks and the results that apply to one framework are also applicable to the other.

To tackle the challenges of long-term planning, combinatorial action space and large state spaces we propose multiple methods. To scale POMDP planning to the combinatorial action space of the active perception POMDP, we propose greedy PBVI that uses greedy maximization instead of full maximization to scale in the action space of the active perception POMDP. By exploiting the theory of submodularity we show that greedy PBVI is guaranteed to have bounded error. We establish the sufficient conditions for the value function of an active perception POMDP to be submodular. To scale greedy maximization to large state spaces (in addition to combinatorial action spaces), we propose PAC greedy maximization for submodular function maximization. Instead of assuming oracle access to the submodular function, PAC greedy maximization uses upper and lower confidence bounds on the submodular function to maximize it. Since it is not straightforward to obtain computationally cheap confidence bounds on information gain we propose new cheap confidence bounds on information gain for PAC greedy maximization. To track people in ultra-high resolution images we propose a new tracking system PartiMax that exploits the information in the previous beliefs of an agent to quickly identify the most relevant region in an ultra high resolution image to process to track people. Furthermore, we give error bounds for PartiMax that are crucially independent of the problem size. Finally, we present deep anticipatory networks that enable an agent to learn deep representations of the world to predict the future states from the partial observations of the world.

The empirical analysis in Chapter 3 establishes the critical factors for active perception tasks. We show that long term planning beats short term planning for tracking people, however, the gain was observed to be marginal, except for cases involving mobile robots or budgeted settings. Experiments on real-world dataset showed that greedy PBVI performs similar to existing methods but requires only a fraction of the computational cost, leading to much better scalability for solving active perception tasks. Similarly, experiments comparing PAC greedy maximization with greedy maximization show that as k increases PAC greedy maximization scales better than greedy maximization for maximizing a submodular function. We show that PartiMax retains 80% of the original tracking performance while processing only 10% of the image, thus, enabling tracking people in large ultra high resolution images. Finally, we show that DAN can potentially learn a good policy for active perception in a model free manner that removes the need to explicitly design and learn the model of the world. However, performance of DAN on real-world problems is yet to be seen.

In conclusion, this thesis presents multiple methods for active perception. The main premise of the thesis is the information gathering as an end task. Learning and applying strategies for information gathering, remembering important details, focussing and attending to details, etc. is an indispensable part of any definition of intelligence and this thesis pushes the state of the art for incorporating these features in an intelligent autonomous agent.

11 Appendix

11.1 Results from Chapter 3

11.1.1 Proofs from section 3.1

Theorem 3 *Let* $\mathrm{M}_\rho = \langle S, A_\rho, \Omega, T_\rho, O_\rho, \Gamma_\rho, b_0, h\rangle$ *be a ρPOMDP, and* π_ρ *an arbitrary policy for* M_ρ*. Furthermore let* M_{IR} = REDUCE-POMDP-ρ-IR(M_ρ) *and* π_{IR} = REDUCE-POLICY-ρ-IR(π_ρ)*. Then, for all b,*

$$V_t^{IR}(b) = V_t^\rho(b), \tag{11.1}$$

where

$$V_t^{IR} = \max_{a_p} \sum_s b(s) R_{IR}(s, a_p) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_{IR}^n(b)) V_{t-1}^{IR}(b_{\mathbf{z}}^{\pi_{IR}^n(b)}), \tag{11.2}$$

is the t-step value function of policy π_{IR} in POMDP-IR M_{IR} and

$$V_t^\rho = [\rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_\rho(b)) V_{t-1}^\rho(b_{\mathbf{z}}^{\pi_\rho(b)})], \tag{11.3}$$

is the t-step value function of policy π_ρ in ρPOMDP M_ρ.

Proof. By induction on t. To prove the base case, we observe that, from the definition of $\rho(b)$,

$$V_0^\rho(b) = \rho(b) = \max_{\alpha_\rho^{a_p} \in \Gamma_\rho} \sum_s b(s) \alpha_\rho^{a_p}(s). \tag{11.4}$$

Since $A_{p,IR}$ in M_{IR} has a prediction action corresponding to each $\alpha_\rho^{a_p}$, and π_{IR} is such that it takes the prediction action that maximizes the immediate belief based reward. Then,

$$V_0^{IR}(b) = \max_{a_p \in A_{p,IR}} \sum_s b(s) R(s, a_p) \tag{11.5}$$

(Since there is a prediction action corresponding to every alpha, this implies,)

$$= \max_{a_p \in A_{p,IR}} \sum_s b(s) \alpha_\rho^{a_p}(s) = \max_{\alpha_\rho^{a_p} \in \Gamma_\rho} \sum_s b(s) \alpha_\rho^{a_p}(s) \tag{11.6}$$

$$= V_0^\rho(b) \tag{11.7}$$

For the inductive step, we assume that $V_{t-1}^{IR}(b) = V_{t-1}^{\rho}(b)$ and must show that $V_t^{IR}(b) = V_t^{\rho}(b)$. Starting with $V_t^{IR}(b)$,

$$V_t^{IR}(b) = \max_{a_p} \sum_s b(s) R(s, a_p) + \sum_z \Pr(\mathbf{z}|b, \pi_{IR}^n(b)) V_{t-1}^{IR}(b_{\mathbf{z}}^{\pi_{IR}^n(b)}), \tag{11.8}$$

where $\pi_{IR}^n(b)$ denotes the normal action of the tuple specified by $\pi_{IR}(b)$ and:

$$\Pr(\mathbf{z}|b, \pi_{IR}^n(b)) = \sum_s \sum_{s''} O_{IR}(s'', \pi_{IR}^n(b), \mathbf{z}) T_{IR}(s, \pi_{IR}^n(b), s'') b(s). \tag{11.9}$$

Using the reduction procedure, we know, for all $s, s'', \mathbf{z}$,

$$\pi_\rho(b) = \pi_{IR}^n(b), \tag{11.10}$$

$$O_\rho(s'', \pi_\rho(b), \mathbf{z}) = O_{IR}(s'', \pi_{IR}^n(b), \mathbf{z}) \tag{11.11}$$

$$T_\rho(s, \pi_\rho(b), s'') = T_{IR}(s, \pi_{IR}^n(b), s'') \tag{11.12}$$

Substituting these in (11.9)

$$\Pr(\mathbf{z}|b, \pi_{IR}^n(b)) = \sum_s \sum_{s''} O_\rho(s'', \pi_\rho(b), z) T_\rho(s, \pi_\rho(b), s'') b(s) = \Pr(\mathbf{z}|b, \pi_\rho(b)). \tag{11.13}$$

Similarly, for the belief update equation,

$$\begin{aligned} b_{\mathbf{z}}^{\pi_{IR}^n(b)}(s') &= \frac{O_{IR}(s', \pi_{IR}^n(b), \mathbf{z})}{\Pr(\mathbf{z}|\pi_{IR}^n(b), b)} \sum_s b(s) T_{IR}(s, \pi_{IR}^n(b), s') \ \forall \ s' \\ &= \frac{O_\rho(s', \pi_\rho(b), \mathbf{z})}{\Pr(\mathbf{z}|\pi_\rho(b), b)} \sum_s b(s) T_\rho(s, \pi_\rho(b), s') \ \forall \ s' \\ &= b_{\mathbf{z}}^{\pi_\rho(b)}(s') \ \forall \ s'. \end{aligned} \tag{11.14}$$

Substituting the above result in (11.8) yields:

$$V_t^{IR}(b) = \max_{a_p} \sum_s b(s)R(s, a_p) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_\rho(b)) V_{t-1}^{IR}(b_{\mathbf{z}}^{\pi_\rho(b)}). \tag{11.15}$$

Since the inductive assumption tells us that $V_{t-1}^{IR}(b) = V_{t-1}^{\rho}(b)$ and (11.7) and (11.4) show that $\rho(b) = \max_{a_p} \sum_s b(s)R(s, a_p)$:

$$\begin{aligned} V_t^{IR}(b) =& [\rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_\rho(b)) V_{t-1}^{\rho}(b_{\mathbf{z}}^{\pi_\rho(b)})] \\ =& V_t^{\rho}(b). \end{aligned} \tag{11.16}$$

□

Theorem 4 *Let* $\mathbf{M}_{IR} = \langle S, A_{IR}, \Omega, T_{IR}, O_{IR}, R_{IR}, b_0, h \rangle$ *be a POMDP-IR and* $\pi_{IR}(b) = \langle \pi_{IR}^n(b), \pi_{IR}^p(b) \rangle = \langle \mathbf{a}_n, a_p \rangle$ *a policy for* $\mathbf{M}_{IR}$*, such that* $\pi_{IR}^p(b) = a_p = \arg\max_{a'_p} \sum_s b(s)R(s, a'_p)$*. Furthermore let* $\mathbf{M}_\rho$ = REDUCE-POMDP-IR-ρ($\mathbf{M}_{IR}$) *and* π_ρ = REDUCE-POLICY-IR-ρ(π_{IR})*. Then, for all b,*

$$V_t^{\rho}(b) = V_t^{IR}(b), \tag{11.17}$$

where

$$V_t^{IR}(b) = \max_{a_p} \sum_s b(s)R_{IR}(s, a_p) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_{IR}^n(b)) V_{t-1}^{IR}(b_{\mathbf{z}}^{\pi_{IR}^n(b)}), \tag{11.18}$$

is the value of following π_{IR} *in* $\mathbf{M}_{IR}$ *and*

$$V_t^{\rho}(b) = [\rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_\rho(b)) V_{t-1}^{\rho}(b_{\mathbf{z}}^{\pi_\rho(b)})], \tag{11.19}$$

is the value of following π_ρ *in* $\mathbf{M}_\rho$*.*

Proof. By induction on t. To prove the base case, we observe that, from the definition of $\rho(b)$,

$$\begin{aligned}
V_0^{IR}(b) &= \max_{a_p} \sum_s b(s)R(s,a_p) \\
&= \max_{\alpha_\rho^{a_p} \in \Gamma_\rho} \sum_s b(s)\alpha_\rho^{a_p}(s) \ \{\text{since } \Gamma_\rho \text{ is made up of } \alpha \text{ vectors} \\
&\text{such that } R(s,a_p) = \alpha_\rho^{a_p}(s) \text{ for all } a_p \in A_{p,IR}.\} \\
&= \rho(b) \\
&= V_0^\rho(b)
\end{aligned} \tag{11.20}$$

For the inductive step, we assume that $V_{t-1}^\rho(b) = V_{t-1}^{IR}(b)$ and must show that $V_t^\rho(b) = V_t^{IR}(b)$. Starting with $V_t^\rho(b)$,

$$V_t^\rho(b) = \rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi_\rho(b)) V_{t-1}^\rho(b_{\mathbf{z}}^{\pi_\rho(b)}), \tag{11.21}$$

and:

$$\Pr(\mathbf{z}|b, \pi_\rho(b)) = \sum_s \sum_{s''} O_\rho(s'', \pi_\rho(b), \mathbf{z}) T_\rho(s, \pi_\rho(b), s'') b(s). \tag{11.22}$$

Using the reduction procedure we know, for all $s, s'', \mathbf{z}$:

$$\pi_{IR}^n(b) = \pi_\rho(b), \tag{11.23}$$

$$O_{IR}(s'', \pi_{IR}^n(b), \mathbf{z}) = O_\rho(s'', \pi_\rho(b), \mathbf{z}) \tag{11.24}$$

$$T_{IR}(s, \pi_{IR}^n(b), s'') = T_\rho(s, \pi_\rho(b), s'') \tag{11.25}$$

Substituting these in (11.21)

$$\Pr(\mathbf{z}|b, \pi_\rho(b)) = \sum_s \sum_{s''} O_{IR}(s'', \pi_{IR}^n(b), \mathbf{z}) T_{IR}(s, \pi_{IR}^n(b), s'') b(s) = \Pr(\mathbf{z}|b, \pi_{IR}^n(b)). \tag{11.26}$$

Similarly, for the belief update equation,

$$\begin{aligned}
b_{\mathbf{z}}^{\pi_\rho(b)}(s') &= \frac{O_\rho(s', \pi_\rho(b), \mathbf{z})}{\Pr(\mathbf{z}|\pi_\rho(b), b)} \sum_s b(s) T_\rho(s, \pi_\rho(b), s') \ \forall \ s' \\
&= \frac{O_{IR}(s', \pi^n_{IR}(b), \mathbf{z})}{\Pr(\mathbf{z}|\pi^n_{IR}(b), b)} \sum_s b(s) T_{IR}(s, \pi^n_{IR}(b), s') \ \forall \ s' \\
&= b_{\mathbf{z}}^{\pi^n_{IR}(b)}(s') \ \forall \ s'.
\end{aligned} \tag{11.27}$$

Substituting the above result in (11.21) yields:

$$V_t^\rho(b) = \rho(b) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi^n_{IR}(b)) V^\rho_{t-1}(b_{\mathbf{z}}^{\pi^n_{IR}(b)}). \tag{11.28}$$

Since the inductive assumption tells us that $V^\rho_{t-1}(b) = V^{IR}_{t-1}(b)$ and (11.20) shows that $\max_{a_p} \sum_s b(s)R(s, a_p) = \rho(b)$:

$$\begin{aligned}
V_t^\rho(b) = &[\max_{a_p} \sum_s b(s)R(s, a_p) + \sum_{\mathbf{z}} \Pr(\mathbf{z}|b, \pi^n_{IR}(b)) V^{IR}_{t-1}(b_{\mathbf{z}}^{\pi^n_{IR}(b)})] \\
=&V_t^{IR}(b).
\end{aligned} \tag{11.29}$$

□

11.2 Results from Chapter 4

11.2.1 Proofs from subsection 4.1.2

Lemma 2 *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h \rangle$ be an active perception POMDP. Let π be a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \ldots, n\}$ denotes the set of n sensors. Let b^t and $\mathcal{A}^t$ denote the belief and action at time step t. If the immediate belief-based reward ρ in $\mathbf{M}$ is defined as the negative belief entropy, that is, $\rho(b) = -H_b(s)$ then the expected reward at time step $j(< t)$ as a consequence of taking action $\mathcal{A}^t$ in belief b^t and following policy π in between the time interval t to j equals the negative discounted conditional entropy of b^j over s^j given $\mathbf{z}^{t:j}$:*

$$G^\pi_j(b^t, \mathcal{A}^t) = \sum_{\mathbf{z}^{t:j}} \Pr(\mathbf{z}^{t:j}|b^t, \mathcal{A}^t, \pi)(-H_{b^j}(s^j)) = -H_{b^j}(s^j|\mathbf{z}^{t:j}). \tag{11.30}$$

Here b^j is the belief at time step j, s^j denote the hidden state at time step j and $\mathbf{z}^{t:j}$ denote the sequence of observations that can be obtained in between the time interval *from t to j time step.*

Proof. To prove the above lemma, we take help of some additional notations and definitions, first we must elaborate on the definition of b^j:

$$b^j(s^j) \triangleq \Pr(s^j|b^t, \mathcal{A}^t, \pi, \mathbf{z}^{t:j}) = \frac{\Pr(\mathbf{z}^{t:j}, s^j|b^t, \mathcal{A}^t, \pi)}{\Pr(\mathbf{z}^{t:j}|b^t, \mathcal{A}^t, \pi)}. \tag{11.31}$$

For notational convenience, we also write this as:

$$b^j(s^j) \triangleq \frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})}. \tag{11.32}$$

The entropy of b^j is thus (using the definition given by (2.16)):

$$H_{b^j}(s^j) = -\sum_{s^j} \frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})} \log(\frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})}), \tag{11.33}$$

and the conditional entropy of b^j over s^j given $\mathbf{z}^{t:j}$ is (using definition given by (4.33)):

$$H^{\mathcal{A}^t}_{b^j}(s^j|\mathbf{z}^{t:j}) = -\sum_{s^j}\sum_{\mathbf{z}^{t:j}} \Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j) \log(\frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})}). \tag{11.34}$$

Starting with the definition of $G^{\pi}_j(b^t, \mathcal{A}^t)$ (using (4.31)),

$$G^{\pi}_j(b^t, \mathcal{A}^t) = (-\sum_{\mathbf{z}^{t:j}} \Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}) H_{b^j}(s^j)) \tag{11.35}$$

By definition of entropy, as in (11.33) (11.36)

$$= \sum_{\mathbf{z}^{t:j}} \Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}) \left[\sum_{s^j} \frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})} \log(\frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})}) \right] \tag{11.37}$$

$$= \sum_{\mathbf{z}^{t:j}} \left[\sum_{s^j} \Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j) \log(\frac{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j}, s^j)}{\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})}) \right] \tag{11.38}$$

By definition of conditional entropy, as in (11.34) (11.39)

$$= (-H^{\mathcal{A}^t}_{b^j}(s^j|\mathbf{z}^{t:j})). \tag{11.40}$$

Eq. (11.38) follows from (11.37) because $\Pr^{\pi}_{b^t,\mathcal{A}^t}(\mathbf{z}^{t:j})$ is independent of s^j and can be moved out of the summation over s^j.

□

Lemma 3 *Let $\mathcal{Z} = \{z_1, z_2, \dots z_n\}$ be the set of all observation features about a hidden* *variable s. If $\mathcal{Z}$ is conditionally independent given s then $-H(s|\mathcal{Z})$ is submodular in $\mathcal{Z}$, i.e., for any two observations set $\mathcal{Z}_M, \mathcal{Z}_N \subseteq \mathcal{Z}$,*

$$H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) + H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) \geq H(s|\mathcal{Z}_M) + H(s|\mathcal{Z}_N). \tag{11.41}$$

Proof. By Bayes' rule for conditional entropy :

$$H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) = H(\mathcal{Z}_M \cup \mathcal{Z}_N|s) + H(s) - H(\mathcal{Z}_M \cup \mathcal{Z}_N). \tag{11.42}$$

Eq. (11.42) is true because:

$$\text{By chain rule of entropy [Cover and Thomas, 1991]:} \tag{11.43}$$
$$H(s, \{\mathcal{Z}_M \cup \mathcal{Z}_N\}) = H(s|\{\mathcal{Z}_M \cup \mathcal{Z}_N\}) + H(\{\mathcal{Z}_M \cup \mathcal{Z}_N\}) \tag{11.44}$$
$$= H(\{\mathcal{Z}_M \cup \mathcal{Z}_N\}|s) + H(s) \tag{11.45}$$

Using (11.44) and (11.45), (11.42) can be obtained. $\mathcal{Z}_M$ and $\mathcal{Z}_N$ are subsets of $\mathcal{Z}$, thus, $\mathcal{Z}_M \cup \mathcal{Z}_N$ can only contain observation features z_i that are in $\mathcal{Z}$. Thus, if $\mathcal{Z}$ is conditionally indepdent given s (which we have assumed), then $\mathcal{Z}_M \cup \mathcal{Z}_N$ is conditionally independent given s.

Since $\mathcal{Z}_M \cup \mathcal{Z}_N$ is conditionally independent given s and the entropy of two independent variables is just the sum of their individual entropies, we get $H(\mathcal{Z}_M \cup \mathcal{Z}_N|s) = H(\mathcal{Z}_M|s) + H(\mathcal{Z}_N|s)$ [1]

[1] A short and simple proof for this is:

$$H(\mathcal{Z}_M \cup \mathcal{Z}_N|s) = \sum_{z_i \in \mathcal{Z}_M, z_j \in \mathcal{Z}_N} \Pr(z_i, z_j|s) \log(\Pr(z_i, z_j|s)) \tag{11.46}$$
$$\text{Using conditional independence,} \tag{11.47}$$
$$= \sum_{z_i \in \mathcal{Z}_M, z_j \in \mathcal{Z}_N} \Pr(z_i, z_j|s) \log(\Pr(z_i|s)\Pr(z_j|s)) \tag{11.48}$$
$$= \sum_{z_i \in \mathcal{Z}_M, z_j \in \mathcal{Z}_N} \Pr(z_i|s)\Pr(z_j|s) \log(\Pr(z_i|s)) + \sum_{z_i \in \mathcal{Z}_M, z_j \in \mathcal{Z}_N} \Pr(z_i|s)\Pr(z_j|s) \log(\Pr(z_j|s)) \tag{11.49}$$
$$= \sum_{z_i \in \mathcal{Z}_M} \Pr(z_i, |s) 1 \log(\Pr(z_i|s)) + \sum_{z_j \in \mathcal{Z}_N} 1 \Pr(z_j|s) \log(\Pr(z_j|s)) \tag{11.50}$$
$$= H(\mathcal{Z}_M|s) + H(\mathcal{Z}_N|s). \tag{11.51}$$

Substituting this in (11.42), we get:

$$H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) = H(\mathcal{Z}_M|s) + H(\mathcal{Z}_N|s) + H(s) - H(\mathcal{Z}_M \cup \mathcal{Z}_N). \tag{11.52}$$

Using Bayes' rule for conditional entropy:

$$H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) = H(\mathcal{Z}_M \cap \mathcal{Z}_N|s) + H(s) - H(\mathcal{Z}_M \cap \mathcal{Z}_N). \tag{11.53}$$

Adding (11.52) and (11.53):

$$\begin{aligned} H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) + H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) = H(\mathcal{Z}_M|s) + H(\mathcal{Z}_N|s) \\ + H(\mathcal{Z}_M \cap \mathcal{Z}_N|s) + 2H(s) \\ - H(\mathcal{Z}_M \cup \mathcal{Z}_N) - H(\mathcal{Z}_M \cap \mathcal{Z}_N). \end{aligned} \tag{11.54}$$

Using Bayes' rule for conditional entropy:

$$\begin{aligned} H(\mathcal{Z}_M|s) = H(s|\mathcal{Z}_M) + H(\mathcal{Z}_M) - H(s), \text{and} \\ H(\mathcal{Z}_N|s) = H(s|\mathcal{Z}_N) + H(\mathcal{Z}_N) - H(s) \end{aligned} \tag{11.55}$$

Substituting $H(\mathcal{Z}_M|s)$ and $H(\mathcal{Z}_N|s)$ in (11.54):

$$\begin{aligned} H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) + H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) = H(s|\mathcal{Z}_M) + H(s|\mathcal{Z}_N) \\ + H(\mathcal{Z}_M \cap \mathcal{Z}_N|s) + [H(\mathcal{Z}_M) \\ + H(\mathcal{Z}_N) - H(\mathcal{Z}_M \cup \mathcal{Z}_N) - H(\mathcal{Z}_M \cap \mathcal{Z}_N)]. \end{aligned}$$

Since entropy is submodular $[H(\mathcal{Z}_M) + H(\mathcal{Z}_N) - H(\mathcal{Z}_M \cup \mathcal{Z}_N) - H(\mathcal{Z}_M \cap \mathcal{Z}_N)]$ is positive and since entropy is positive, $H(\mathcal{Z}_M \cap \mathcal{Z}_N|s)$ is positive [Fujishige, 1978]. Thus,

$$\begin{aligned} H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) + H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) = H(s|\mathcal{Z}_M) + H(s|\mathcal{Z}_N) \\ + \text{a positive term.} \end{aligned}$$

This implies $H(s|\mathcal{Z}_M \cup \mathcal{Z}_N) + H(s|\mathcal{Z}_M \cap \mathcal{Z}_N) \geq H(s|\mathcal{Z}_M) + H(s|\mathcal{Z}_N)$. $\square$

Lemma 4 *Let* $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ *be an active perception POMDP. Let* π *be a policy that maps beliefs to actions in* $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, *where* $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ *and* $\mathcal{X} = \{1, 2, \ldots, n\}$ *denotes the set of* n *sensors. Let* b^t *and* $\mathcal{A}^t$ *denote the belief and action at time step* t. *Let* $\mathcal{Z}^{t:j} = \{z_1^t, z_2^t, \ldots, z_n^t, z_1^{t-1}, z_2^{t-1}, \ldots, z_n^j\}$ *be the set of observation features that can be obtained between the time interval from* t *to* j. *If the* *immediate belief-based reward ρ in $\mathbf{M}$ is defined as the negative belief entropy, that is, $\rho(b) = -H_b(s)$, and if $\mathcal{Z}^{t:j}$ is conditionally independent given s^j, then $G^\pi_j(b^t, \mathcal{A}^t) = -H_{b^j}(s^j|\mathcal{Z}^{t:j})$ is submodular in $\mathcal{A}^t$ $\forall$ π. Here b^j is the belief at time step j, s^j denote the hidden state at time step j.*

Proof. Let $\mathcal{A}^t_M$ and $\mathcal{A}^t_N$ be two actions and $\mathcal{Z}^{t:j}_M$ and $\mathcal{Z}^{t:j}_N$ the observations they induce. Then, from Lemma 2,

$$G^\pi_j(b^t, \mathcal{A}^t_M) = (-H_{b^j}(s^j|\mathcal{Z}^{t:j}_M)). \tag{11.56}$$

From Lemma 3,
$H_{b^j}(s^j|\mathcal{Z}^{t:j}_M \cup \mathcal{Z}^{t:j}_N) + H_{b^j}(s^j|\mathcal{Z}^{t:j}_M \cap \mathcal{Z}^{t:j}_N) \geq H_{b^j}(s^j|\mathcal{Z}^{t:j}_M) + H_{b^j}(s^j|\mathcal{Z}^{t:j}_N)$
Multiplying by -1 on both sides and using definition of G
$G^\pi_j(b^t, \mathcal{A}^t_M \cup \mathcal{A}^t_N) + G^\pi_j(b^t, \mathcal{A}^t_N \cap \mathcal{A}^t_M) \leq G^\pi_j(b^t, \mathcal{A}^t_M) + G^\pi_j(b^t, \mathcal{A}^t_N)$.

□

Lemma 5 *Let $\mathbf{M} = \langle S, \mathcal{A}^+, \Omega, T, O, \rho, b_0, h\rangle$ be an active perception POMDP. Let π be a policy that maps beliefs to actions in $\mathcal{A}^+$, $\pi(b) = \mathcal{A}$, where $\mathcal{A}^+ = \{\mathcal{A} \subseteq \mathcal{X} : |\mathcal{A}| \leq k\}$ and $\mathcal{X} = \{1, 2, \ldots, n\}$ denotes the set of n sensors. Let $Q^\pi_t(b, \mathcal{A}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathcal{A})V^\pi_{t-1}(b^{\mathcal{A}}_{\mathbf{z}})$ be the t-step action value function of policy π. If $V^\pi_{t-1}(b')$ is a convex function of b', then $Q^\pi_t(b, \mathcal{A})$ is monotone in $\mathcal{A}$, i.e., for all b and $\mathcal{A}_M \subseteq \mathcal{A}_N$ ($\mathcal{A}_M, \mathcal{A}_N \in \mathcal{A}^+$), $Q^\pi_t(b, \mathcal{A}_M) \leq Q^\pi_t(b, \mathcal{A}_N)$.*

Proof. Since $\rho(b)$ is independent of $\mathcal{A}_M$, we need only to show that the second term is monotone in $\mathcal{A}$. Let $\mathcal{A}_P = \mathcal{A}_N \setminus \mathcal{A}_M$ and

$$F^\pi_b(\mathcal{A}_N) = \mathbb{E}_{\mathcal{Z}_N}[V^\pi_{t-1}(b^{\mathcal{A}_N}_{\mathcal{Z}_N})|b, \mathcal{A}_N]. \tag{11.57}$$

Since $\mathcal{A}_N = \{\mathcal{A}_M \cup \mathcal{A}_P\}$, we start with following set of inequalities and provide the explanation below:

$$
\begin{aligned}
F_b^\pi(\mathcal{A}_N) &= \mathbb{E}_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}[V_{t-1}^\pi(b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}})|b,\{\mathcal{A}_M,\mathcal{A}_P\}] && (11.58)\\
&= \sum_{\mathcal{Z}_M,\mathcal{Z}_P} \Pr(\mathcal{Z}_M,\mathcal{Z}_P|b,\mathcal{A}_M,\mathcal{A}_P)[V_{t-1}^\pi(b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}})] && (11.59)\\
&= \sum_{\mathcal{Z}_M}\sum_{\mathcal{Z}_P} \Pr(\mathcal{Z}_P|\mathcal{Z}_M,b,\mathcal{A}_M,\mathcal{A}_P)\Pr(\mathcal{Z}_M|b,\mathcal{A}_M)[V_{t-1}^\pi(b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}})] && (11.60)\\
&= \sum_{\mathcal{Z}_M} \Pr(\mathcal{Z}_M|b,\mathcal{A}_M)\sum_{\mathcal{Z}_P} \Pr(\mathcal{Z}_P|\mathcal{Z}_M,b,\mathcal{A}_M,\mathcal{A}_P)[V_{t-1}^\pi(b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}})] && (11.61)\\
&\geq \sum_{\mathcal{Z}_M} \Pr(\mathcal{Z}_M|b,\mathcal{A}_M)[V_{t-1}^\pi\Big(\sum_{\mathcal{Z}_P} \Pr(\mathcal{Z}_P|\mathcal{Z}_M,b,\mathcal{A}_M,\mathcal{A}_P)b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}}\Big)] && (11.62)\\
&= \sum_{\mathcal{Z}_M} \Pr(\mathcal{Z}_M|b,\mathcal{A}_M)[V_{t-1}^\pi(b_{\mathcal{Z}_M}^{\mathcal{A}_M})] && (11.63)\\
&= F_b^\pi(\mathcal{A}_M) && (11.64)
\end{aligned}
$$

Eq. (11.58) and (11.59) are simple re-writing of same terms. Eq. (11.60) follows from rule of conditional probability, that, $\Pr(A \cup B) = \Pr(A|B)\Pr(B)$, eq. (11.62) follows from pulling the $\Pr(\mathcal{Z}_M|b,\mathcal{A}_M)$ outside the summation over $\mathcal{Z}_P$ since $\Pr(\mathcal{Z}_M|b,\mathcal{A}_M)$ is independent of $\mathcal{Z}_P$. Eq. (11.60) follows from Jensen's inequality since V_{t-1}^π is convex in b (this is belief at time step $t-1$), that says that for a convex function $\psi(X)$ of X, $\mathbb{E}[\psi(X)] \geq \psi(\mathbb{E}(X))$. Eq. (11.62) follows from the observation that the expected posterior belief after observing $\mathcal{Z}^P$ is the same as the prior belief $b_{\mathcal{Z}_M}^{\mathcal{A}_M}$ [2].

Consequently, we have:

$$
\begin{aligned}
\rho(b) + F_b^\pi(\mathcal{A}_N) &\geq \rho(b) + F_b^\pi(\mathcal{A}_M)\\
Q_t^\pi(b,\mathcal{A}_N) &\geq Q_t^\pi(b,\mathcal{A}_M).
\end{aligned} \qquad (11.65)
$$

□

11.2.2 Proofs from section 4.2

Lemma 7 *Let* $\mathbf{M} = \langle S,\mathcal{M},\Omega,T,O,R,b_0,h\rangle$ *be an active perception POMDP, where* $\mathcal{M} = (V,I)$ *is a matroid. Let* π *be a policy that maps beliefs* b *to actions* $\mathfrak{a}$ *in* I,

[2] $b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}}(s')$ is $\Pr(s'|\mathcal{Z}_M,\mathcal{Z}_P,b,\mathcal{A}_M,\mathcal{A}_P)$. Thus, $\sum_{\mathcal{Z}_P}\Pr(\mathcal{Z}_P|\mathcal{Z}_M,b,\mathcal{A}_M,\mathcal{A}_N)b_{\{\mathcal{Z}_M,\mathcal{Z}_P\}}^{\{\mathcal{A}_M,\mathcal{A}_P\}}(s') = \sum_{\mathcal{Z}_P}\Pr(\mathcal{Z}_P|\mathcal{Z}_M,b,\mathcal{A}_M,\mathcal{A}_N)\Pr(s'|\mathcal{Z}_M,\mathcal{Z}_P,b,\mathcal{A}_M,\mathcal{A}_P) = \Pr(s'|\mathcal{Z}_M,b,\mathcal{A}_M) = b_{\mathcal{Z}_M}^{\mathcal{A}_M}(s')$.

$\pi(b) = \mathfrak{a}$. *Let* $Q_t^\pi(b,\mathfrak{a})$ *be the t-step value function for following policy* π*, that is,* $Q_t^\pi(b,\mathfrak{a}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b,\mathfrak{a}) V_{t-1}^\pi(b_\mathbf{z}^\mathfrak{a})$*. If for all b,* $\rho(b) \geq 0$*,*

$$V_t^\pi(b) \geq (1-\epsilon)V_t^*(b), \tag{11.66}$$

and $Q_t^\pi(b,\mathfrak{a})$ *is non-negative, monotone, and submodular then, for* $\epsilon \in [0,1]$*,*

$$(\mathfrak{B}^M V_t^\pi)(b) \geq (\frac{1}{2})(1-\epsilon)(\mathfrak{B}^M V_t^*)(b). \tag{11.67}$$

Proof. Starting from (11.66) and, for a given $\mathfrak{a}$, on both sides taking the expectation over $\mathbf{z}$, and adding $\rho(b)$ (since $\rho(b) \geq 0$ and $\epsilon \leq 1$):

$$\rho(b) + \mathbb{E}_{\mathbf{z}|b,\mathfrak{a}}[V_t^\pi(b_\mathbf{z}^\mathfrak{a})] \geq \frac{1}{2}(\rho(b) + \mathbb{E}_{\mathbf{z}|b,\mathfrak{a}}[V_t^*(b_\mathbf{z}^\mathfrak{a})]).$$

From the definition of Q_t^π (2.5), we thus have:

$$Q_{t+1}^\pi(b,\mathfrak{a}) \geq (1-\epsilon)Q_{t+1}^*(b,\mathfrak{a}) \ \ \forall \mathfrak{a}. \tag{11.68}$$

From Theorem 10, we know

$$Q_{t+1}^\pi(b,\mathfrak{a}_\pi^G) \geq \frac{1}{2}Q_{t+1}^\pi(b,\mathfrak{a}_\pi^*), \tag{11.69}$$

where $\mathfrak{a}_\pi^G = \texttt{greedy-argmaxM}(Q_{t+1}^\pi(b,\cdot),V,I,K)$ and $\mathfrak{a}_\pi^* = \arg\max_{\mathfrak{a}\in I} Q_{t+1}^\pi(b,\mathfrak{a})$. Since $Q_{t+1}^\pi(b,\mathfrak{a}_\pi^*) \geq Q_{t+1}^\pi(b,\mathfrak{a})$ for any $\mathfrak{a} \in I$,

$$Q_{t+1}^\pi(b,\mathfrak{a}_\pi^G) \geq \frac{1}{2}Q_{t+1}^\pi(b,\mathfrak{a}_*^G), \tag{11.70}$$

where $\mathfrak{a}_*^G = \texttt{greedy-argmaxM}(Q_t^*(b,\cdot),V,I,K)$. Finally, (11.68) implies that $Q_{t+1}^\pi(b,\mathfrak{a}_*^G) \geq (1-\epsilon)Q_{t+1}^*(b,\mathfrak{a}_*^G)$, so:

$$\begin{aligned} Q_{t+1}^\pi(b,\mathfrak{a}_\pi^G) &\geq \frac{1}{2}(1-\epsilon)Q_{t+1}^*(b,\mathfrak{a}_*^G) \\ (\mathfrak{B}^M V_t^\pi)(b) &\geq \frac{1}{2}(1-\epsilon)(\mathfrak{B}^M V_t^*)(b). \end{aligned} \tag{11.71}$$

□

Theorem 11 *Let* $\mathrm{M} = \langle S, \mathcal{M}, \Omega, T, O, R, b_0, h\rangle$ *be an active perception POMDP where* $\mathcal{M}$ *is a matroid. Let* π *be a policy that maps beliefs b to actions* $\mathfrak{a}$ *in I,* $\pi(b) = \mathfrak{a}$*. Let*

$Q_t^\pi(b, \mathfrak{a})$ *be the* t*-step value function for following policy* π*, that is,* $Q_t^\pi(b, \mathfrak{a}) = \rho(b) + \sum_{\mathbf{z}\in\Omega} \Pr(\mathbf{z}|b, \mathfrak{a}) V_{t-1}^\pi(b_{\mathbf{z}}^{\mathfrak{a}})$*. If for all policies* π*,* $Q_t^\pi(b, \mathfrak{a})$ *is non-negative, monotone and submodular in* $\mathfrak{a}$*, then for all* b*,*

$$V_t^G(b) \geq (\frac{1}{2})^{2t} V_t^*(b). \tag{11.72}$$

Proof. By induction on t. The base case, $t = 0$, holds because $V_0^G(b) = \rho(b) = V_0^*(b)$.

In the inductive step, for all b, we assume that

$$V_{t-1}^G(b) \geq (\frac{1}{2})^{2t-2} V_{t-1}^*(b), \tag{11.73}$$

and must show that

$$V_t^G(b) \geq (\frac{1}{2})^{2t} V_t^*(b). \tag{11.74}$$

Applying Lemma 6 with $V_t^\pi = V_{t-1}^G$ and $(1-\epsilon) = (\frac{1}{2})^{2t-2}$ to (4.23):

$$\begin{aligned}(\mathfrak{B}^M V_{t-1}^G)(b) &\geq (\frac{1}{2})^{2t-2}(\frac{1}{2})(\mathfrak{B}^M V_{t-1}^*)(b)\\ V_t^G(b) &\geq (\frac{1}{2})^{2t-1}(\mathfrak{B}^M V_{t-1}^*)(b).\end{aligned}$$

Now applying Corollary 2 with $V_{t-1}^\pi = V_{t-1}^*$:

$$\begin{aligned}V_t^G(b) &\geq (\frac{1}{2})^{2t-1}\frac{1}{2}(\mathfrak{B}^* V_{t-1}^*)(b)\\ V_t^G(b) &\geq (\frac{1}{2})^{2t} V_t^*(b).\end{aligned} \tag{11.75}$$

□

Summary

Active perception is the ability of an agent to take actions to reduce its uncertainty when it is uncertain about the world it is acting in. This thesis tackles the challenge of active perception for tracking people in multi-camera networks. Multi-camera systems are routinely used for security, surveillance and person tracking. A key challenge in the design of such networks is the efficient allocation of scarce resources such as the *bandwidth* required to communicate the collected data to a central server, the *CPU cycles* required to process that data, the *energy* costs of the entire system or the *manpower* required to manually monitor all the collected data. Maintaining surveillance is an example of an *active perception* task where an agent must select k out of the n available cameras to allocate the scarce resources to minimize its uncertainty about the state of the world. To this end, in this thesis we propose and give multiple results and methods for resource allocation in multi-camera networks, that in principle, reduce the uncertainty about the about the position of each person in the scene, in turn, enabling an agent to take actions to track people with a resource constrained multi-camera network.

Specifically, we propose:

- Equivalence of ρPOMDP and POMDP-IR, two frameworks that allow to expressing the reward in a partially observable Markov decision process (POMDP) as a function of the *belief* of the agent. This equivalence shows that entropy (uncertainty) of a probability distribution over a hidden variable can be approximated by giving the agent a choice to make predictions about the hidden state and then rewarding the agent for making right predictions.
- Greedy PBVI: a new POMDP planning methods that uses *greedy* maximization to scale in the large combinatorial action space of an active perception POMDP that consists of $\binom{n}{k}$ subset of cameras.
- Probably approximately correct (PAC) greedy maximization that is an approximate but computationally cheaper version of greedy maximization that requires access to only *upper* and *lower* confidence bounds on a submodular function to maximize it.
- PartiMax: a particle filter based algorithm that greatly reduces the computational cost of tracking people in ultra high resolution images by applying a trained person detector only to the k most relevant part of an image.

For each of the above mentioned methods, we prove results that guarantee firm error bounds and establish the conditions for application of those error bounds. To test the empirical performance of our methods we conduct experiments on multiple real-life datasets that show that greedy PBVI and PAC greedy maximization achieve similar performance as the existing methods but at a fraction of the computational cost. PartiMax when applied on a real-life tracking problem retains 80% of the original tracking performance while processing only 10% of the original image while running in real-time.

Finally, we propose deep anticipatory network (DAN) that enable an model-free approach for active perception, thus, enabling an agent to learn the optimal policy from its experience given some ground truth data. We show that a DAN agent can potentially be trained in a supervised fashion to take actions to reduce its uncertainty.

In summary, this thesis pushes the state-of-the-art in active perception for person tracking, by proposing multiple methods and results applicable in general and especially for tracking people in resource constrained multi-camera networks.

Yash Satsangi
University of Amsterdam
November 2018

Samenvatting

Actieve perceptie is het vermogen van een agent om acties te ondernemen om zijn onzekerheid te verminderen wanneer deze onzeker is over de wereld waarin het actief is. Dit proefschrift gaat in op de uitdaging van actieve perceptie bij het volgen van mensen in multicameranetwerken. Multicamerasystemen worden routinematig gebruikt in beveiliging, bewaking en persoonsregistratie. Een belangrijke uitdaging bij het ontwerp van dergelijke netwerken is de efficiënte toewijzing van schaarse middelen, zoals de *bandwidth* die nodig is om de verzamelde gegevens te communiceren naar een centrale server, de *CPU cycles* die nodig is om die gegevens te verwerken, de *energiekosten* van het gehele systeem of de *mankracht* die nodig is om alle verzamelde gegevens handmatig te controleren. Cameratoezicht is een voorbeeld van een *actieve perceptie* taak, waarbij een agent k van de n beschikbare camera's moet selecteren om de schaarse middelen toe te wijzen om de onzekerheid over de toestand van de wereld tot een minimum te beperken. Hiertoe stellen we in dit proefschrift meerdere resultaten en methoden voor brontoewijzing in multi-cameranetwerken voor, die in principe de onzekerheid over de positie van elke persoon in de scene verminderen. Hierdoor, een agent op zijn beurt in staat is om acties te ondernemen om mensen te traceren met een beperkt netwerk met meerdere camera's.

Concreet stellen we voor:

- Gelijkwaardigheid van ρPOMDP en POMDP-IR, twee raamwerken die toelaten om de beloning uit te drukken in een partially observable Markov decision process (POMDP) als een functie van het *belief* van de agent. Deze equivalentie toont aan dat de entropie (onzekerheid) van een kansverdeling over een verborgen variabele kan worden benaderd door de agent een keuze te geven om voorspellingen te doen over de verborgen status en vervolgens de agent te belonen voor het maken van juiste voorspellingen.

- Greedy PBVI: een nieuwe POMDP-planningsmethode die de *greedy* maximization gebruikt om te schalen in de grote combinatorische actieruimte van een actieve perceptie-POMDP die bestaat uit $\binom{n}{k}$ subset van camera's.

- Probably approximately correct (PAC) greedy maximization die een benaderende maar computationeel goedkopere versie van greedy maximization is die toegang tot alleen *upper* en *lower* vertrouwenslimieten op een submodular functie vereist om deze te maximaliseren.

- PartiMax: een op deeltjesfilter gebaseerd algoritme dat de computationele kosten

van het volgen van mensen in afbeeldingen met ultrahogeresolutie aanzienlijk vermindert door een detector voor getrainde personen alleen toe te passen op het k meest relevante deel van een afbeelding.

Voor elk van de bovengenoemde methoden bewijzen we resultaten die foutegrenzen garanderen en de voorwaarden voor toepassing van die foutgrenzen bepalen. Om de empirische prestaties van onze methoden te testen, voeren we experimenten uit op meerdere real-life datasets die aantonen dat greedy PBVI en PAC greedy maximization vergelijkbare prestaties als de bestaande methoden bereiken, maar tegen een fractie van de computerkosten. PartiMax behoudt bij toepassing op een real-life trackingprobleem 80% van de oorspronkelijke trackingprestaties, terwijl slechts 10% van het originele beeld wordt verwerkt en het in realtime wordt uitgevoerd.

Tenslotte stellen we een Deep Anticipatory Network (DAN) voor dat een modelvrije benadering voor actieve perceptie mogelijk maakt, waardoor een agent het optimale beleid kan leren aan de hand van zijn ervaring, gegeven enkele *ground truth data*. We laten zien dat een DAN-agent potentieel op een gecontroleerde manier getraind kan worden om acties te ondernemen om zijn onzekerheid te verminderen.

Samengevat verbetert dit proefschrift de state-of-the-art in actieve perceptie voor het volgen van personen door het voorstellen van meerdere methoden en resultaten die van toepasbaar zijn in het algemeen en in het bijzonder voor het traceren van mensen in beperkte multicameranetwerken.

Yash Satsangi
University of Amsterdam
November 2018

Hindi Abstract

एक्टिव परसेप्शन एक एजेंट की अपनी अनिश्चितता को कम करने व् अपनी जानकारी बढ़ाने हेतु कार्रवाई करने की क्षमता को कहते हैं।यह थीसिस बहु-कैमरा नेटवर्क में व्यक्तियों को ट्रैक करने हेतु एक्टिव परसेप्शन की चुनौतियों को सम्भोधित करती है। आजकल सार्वजनिक स्थान, जैसे की शॉपिंग मॉल या एयरपोर्ट, व्यक्तियों को ट्रैक करने हेतु या सुरक्षा व्यवस्था बनाए रखने के लिए, नियमित रूप से बहु-कैमरा नेटवर्क का उपयोग करते हैं। इन नेटवर्क के डिज़ाइन में दुर्लभ संसाधनों का कुशल वितरण करना एक महत्वपूर्ण चुनौती है, जैसे की, एकत्रित डाटा को सेंट्रल सर्वर तक पहुंचने के लिए बैंडविड्थ, या एकत्रित डाटा को प्रोसेस करने के लिए सेंट्रल सर्वर की क्षमता, या बहु-कैमरा नेटवर्क को चलाने के लिए पावर/एनर्जी, या एकत्रित डाटा को मॉनिटर करने के लिए श्रम-शक्ति। यह दुर्लभ संसाधनों का कुशल वितरण करने की समस्या एक एक्टिव परसेप्शन की समस्या का उदाहरण है, जिसमे एक एजेंट को n उपलब्ध कैमरा में से k कैमरा चुनना होता है जिससे की वह अधिक से अधिक जानकारी प्राप्त कर सके और अधिक से अधिक अनिश्चिता घटा सके।इस उद्देश्य हेतु, यह थीसिस विभिन्न तरीको व परिणामो को प्रस्तुत करती है, जिससे की एक संशाधन बाधित बहु कैमरा नेटवर्क, अपने साधनो का इस प्रकार से उपयोग कर सके जिससे की वह सुरक्षा बनाये रखने के लिए अधिक से अधिक जानकारी प्राप्त कर सके।

Overview of Publications

As required, we provide the list of publications this thesis is based upon along with the authors and their contributions:

- Yash Satsangi, Shimon Whiteson, Frans A. Oliehoek, and Matthijs T. J. Spaan. Exploiting Submodular Value Functions for Scaling Up Active Perception. *Autonomous Robots, 2017.* (Chapter 3 and 4 are based on this.)
- Yash Satsangi, Shimon Whiteson, Frans A. Oliehoek, and Henri Bouma. Real-Time Resource Allocation for Tracking Systems. *In Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence (UAI), 2017.* (Chapter 6 is based on this.)
- Yash Satsangi, Shimon Whiteson, and Frans A. Oliehoek. Probably Approximately Correct Greedy Maximization with Efficient Bounds on Information Gain for Sensor Selection. *In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI), 2016.* (Chapter 5 is based on this.)
- Yash Satsangi, Shimon Whiteson, and Frans A. Oliehoek. Exploiting Submodular Value Functions for Faster Dynamic Sensor Selection. *In Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI), 2015.* (Chapter 4 is based on this.)
- Yash Satsangi, Shimon Whiteson, and Frans A. Oliehoek. Probably Approximately Correct Greedy Maximization. *In Proceedings of the Fifteenth International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), 2016.* (Chapter 5 is based on this.)
- Yash Satsangi, Shimon Whiteson, and Matthijs T. J. Spaan. An Analysis of Piecewise-Linear and Convex Value Function for Active Perception POMDPs. *Technical Report IAS-15-01, University of Amsterdam, October 2015.* (Chapter 3 is based on this.)
- Yash Satsangi, Shimon Whiteson, and Frans A. Oliehoek. Exploiting Submodular Value Functions for Faster Dynamic Sensor Selection: Extended Version. *Technical Report IAS-14-02, University of Amsterdam, December 2014.* (Chapter 4 is based on this.)

In all the publications, Yash Satsangi (the first author) performed majority of the research including coming up with the idea, conducting analysis, running experiments and writing the paper. All co-authors provided useful feedback (via multiple brainstorming sessions) that led to enhancement of the initial idea. Shimon Whiteson edited all the papers and provided continuous supervision via many brainstorming sessions. Frans A. Oliehoek provided supervision and feedback on all publications via many brainstorming sessions. Matthijs T. J. Spaan and Henri Bouma provided supervision and useful feedback on their respective publications. For the sake of completion, note that Chapter 7 and section 4.2 from Chapter 4 are yet to be published in any publication other than this thesis.

Acknowledgements

I would like to thank the committee members for their time, efforts, and for carefully reviewing the thesis and pointing to important mistakes and enhancements. I would like to thank my supervisors, Shimon and Frans, for many things, all of which I cannot enlist here, but concisely, for their support, guidance, for taking the time to teach me to reason clearly, and for running the last minute races. I would like to thank Max Welling for agreeing to be my promoter for helping me pull through my phd program, especially during the last few years. I would like to thank Felice Arends and other secretaries/administrative/support staff at the Informatics Institute at the University of Amsterdam for helping me at various stages with numerous things, especially with the administrative processes and the endless paperwork. I would like to thank the University of Amsterdam (UvA) for giving me this opportunity. I would like to thank the University of Oxford (UOx) and the staff at the computer science department at the UOx for providing me with the opportunity to visit the UOx.

I would like to thank my paranymphs for their help in the defence of this thesis. I would like to thank my colleagues and friends at the UvA, especially the members of intelligent sensory information systems, and computer vision research groups for acting as a surrogate social activity group. I would like to thank my colleagues in AMLab and Intelligent Autonomous Systems (IAS) for the many fun moments and lunches. I would like to thank the members of the causality group at the UvA for many good office discussions. I would like to thank my colleagues at the data science department at TNO, especially Maaike, for their help in the completion of this thesis. I would like to thank the members of the Whiteson research lab at the UOx. I would like to thank the many officemates that I had during my time at the UvA. I would like to thank the students and colleagues that I interacted with while my time at the UvA as part of a student thesis, project or a course.

A very special thanks to Bart Joosten for helping me through some very critical moments in the last couple of years.

I would like to thank the STW user committee for its advice regarding active perception for tracking systems. I would like to thank Henri Bouma and Leon Kester from TNO for providing multiple datasets and for their useful feedback throughout my phd. I would like to thank the other user committee members such as Eagle Vision, Saxion, and STW.

I would like to thank many friends outside of my work like the windmill crowd, members of KVA, my volleybal team, USC volleybal group, squash club at USC especially Gwylan, netherlands docents, my trainer Jinga, and many others. I would like to thank

Stevan for all the ballistic stupid things there are in this world. I would like to thanks my friends in India and US, especially Amit Abbi, for their continuous support.

I would like to thank my parents for their endless support and help. For the same, I would like to thank my extended family and relatives.

Yash Satsangi
University of Amsterdam
November 2018

Bibliography

P. K. Allen and R. Bajcsy. *Object recognition using vision and touch*. PhD thesis, University of Pennsylvania., 1985. (Cited on page 141.)

M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, and N. de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016. (Cited on page 140.)

A. Antos and I. Kontoyiannis. Convergence properties of functional estimates for discrete distributions. *Random Structures & Algorithms*, 19(3-4):163–193, 2001. (Cited on page 90.)

M. Araya-lópez, V. Thomas, O. Buffet, and F. Charpillet. A POMDP extension with belief-dependent rewards. In *Advances in Neural Information Processing Systems*, pages 64–72, 2010. (Cited on pages 7, 9, 20, 34, 63, and 64.)

M. Araya-López, O. Buffet, V. Thomas, and F. Charpillet. Active learning of MDP models. *Recent Advances in Reinforcement Learning*, pages 42–53, 2012. (Cited on page 143.)

K. J. Astrom. Optimal control of Markov decision processes with incomplete state estimation. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965. (Cited on page 5.)

J.-Y. Audibert and S. Bubeck. Best arm identification in multi-armed bandits. In *23rd Conference on Learning Theory*, pages 13–p, 2010. (Cited on pages 84, 101, and 145.)

P. Bachman, A. Sordoni, and A. Trischler. Learning algorithms for active learning. *Proceedings of the 34th International Conference on Machine Learning*, pages 301–310, 2017.

A. Badanidiyuru and J. Vondrák. Fast algorithms for maximizing submodular functions. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1497–1514, 2014. (Cited on pages 73 and 144.)

H. Bai, D. Hsu, W. S. Lee, and V. A. Ngo. Monte Carlo value iteration for continuous-state POMDPs. In *Algorithmic foundations of robotics IX*, pages 175–191. Springer, 2010. (Cited on pages 101 and 143.)

R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988. (Cited on pages 1 and 2.)

R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos. Revisiting active perception. *arXiv preprint arXiv:1603.02729*, 2016. (Cited on pages 1, 2, and 141.)

M.-F. Balcan and N. J. Harvey. Learning submodular functions. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 793–802. ACM, 2011. (Cited on page 154.)

R. Benenson, M. Omran, J. Hosang, and B. Schiele. Ten years of pedestrian detection, what have we learned. In *European Conference on Computer Vision*, pages 613–627, 2014. (Cited on page 146.)

B. Benfold and I. Reid. Stable multi-target tracking in real-time surveillance video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3457–3464, 2011. (Cited on page 104.)

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. (Cited on page 131.)

D. P. Bertsekas. *Dynamic Programming and Optimal Control*, volume II. Athena Scientific, 3rd edition, 2007.

J. Binney, A. Krause, and G. S. Sukhatme. Informative path planning for an autonomous underwater vehicle. In *IEEE International Conference on Robotics and Automation*, pages 4791–4796. IEEE, 2010. (Cited on page 2.)

B. Bonet and H. Geffner. Solving POMDPs: Rtdp-bel vs. point-based algorithms. In *Proceedings of the Twenty-First International Jont Conference on Artifical Intelligence*, pages 1641–1646, 2009. (Cited on page 6.)

A. Borji and L. Itti. State-of-the-art in visual attention modeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):185–207, 2013. (Cited on page 141.)

H. Bouma, J. Baan, S. Landsmeer, C. Kruszynski, G. van Antwerpen, and J. Dijk. Real-time tracking and fast retrieval of persons in multiple surveillance cameras of a shopping mall. In *Multisensor, Multisource*

Information Fusion: Architectures, Algorithms, and Applications 2013, volume 8756, 2013. (Cited on page 45.)

C. Boutilier, T. Dean, and S. Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11(1):94, 1999. (Cited on page 5.)

S. Boyd and J. Mattingley. Branch and bound methods. 2007. (Cited on page 85.)

N. D. Bruce and J. K. Tsotsos. Saliency, attention, and visual search: An information theoretic approach. *Journal of Vision*, 9(3):5–5, 2009. (Cited on pages 141 and 146.)

S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, pages 1–122, 2012.

W. Burgard, D. Fox, and S. Thrun. Active mobile robot localization by entropy minimization. In *Proceedings of the Second EUROMICRO Workshop on Advanced Mobile Robots 1997*, pages 155–162. IEEE, 1997. (Cited on page 141.)

A. Cassandra, M. L. Littman, and N. L. Zhang. Incremental pruning: A simple, fast, exact method for partially observable Markov decision processes. In *Proceedings of the Thirteenth conference on Uncertainty in Artificial Intelligence*, pages 54–61. Morgan Kaufmann Publishers Inc., 1997. (Cited on page 35.)

Y. Chen and A. Krause. Near-optimal batch mode active learning and adaptive submodular optimization. In *Proceedings of the 31st International Conference on Machine Learning*, pages 160–168, 2013. (Cited on pages 23, 27, and 144.)

Y. Chen, S. Javdani, A. Karbasi, J. A. Bagnell, S. Srinivasa, and A. Krause. Submodular surrogates for value of information. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 3511–3518, 2015. (Cited on page 146.)

Y. Chen, H. Hassani, and A. Krause. Near-optimal Bayesian active learning with correlated and noisy tests. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics.*, pages 223–231, 2017. (Cited on page 146.)

H.-T. Cheng. *Algorithms for partially observable Markov decision processes*. PhD thesis, University of British Columbia, 1988. (Cited on pages 6 and 17.)

K. Ciosek and S. Whiteson. Expected policy gradients. *arXiv preprint arXiv:1706.05374*, 2017. (Cited on page 140.)

M. Conforti and G. Cornuéjols. Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem. *Discrete applied mathematics*, 7(3): 251–274, 1984. (Cited on page 145.)

T. Cover and J. Thomas. *Entropy, relative entropy and mutual information*. Wiley-Interscience, 1991. (Cited on pages 7, 57, 61, 93, 107, and 165.)

N. Dalal and B. Triggs. Histogram of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 886–893, 2005. (Cited on page 146.)

Y. David, D. Di Castro, and Z. Karnin. One-shot session recommendation systems with combinatorial items. *arXiv preprint arXiv:1607.01381*, 2016. (Cited on pages 144, 154, and 155.)

F. Dellaert and R. Collins. Fast image-based tracking by selective pixel integration. In *ICCV Workshop on Frame-Rate Vision*, 1999. (Cited on page 146.)

M. Denil, L. Bazzani, H. Larochelle, and N. de Freitas. Learning where to attend with deep architectures for image tracking. *Neural computation*, 2012. (Cited on pages 7, 147, and 148.)

P. Dollár, S. Belongie, and P. Perona. The fastest pedestrian detector in the west. In *British Machine Vision Conference*, volume 2, 2010. (Cited on pages 5 and 45.)

P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4), 2012. (Cited on page 7.)

P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE Transactions* *on Pattern Analysis and Machine Intelligence*, 36(8), 2014. (Cited on pages 7, 123, and 146.)

A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer Science & Business Media, 2001. (Cited on pages 92, 100, and 107.)

A. Eck and L.-K. Soh. Evaluating POMDP rewards for active perception. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pages 1221–1222. International Foundation for Autonomous Agents and Multiagent Systems, 2012. (Cited on pages 3, 18, and 143.)

E. Even-Dar, S. Mannor, and Y. Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of machine learning research*, 7(Jun):1079–1105, 2006. (Cited on pages 84 and 85.)

G. Farquhar, T. Rocktschel, M. Igl, and S. Whiteson. TreeQN and ATreeC: Differentiable tree planning for deep reinforcement learning. 2017. (Cited on page 154.)

P. Felzenszwalb, Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1627–1645, 2010. (Cited on page 146.)

M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. *An analysis of approximations for maximizing submodular set functionsII*. Springer, 1978. (Cited on pages 66, 68, 69, and 145.)

J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson. Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926*, 2017. (Cited on page 140.)

S. Fujishige. Polymatroidal dependence structure of a set of random variables. *Information and control*, 39(1): 55–72, 1978. (Cited on page 166.)

Y. Gal, R. Islam, and Z. Ghahramani. Deep Bayesian active learning with image data. *Proceedings of the 34th International Conference on Machine Learning*, pages 1183–1192, 2017. (Cited on page 140.)

S. Gelly and D. Silver. Combining online and offline knowledge in UCT. In *Proceedings of the 24th International Conference on Machine Learning*, pages 273–280. ACM, 2007. (Cited on page 154.)

D. Gilbarg and N. Trudinger. *Elliptic Partial Differential Equations of Second Order*. U.S. Government Printing Office, 2001. (Cited on page 63.)

R. Girshick, J. Donahue, T. Darrel, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2014. (Cited on page 146.)

D. Golovin and A. Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, pages 427–486, 2011. (Cited on page 146.)

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014. (Cited on page 132.)

I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein GANs. *arXiv preprint arXiv:1704.00028*, 2017.

A. Gupta, A. Mittal, and L. Davis. Cost: An approach for camera selection and multi-object inference ordering in dynamic scenes. In *IEEE International Conference on Computer Vision*, pages 1–8, 2007. (Cited on page 142.)

M. Hausknecht and P. Stone. Deep recurrent Q-learning for partially observable MDPs. In *2015 AAAI Fall Symposium Series*, 2015. (Cited on pages 8, 47, 134, and 147.)

M. Hauskrecht. Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, pages 33–94, 2000. (Cited on page 6.)

A. Hero and D. Cochran. Sensor management: Past, present, and future. *IEEE Sensors Journal*, 11(12): 3064–3075, 2011. (Cited on page 142.)

S.-W. Ho, T. Chan, and A. Grant. The confidence interval of entropy estimation through a noisy channel. In

IEEE Information Theory Workshop (ITW), pages 1–5, 2010. (Cited on page 145.)

D. Hochbaum. Approximating covering and packing problems: set cover, vertex cover, independent set, and related problems. In *Approximation algorithms for NP-hard problems*. PWS Publishing Co., 1996. (Cited on page 128.)

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. (Cited on page 134.)

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963. (Cited on pages 74, 89, and 145.)

J. Hoey, R. St-Aubin, A. J. Hu, and C. Boutilier. Spudd: Stochastic planning using decision diagrams. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence*, pages 279–288, Stockholm, Sweden, 1999. (Cited on page 6.)

G. A. Hollinger and G. S. Sukhatme. Sampling-based motion planning for robotic information gathering. Citeseer, 2013. (Cited on page 141.)

G. A. Hollinger, B. Englot, F. Hover, U. Mitra, and G. S. Sukhatme. Uncertainty-driven view planning for underwater inspection. In *IEEE International Conference on Robotics and Automation*, pages 4884–4891. IEEE, 2012. (Cited on page 141.)

J. Hosang, M. Omran, R. Beneson, and B. Schiele. Taking a deeper look at pedestrians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4073–4082, 2015. (Cited on page 146.)

N. Hu, H. Bouma, and M. Worring. Tracking individuals in surveillance video of a high-density crowd. SPIE, 2012. (Cited on page 104.)

C. Huang, S. Lucey, and D. Ramanan. Learning policies for adaptive tracking with deep feature cascades. *Proceedings of International Conference on Computer Vision*, 2017.

R. Iyer and J. Bilmes. Submodular point processes with applications to machine learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pages 388–397, 2015. (Cited on page 128.)

M. Jaderberg, V. Mnih, W. M. Czarnecki, T. Schaul, J. Z. Leibo, D. Silver, and K. Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. *International Conference on Learning Representations*, 2016. (Cited on page 147.)

M. R. James and S. Singh. Sarsalandmark: an algorithm for learning in POMDPs with landmarks. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 585–591. International Foundation for Autonomous Agents and Multiagent Systems, 2009. (Cited on page 149.)

K. Jamieson, M. Malloy, R. Nowak, and S. Bubeck. lil'ucb: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory*, pages 423–439, 2014. (Cited on page 84.)

S. Ji, R. Parr, and L. Carin. Nonmyopic multiaspect sensing with partially observable Markov decision processes. *IEEE Transactions on Signal Processing*, pages 2720–2730, 2007. (Cited on page 142.)

J. Joseph, F. Doshi-Velez, A. S. Huang, and N. Roy. A Bayesian nonparametric approach to modeling motion patterns. *Autonomous Robots*, 31(4):383, 2011. (Cited on page 7.)

S. Joshi and S. Boyd. Sensor selection via convex optimization. *IEEE Transactions on Signal Processing*, pages 451–462, 2009. (Cited on page 142.)

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, pages 99–134, 1998. (Cited on pages 5, 13, 17, and 143.)

S. Kalyanakrishnan, A. Tewari, P. Auer, and P. Stone. PAC subset selection in stochastic multi-armed bandits. 2012. (Cited on page 84.)

Z. Karnin, T. Koren, and O. Somekh. Almost optimal exploration in multi-armed bandits. In *International*

Conference on Machine Learning, pages 1238–1246, 2013. (Cited on page 84.)

S. Katt, F. A. Oliehoek, and C. Amato. Learning in POMDPs with Monte Carlo tree search. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1819–1827. PMLR, 2017. (Cited on page 149.)

M. D. Kelly. Edge detection in pictures by computer using planning. Technical report, 1971. (Cited on page 141.)

K. Kim, D. Lee, and I. Essa. Detecting regions of interest in dynamic scenes with camera motions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1258–1265. IEEE, 2012. (Cited on page 146.)

K. Kirchhoff and J. Bilmes. Submodularity for data selection in statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 131–141, 2014. (Cited on page 155.)

M. J. Kochenderfer. *Decision Making Under Uncertainty: Theory and Application*. MIT Press, 2015. (Cited on page 5.)

L. Kocsis and C. Szepesvári. Bandit based Monte-Carlo planning. In *European conference on machine learning*, pages 282–293. Springer, 2006. (Cited on pages 101 and 144.)

J. Kooij, G. Englebienne, and D. Gavrila. A non-parametric hierarchical model to discover behavior dynamics from tracks. In *European Conference on Computer Vision*, pages 270–283. Springer, 2012. (Cited on page 7.)

G. Kootstra. *Visual attention and active vision: From natural to artificial systems*. PhD thesis, University of Groningen, 2010. (Cited on page 146.)

I. Kostrikov, D. Erhan, and S. Levine. End to end active perception. 2016. (Cited on page 148.)

A. Krause and D. Golovin. Submodular function maximization. Cambridge University Press, 2014. (Cited on pages 25, 58, 87, 110, 144, 146, and 152.)

A. Krause and R. G. Gomes. Budgeted nonparametric learning from data streams. In *Proceedings of the 27th International Conference on Machine Learning*, pages 391–398, 2010. (Cited on page 144.)

A. Krause and C. Guestrin. Optimal nonmyopic value of information in graphical models - efficient algorithms and theoretical limits. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 1339–1345, 2005a.

A. Krause and C. Guestrin. Near-optimal nonmyopic value of information in graphical models. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pages 324–331, 2005b. (Cited on pages 57, 98, 107, and 144.)

A. Krause and C. Guestrin. Near-optimal observation selection using submodular functions. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pages 481–492, 2007.

A. Krause and C. Guestrin. Optimal value of information in graphical models. *Journal of Artificial Intelligence Research*, pages 557–591, 2009.

C. Kreucher, K. Kastella, and A. O. Hero. Sensor management using an active sensing approach. *Signal Processing*, 85(3):607–624, 2005. (Cited on pages 3 and 142.)

V. Krishnamurthy and D. V. Djonin. Structured threshold policies for dynamic sensor schedulinga partially observed Markov decision process approach. *IEEE Transactions on Signal Processing*, 55(10):4938–4957, 2007. (Cited on page 142.)

A. Kumar and S. Zilberstein. Event-detecting multi-agent MDPs: Complexity and constant-factor approximation. In *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence*, pages 201–207, 2009. (Cited on page 143.)

R. R. Kumar, P. Varakantham, and A. Kumar. Decentralized planning in stochastic environments with submodular rewards. pages 3021–3028, 2017. (Cited on pages 144 and 154.)

H. Kurniawati, D. Hsu, and W. S. Lee. SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *In Proceedings of Robotics: Science and Systems*, 2008. (Cited on page 143.)

H. Kurniawati, Y. Du, D. Hsu, and W. S. Lee. Motion planning under uncertainty for robotic tasks with long time horizons. *The International Journal of Robotics Research*, 30(3):308–323, 2011. (Cited on page 143.)

M. La Cascia, S. Sclaroff, and V. Athitsos. Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3D models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 322–336, 2000. (Cited on page 104.)

Q. V. Le, A. Saxena, and A. Y. Ng. Active perception: Interactive manipulation for improving object detection. *Stanford University Journal*, 2008. (Cited on pages 2, 141, and 148.)

J. Li, L. Li, and T. Li. Multi-document summarization via submodularity. *Applied Intelligence*, 37:420–430, 2012. (Cited on page 144.)

H. Lin and J. Bilmes. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 912–920, 2010. (Cited on page 23.)

M. L. Littman. *Algorithms for sequential decision making*. PhD thesis, Brown University, 1996.

P.-L. Loh and S. Nowozin. Faster hoeffding racing: Bernstein races via jackknife estimates. In *International Conference on Algorithmic Learning Theory*, pages 203–217, 2013. (Cited on pages 74 and 145.)

W. S. Lovejoy. Computationally feasible bounds for partially observed Markov decision processes. *Operations Research*, pages 162–175, 1991.

T. Mandel, Y.-E. Liu, E. Brunskill, and Z. Popovic. Where to add actions in human-in-the-loop reinforcement learning. pages 2322–2328, 2017. (Cited on page 154.)

O. Maron and A. Moore. Hoeffding races: Accelerating model selection search for classification and function approximation. In *Advances in Neural Information Processing Systems*, January 1994. (Cited on pages 77 and 81.)

O. Maron and A. W. Moore. The racing algorithm: Model selection for lazy learners. In *Lazy learning*, pages 193–225. Springer, 1997. (Cited on pages 77 and 81.)

M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. *Optimization Techniques*, pages 234–243, 1978. (Cited on pages 27, 73, and 144.)

B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pages 2049–2057, 2013. (Cited on page 144.)

B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrák, and A. Krause. Lazier than lazy greedy. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 1812–1818, 2015. (Cited on pages 10, 27, 73, 105, and 144.)

B. Mirzasoleiman, A. Karbasi, and A. Krause. Deletion-robust submodular maximization: Data summarization with the right to be forgotten. In *International Conference on Machine Learning*, pages 2449–2458, 2017. (Cited on page 152.)

V. Mnih, C. Szepesvári, and J.-Y. Audibert. Empirical bernstein stopping. In *Proceedings of the 25th international conference on Machine learning*, pages 672–679. ACM, 2008. (Cited on pages 84 and 85.)

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013. (Cited on pages 8 and 134.)

V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014. (Cited on pages 141, 146, 147, and 148.)

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518

(7540):529–533, 2015. (Cited on pages 131 and 132.)

V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1928–1937, 2016. (Cited on pages 140 and 147.)

G. E. Monahan. A survey of partially observable Markov decision processes: Theory, models, and algorithms. *Management Science*, pages 1–16, 1982. (Cited on pages 6, 16, 51, and 143.)

E. Monari and K. Kroschel. Dynamic sensor selection for single target tracking in large video surveillance networks. In *Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 539–546. IEEE, 2010. (Cited on page 142.)

P. Natarajan, T. Hoang, K. Low, and M. Kankanhalli. Decision-theoretic approach to maximizing observation of multiple targets in multi-camera surveillance. In *Proceedings of International Conference on Autonomous Agents and Multiagent Systems*, pages 155–162, 2012. (Cited on pages 4 and 142.)

G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions. *Mathematical Programming*, 14(1):265–294, 1978. (Cited on pages 10, 23, 25, 50, 52, 66, 77, and 145.)

X. Nguyen, M. J. Wainwright, and M. I. Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 56(11):5847–5861, 2010.

S. Nowozin. Improved information gain estimates for decision tree induction. In *Proceedings of the 29th International Conference on Machine Learning*, pages 571–578, 2012a. (Cited on pages 74, 76, and 145.)

S. Nowozin. http://www.nowozin.net/sebastian/blog/estimating-discrete-entropy-part-1.html, 2012b. (Cited on page 90.)

S. Nowozin, B. Cseke, and R. Tomioka. f-GAN: Training generative neural samplers using variational divergence minimization. In *Advances in Neural Information Processing Systems*, pages 271–279, 2016.

J. Oh, V. Chockalingam, S. Singh, and H. Lee. Control of memory, active perception, and action in minecraft. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, pages 2790–2799. JMLR. org, 2016. (Cited on page 141.)

F. A. Oliehoek, S. Whiteson, and M. T. Spaan. Approximate solutions for factored dec-POMDPs with many agents. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, pages 563–570. International Foundation for Autonomous Agents and Multiagent Systems, 2013. (Cited on page 69.)

P. Ondruska and I. Posner. Deep tracking: Seeing beyond seeing using recurrent neural networks. *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, pages 3361–3367, 2016. (Cited on page 148.)

S. Ontañón. Combinatorial multi-armed bandits for real-time strategy games. *Journal of Artificial Intelligence Research*, 58:665–702, 2017. (Cited on page 155.)

L. Orseau, T. Lattimore, and M. Hutter. Universal knowledge-seeking agents for stochastic environments. In *International Conference on Algorithmic Learning Theory*, pages 158–172. Springer, 2013.

W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *IEEE International Conference on Computer Vision*, pages 2056–2063, 2013.

L. Paninski. Estimation of entropy and mutual information. *Neural computation*, 15(6), 2003. (Cited on pages 74, 76, 90, 145, and 152.)

D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell. Curiosity-driven exploration by self-supervised prediction. pages 2778–2787, 2017. (Cited on page 147.)

T. Patten, A. Kassir, W. Martens, B. Douillard, R. Fitch, and S. Sukkarieh. A Bayesian approach for time-constrained 3D outdoor object recognition. In *Proc. of IEEE ICRA Workshop on Scaling Up Active Perception*, 2015. (Cited on page 2.)

J. Pineau, G. J. Gordon, and S. Thrun. Anytime point-based approximations for large POMDPs. *Journal of*

Artificial Intelligence Research, pages 335–380, 2006. (Cited on pages 6, 9, 17, and 143.)

P. Poupart. *Exploiting structure to efficiently solve large scale partially observable Markov decision processes*. PhD thesis, University of Toronto, 2005.

P. Poupart, K.-E. Kim, and D. Kim. Closing the gap: Improved bounds on optimal POMDP solutions. In *Proceedings of the 21st International Conference on Automated Planning and Scheduling*, pages 194–201, 2011. (Cited on page 143.)

Y. Pu, L. P. Kaelbling, and A. Solar-Lezama. Learning to acquire information. *Proceedings of the 33rd Conference on Uncertainty in Artificial Intelligence*, 2017.

F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of the 25th international conference on Machine learning*, pages 784–791, 2008. (Cited on page 145.)

C. Raphael and G. Shani. The skyline algorithm for POMDP value function pruning. *Annals of Mathematics and Artificial Intelligence*, 65(1):61–77, 2012.

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: unified real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016. (Cited on page 146.)

S. Reeves and R. Hezar. Selection of observations in magnetic resonance spectroscopic imaging. In *Proceedings of International Conference on Image Processing.*, volume 1, pages 641–644. IEEE, 1995.

S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, pages 663–704, 2008. (Cited on pages 6 and 144.)

N. Roy and C. Earnest. Dynamic action spaces for information gain maximization in search and exploration. In *American Control Conference*, pages 6–pp. IEEE, 2006. (Cited on page 147.)

S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition, 2009. ISBN 0136042597, 9780136042594. (Cited on page 1.)

Y. Satsangi, S. Whiteson, and F. Oliehoek. Exploiting submodular value functions for faster dynamic sensor selection:extended version. Technical Report IAS-UVA-14-02, University of Amsterdam, Informatics Institute, 2014. (Cited on page 49.)

Y. Satsangi, S. Whiteson, and F. Oliehoek. Exploiting submodular value functions for faster dynamic sensor selection. In *AAAI 2015: Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 3356–3363, January 2015a. (Cited on pages 3, 4, 23, and 27.)

Y. Satsangi, S. Whiteson, and M. T. J. Spaan. An analysis of piecewise-linear and convex value functions for active perception POMDPs. Technical Report IAS-UVA-15-01, University of Amsterdam, Informatics Institute, 2015b. (Cited on page 30.)

Y. Satsangi, S. Whiteson, and F. Oliehoek. Probably approximately correct greedy maximization: Extended abstract. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 1387–1388, May 2016a. (Cited on page 74.)

Y. Satsangi, S. Whiteson, and F. Oliehoek. PAC greedy maximization with efficient bounds on information gain for sensor selection. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 3220–3227, July 2016b. (Cited on pages 73 and 74.)

Y. Satsangi, S. Whiteson, F. Oliehoek, and H. Bouma. Real-time resource allocation for tracking systems. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*, 2017a. (Cited on pages 2 and 11.)

Y. Satsangi, S. Whiteson, F. Oliehoek, and M. Spaan. Exploiting submodularity for scaling up active perception. *Autonomous Robots*, 2017b. (Cited on pages 30 and 49.)

A. Schumann, M. Bäuml, and R. Stiefelhagen. Person tracking-by-detection with efficient selection of part-detectors. In *IEEE Advanced Video and Signal Based Surveillance*, pages 43–50, 2013.

T. Schürmann. Bias analysis in entropy estimation. *Journal of Physics A: Mathematical and General*, 37(27):

L295, 2004. (Cited on pages 76 and 145.)

K. Schutte, G. Burghouts, N. Van der Stap, V. Westerwoudt, et al. Long-term behavior understanding based on the expert-based combination of short-term observations in high-resolution CCTV. In *Proceedings of SPIE*, volume 9995, 2016. (Cited on pages 104 and 123.)

P. Sermanet, D. Eigen, X. Zhang, et al. Overfeat: integrated recognition, localization and detection using convolutional networks. In *Internation Conference on Learning Representations*, 2014. (Cited on page 146.)

G. Shani, R. I. Brafman, and S. E. Shimony. Forward search value iteration for POMDPs. In *Proceedings of the 20th International Joint Conference on Artifical intelligence*, pages 2619–2624, 2007. (Cited on page 143.)

G. Shani, J. Pineau, and R. Kaplow. A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems*, pages 1–51, 2012. (Cited on pages 6, 9, 16, 17, 142, and 143.)

D. Sharma, A. Kapoor, and A. Deshpande. On greedy maximization of entropy. In *Proceedings of The 32nd International Conference on Machine Learning*, pages 1330–1338, 2015. (Cited on page 145.)

E. Shtrom, G. Leifman, and A. Tal. Saliency detection in large point sets. In *IEEE International Conference on Computer Vision*, pages 3591–3598, 2013. (Cited on page 146.)

D. Silver and J. Veness. Monte-carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems*, pages 2164–2172, 2010. (Cited on pages 7, 101, 107, and 144.)

D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, pages 387–395, 2014. (Cited on page 140.)

A. Singla, S. Tschiatschek, and A. Krause. Noisy submodular maximization via adaptive sampling with applications to crowdsourced image collection summarization. In *Proceedings of 30th Conference on Artificial Intelligence.*, pages 2037–2043, February 2016. (Cited on pages 73 and 145.)

R. D. Smallwood and E. J. Sondik. The optimal control of partially observable Markov processes over a finite horizon. *Operations Research*, pages 1071–1088, 1973. (Cited on page 17.)

A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2014. (Cited on pages 7 and 104.)

T. Smith and R. Simmons. Heuristic search value iteration for POMDPs. In *Proceedings of the 20th conference on Uncertainty in Artificial Intelligence*, pages 520–527. AUAI Press, 2004. (Cited on page 143.)

E. J. Sondik. *The Optimal Control of Partially Observable Markov Processes*. PhD thesis, Stanford University, United States – California, 1971. (Cited on pages 5, 6, 15, 17, and 143.)

M. T. J. Spaan. Cooperative active perception using POMDPs. In *AAAI Conference on Artificial Intelligence 2008: Workshop on Advancements in POMDP Solvers*, 2008. (Cited on pages 38, 142, and 147.)

M. T. J. Spaan. Partially observable Markov decision processes. In M. Wiering and M. van Otterlo, editors, *Reinforcement Learning: State of the Art*, pages 387–414. Springer Verlag, 2012.

M. T. J. Spaan and P. U. Lima. A decision-theoretic approach to dynamic sensor selection in camera networks. In *International Conference on Automated Planning and Scheduling*, pages 279–304, 2009. (Cited on pages 3, 4, 38, 107, and 142.)

M. T. J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005. (Cited on pages 17, 142, and 143.)

M. T. J. Spaan, T. S. Veiga, and P. U. Lima. Active cooperative perception in network robot systems using POMDPs. In *International Conference on Intelligent Robots and Systems*, pages 4800–4805, 2010.

M. T. J. Spaan, T. S. Veiga, and P. U. Lima. Decision-theoretic planning under uncertainty with information rewards for active cooperative perception. *Proceedings of the 2015 International Conference on Autonomous Agents & Multiagent Systems*, 29(6):1157–1185, 2015. (Cited on pages 2, 3, 7, 9, 22, 42, and 142.)

M. Sridharan, J. Wyatt, and R. Dearden. Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs. *Artificial Intelligence*, 174(11):704–725, 2010. (Cited on page 17.)

S. Stan, M. Zadimoghaddam, A. Krause, and A. Karbasi. Probabilistic submodular maximization in sub-linear time. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3241–3250, 2017. (Cited on pages 152 and 154.)

C. Stangor. *Introduction to Psychology*. Open Textbook Library. Flat World Knowledge, L.L.C., 2010. ISBN 9781936126484. (Cited on page 1.)

M. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. In *Advances in Neural Information Processing Systems*, pages 1577–1584, 2009. (Cited on pages 145 and 146.)

R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, Aug 1988. ISSN 1573-0565. doi: 10.1007/BF00115009. URL `https://doi.org/10.1007/BF00115009`. (Cited on page 8.)

R. S. Sutton and A. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. (Cited on pages 8 and 131.)

R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, pages 1057–1063, 2000. (Cited on page 140.)

H. Takamura and M. Okumura. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 781–789, 2009. (Cited on pages 23, 110, and 144.)

T. Taniguchi, T. Takano, and R. Yoshino. Multimodal hierarchical dirichlet process-based active perception. *arXiv preprint arXiv:1510.00331*, 2015. (Cited on page 141.)

L. Tessens, M. Morbee, H. Aghajan, and W. Philips. Camera selection for tracking in distributed smart camera networks. *ACM Transactions on Sensor Networks (TOSN)*, 10(2):23, 2014. (Cited on page 142.)

C.-K. Tham and M. Han. Information-driven sensor selection for energy-efficient human motion tracking. In *IEEE International Conference Distributed Computing in Sensor Systems.*, pages 11–19, 2013. (Cited on page 107.)

S. Thrun. Monte carlo POMDPs. In *Advances in Neural Information Processing Systems*, pages 1064–1070, 2000. (Cited on page 143.)

Y. Tian, P. Luo, X. Wang, and X. Tang. Pedestrian detection aided by deep learning semantic tasks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5079–5087, 2015. (Cited on page 146.)

M. Toussaint, L. Charlin, and P. Poupart. Hierarchical POMDP controller optimization by likelihood maximization. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, pages 562–570, 2008. (Cited on page 6.)

J. K. Tsotsos and K. Shubina. Attention and visual search: Active robotic vision systems that search. volume 114, pages 535–547, 2010. (Cited on page 146.)

L. Valiant. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books, 2013. (Cited on page 74.)

T. Veiga, M. T. J. Spaan, and P. U. Lima. Point-based POMDP solving with factored value function approximation. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014. (Cited on page 71.)

H. Wang, K. Yao, and D. Estrin. Information-theoretic approaches for sensor selection and placement in sensor networks for target localization and tracking. *Journal of Communications and Networks*, 7(4):438–449, 2005. (Cited on page 147.)

C. J. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, May 1989. (Cited on page 131.)

C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992. (Cited on page 8.)

K. Wei, R. Iyer, and J. Bilmes. Fast multi-stage submodular maximization. In *Proceedings of the 31st Interna-*

tional Conference on Machine Learning, pages 1494–1502, 2014. (Cited on pages 73 and 144.)

C. C. White. A survey of solution techniques for the partially observed Markov decision process. *Annals of Operations Research*, 32(1):215–230, 1991. (Cited on page 16.)

D. Wilkes and J. K. Tsotsos. Active object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition.*, pages 136–141. IEEE, 1992. (Cited on page 141.)

J. Williams, J. Fisher III, and A. Willsky. Sensor management for multiple target tracking with heterogeneous sensor models. In *Proc. SPIE*, volume 6235, 2006. (Cited on page 4.)

J. L. Williams, J. W. Fisher, and A. S. Willsky. Approximate dynamic programming for communication-constrained sensor network management. *IEEE Transactions on Signal Processing*, 55(8):4300–4311, 2007. (Cited on pages 3 and 142.)

L. A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982. (Cited on page 128.)

K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015. (Cited on page 148.)

Y. Ye and J. K. Tsotsos. Where to look next in 3d object search. In *Proceedings of International Symposium on Computer Vision, 1995*, pages 539–544. IEEE, 1995. (Cited on page 141.)

Y. Yue and C. Guestrin. Linear submodular bandits and their application to diversified retrieval. In *Advances in Neural Information Processing Systems*, pages 2483–2491, 2011. (Cited on pages 145 and 154.)