Learning Search Strategies from Human Demonstrations

Dissertation (2016)

Submitted to the School of Engineering, Doctoral Program on Manufacturing Systems and Robotics

École Polytechnique Fédérale de Lausanne (EPFL)

in partial fulfillment of the requirements for the degree of Doctor of Philosophy

by

Guillaume de Chambrier

Thesis Committee:
Prof. Aude Billard, thesis advisor

Lausanne, Switzerland
October, 2016

Chapter 1

Introduction

1.1 Motivation

Taking long term decisions or spontaneous reactive actions when presented with incomplete information or partial knowledge is paramount to the survival of any biological or synthetic entity. Reasoning given a state of uncertainty is a continuously occurring event throughout our livelihood. When considering long term decisions an abundance of examples come to mind. For instance, in economic investments uncertainty is to the best of efforts quantified and minimised in order to avoid unwarranted risks. Reactive actions are just as common; when looking for the snooze button of an alarm clock, early in the morning, our hand seems to autonomously search the surrounding space picking up sensory cues gradually acquiring information guiding us towards the button. All the above types of decision require the integration of evidence and an ability to predict the outcomes of the taken decisions in order to insure a favourable end state. In Artificial Intelligence (AI) & robotics, the ability to reason whilst taking uncertainty into consideration has resulted in mixed levels of success.

There has been noticeable success in artificial agents beating humans at board games such as backgammon (TD-Backgammon), chess (Deep blue) and now recently go (AlphaGo). The gap between robotic autonomous systems and humans starts to diverge however when the action space is continuous and uncertainty is non-negligible. Although there are recent examples of robots coping with such conditions such as opening doors (pose of the handle's position and shape uncertainty of the handle), walking downstairs (state transition uncertainty)(Asimo), and turning valves (DARPA Robotic challenge, DAC Johnson et al. (2015)), repetition and reproducibility of such behaviour is hard. This was highlighted in the results of the 2015 DAC in which issues (perception, control, software engineering, etc...) resulted in many robots losing balance and falling [1].

There is an increasing number of robotic application domains where perception is limited, such as planetary[2] and deep water exploration, and where optimal decisions taking uncertainty into consideration play a critical role in

[1] http://www.cs.cmu.edu/~cga/drc/
[2] http://exploration.esa.int/mars/

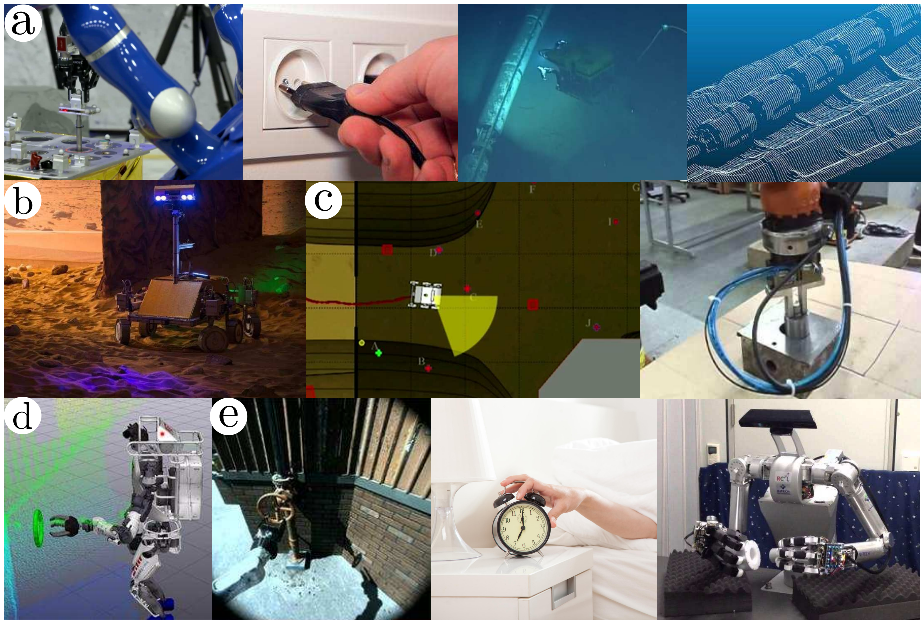


Figure 1.1: Examples of the decision making under uncertainty in both robotics and everyday life situations. (a) European Space Agency (ESA), remote orbital peg in hole task. (b)-(c) ESA, simulated exploration of a cave on Mars in the dark. (d)-(e) MIT DAC team, Atlas robot doing valve task, http://drc.mit.edu/. Other pictures include underwater exploration and industrial peg-in-hole assembly.

the success of the tasks undertaken. It would be advantageous to leverage human decision and action abilities for tasks in which the effect of uncertainty is problematic, such as exploration, search and manipulation.

It is not yet fully understood how decisions are taken, yet alone under uncertainty. The difficulty is that two processes responsible for the synthesis of our actions and decisions, that is our beliefs and desires, are not directly or easily measurable. There is growing interest in Neuroscience to understand the mechanisms underlying perception and decision making under uncertainty, Preuschoff et al. (2013); there is not yet a consensus on the biological mechanisms involved in decision making and efforts are ongoing[3] to construct plausible models of our decision processes. However, seemingly as a result of our prior knowledge and experience, we are better than current robotic systems at handling and dealing with uncertainty. Exploiting human abilities to accomplish tasks in which uncertainty is problematic can help in improving AI algorithms.

AI & robotics considered early on uncertainty in decision making, where the predominant domain of application was spatial navigation, Cassandra et al. (1996a). In these early applications, routes were planned from a start to a goal state, through heuristic methods which chose paths that balanced the reduction in uncertainty and distance taken to reach the goal. The above navigation problem has typically been treated in two parts: the construction and representation

[3] the human brain project: https://www.humanbrainproject.eu/

of a world model (the map) and a planner which can reason with respect to this model in order to accomplish an objective. The world construction problem attracted a large amount of interest and has resulted in many successfully applications in a wide spectrum of robotic domains (AUV, UAV, etc..). However, the most successful mapping algorithms are well suited to situations in which a direct observation exists between the robot and the features of the map which is being built and the uncertainty originates from Gaussian noise corrupting the measurement. This assumption breaks down in tasks in which mostly negative information is present, that is the absence of sighting of a feature, such as when exploring a dark cave (Figure 1.1 (b)-(c)) or in environments in which landmarks are sparse or if insufficient sensory information is available such as in haptic and tactile searches.

The integration of planning with mapping in a single framework is still difficult to achieve and is based on either representing the decision problem as a Partially Observable Markov Decision Process (POMDP) which is notoriously difficult to solve for large scale problems or by search heuristics.

The main difficulty faced by planners is that the dimensionality of the state space and decision time horizon leads to an unmanageable space and time complexity optimisation problem. Most data driven optimisation methods such as Reinforcement Learning make the strong assumption that simple explorative strategies (white noise) are sufficient to find optimal decision rules in a relatively short time. This assumption is no longer valid in continuous POMDPs when the number of parameters of the policy is quite large. We can take advantage of expert knowledge from human teachers who can provide a set of explorative and exploitative actions so that although the optimisation problem is large there is no need to perform expensive and time consuming autonomous explorations to find an optimal policy.

In summary there are still open problems in decision making when considering partial observability, which originate from both how decisions are planned and how a map is constructed. As both humans and animals are far better at navigation than robots, especially when uncertainty is present, Stankiewicz et al. (2006), we decide to leverage human foresight and reasoning in a Programming by Demonstration (PbD) framework (Billard et al. (2008a)), which we coin PbD-POMDP. PbD examples include the transfer of kinematic task constraints, stiffness and impedance constraints and motion primitives, to name only a few. As for the mapping problem, it has been studied and solved within a certain set of constraining assumptions which do not hold when negative information is present, in the case for haptic and tactile search tasks. For the mapping problem we develop a Bayesian filter which is non-parametric and has no explicit representation of a joint distribution and is not restricted to non-negative information.

In this thesis we address both mapping and planning problems under extreme levels of uncertainty.

1.2 Contribution

In this thesis we bring to light three contributions:

The first contribution is the transfer of human behaviour to robots, by learning a policy extracted from from human demonstrations, in tasks where there is much uncertainty, making them difficult to solve using traditional techniques.

The second contribution is an extension of the first contribution, learning to reason with uncertainty as humans. We added a cost function which we demonstrate can be used to refine and improve an original policy solely learned from human demonstrations without any additional simulation or rollouts, in a Reinforcement Learning framework in belief space.

The third contribution is a non-parametric Bayesian state space filter which is efficient under sparse sensory information and high levels of uncertainty.

Throughout this thesis we consider case studies in which vision is not available, leaving only tactile and haptic information. This choice effectively induces a high level of uncertainty making it easier to study its effect on the decision making process. As a consequence the tasks we consider are by nature, haptic and tactile searches. The following three sections detail the contribution of this thesis to research decision making under severe uncertainty constraints.

1.2.1 Learning to reason with uncertainty as humans

A Markov Decision Process (MDP) allows the formulation of a decision problem in terms of states, actions, a discount factor and a cost function. Given this formulation and a suitable optimisation method (dynamic programming, temporal difference, etc..) a set of optimal decision rules are returned, known as a policy. The benefit of this approach is that the policy is non-myopic and sequences of complicated actions can be synthesised to achieve a goal which an opportunistic policy would fail to achieve. A Partially Observable Markov Decision Process (POMDP) is a generalisation of a MDP to a hidden state space in which the state is only observable through measurements. Finding an exact optimal solution to a POMDP problem is notoriously difficult due to the computational complexities involved. Sample based approaches to solve a POMDP rely heavily on a good trade-off between exploration and exploitation actions.

Good explorative actions increase the chance of discovering a set of optimal decisions/actions.

In this thesis we propose a Programming from Demonstration approach to solving POMDP problems, which we call PbD-POMDP, in haptic and tactile search tasks. Our hypothesis is that if we know the cognitive map of the human expert in terms of his believed location and observe his actions we can learn a statistical policy which mimics his behaviour. Since the human's beliefs are not directly observable we infer them by assuming that the way we integrate evidence is similar to a Bayesian filter. There is evidence both in cognitive and neuroscience that this is the case (Bake et al. (2011)). From observing the expert human performing a task we learn a cognitive model of the human's decision process by learning a generative joint distribution over his beliefs and actions. The generative distribution is then used as a control policy. By this approach we are able to have a policy which can handle uncertainty similarly to humans.

1.2.2 Reinforcement learning in belief space

Learning to search and act as humans and thus reproduce their exploratory behaviour is beneficial in POMDP tasks, since traditional approaches are infeasible. The drawback of the PbD-POMDP approach is that the goal of the task is implicitly encoded in the demonstrations performed by the teacher. To be successfully, it is usually a requirement that the teacher is be an expert, with a few notable exceptions, Rai et al. (2013). As a result the quality of the learned behaviour depends on the skill and embodiment constraints of the human. Since we are solely learning a PbD-POMDP statistical controller, both good and bad demonstrations are mixed in together. By introducing a cost function representing the task, we can explicitly obtain a quality metric of the provided demonstrations. In this way we can optimise the parameters of our generative model to maximise the cost function.

Reinforcement learning (RL) is a framework which allows, through repeated interaction with the environment, to learn an optimal policy for a task. There are many variants of RL, but all rely on simple exploration strategies to find the optimal behaviour. These explorative strategies prohibit the application of RL to large and continuous POMDP settings in which the policy is comprised of many parameters. In our previous contribution we showed that it is feasible to learn and extract multiple search strategies from human demonstrations and in a sense we have already solved the exploration/exploitation dilemma which plagues reinforcement learning applications.

We propose a Reinforcement Learning framework for the task of searching and connecting a power plug to a socket, with only haptic information. A set of human teachers demonstrate the task from which we record and build a statistical controller. With the same data we learn a belief space value function which we use to update the parameters of the original statistical controller. In this RL-PbD-POMDP setup a very simple cost function provides a significant policy improvement.

1.2.3 Non-parametric Bayesian state space filter

In both previous contributions we considered searches which can be categorised as localisation problems. In localisation problems the map of the environment is considered to be known while the position of the agent is unknown. There is a wide range of applications for localisation but there are also cases in which both the map and the agent's position is unknown. This kind of problem is known as Simultaneous Localisation and Mapping (SLAM).

SLAM is concerned with the development of filters to accurately and efficiently infer the state parameters of an agent (position, orientation) and aspects of its environment, commonly referred to as the map. It is necessary for the agent to achieve situatedness which is a precondition to planning and reasoning. The predominant assumption in most applications of SLAM algorithms is that uncertainty is related to the noise in the sensor measurements. In our haptic search tasks there is no visual information and a very large amount of uncertainty. Most of the sensory feedback is negative information, a term used to denote the non-event of a sensory response. In the absence of recurrent sightings or direct measurements of objects there are no correlations from the measurement errors which can be exploited.

In this thesis we propose a new SLAM filter, which we name Measurement Likelihood Memory Filter (MLMF), in which no assumptions are made with respect to the shape of the uncertainty (it can be Gaussian, multi-modal, uniform, etc..) and motion noise and we adopt a histogram parametrisation. The conceptual difference between the MLMF and standard SLAM filters, such as the Extended Kalman Filter (EKF), is that we avoid representing the joint distribution since it would entail a unfathomable space and time complexity. This is achieved by keeping track of the history of measurement likelihood functions. We demonstrate that our approach gives the same filtered marginals as a histogram filter. In such a way we achieve a Bayes filter which has both linear space and time complexity. This filter is well suited to tasks in which the landmarks are not directly observable.

1.3 Thesis outline

The thesis is structured according to three main contributions outlined in the previous section, each comprising a chapter and the following paragraphs give a detailed outline of the structure of this thesis, see Figure 1.2.

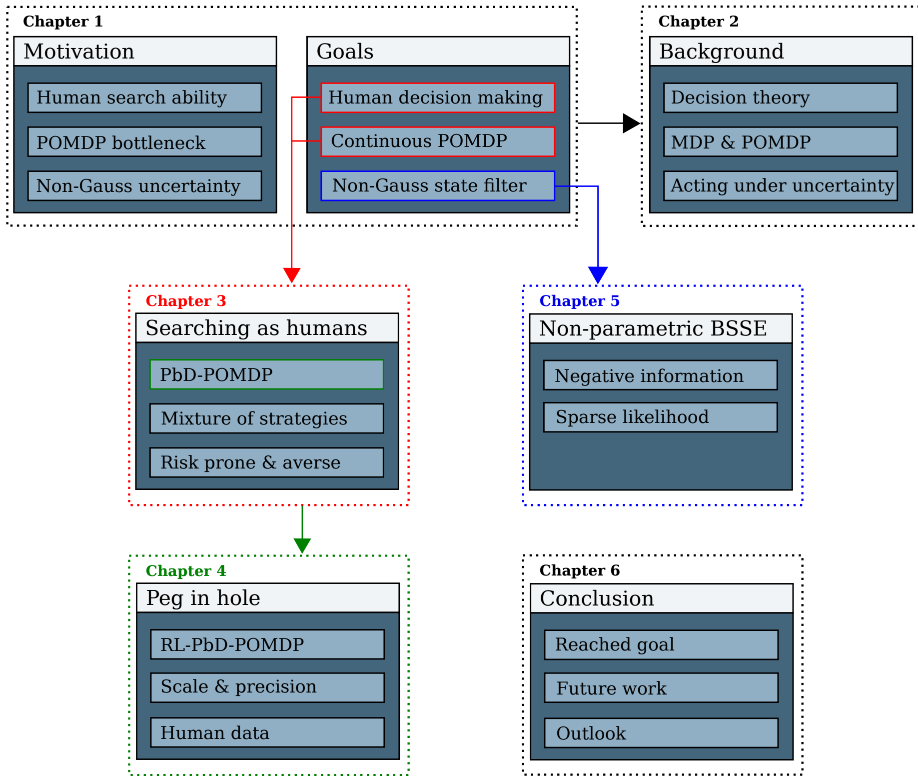


Figure 1.2: Roadmap of the Thesis with key points.

Chapter 2 - Background

In this chapter we introduce and mathematically formalise the sequential decision making problem under uncertainty and we provide a detailed literature review of the related work in this domain. We provide a brief introduction to *Decision Theory* before focusing on the work in AI & robotics relevant to POMDPs whilst highlighting their relevance and contribution to our work.

Chapter 3 - Learning to reason with uncertainty as humans

In this chapter we present an approach for transferring human skills in a blind haptic search task to a robot in our PbD-POMDP framework. The belief of the human is represented by a particle filter and all subsequent beliefs are inferred from the human's motions acquired via a motion tracking system. A generative model of the joint belief and actions distribution is learned and used to reproduce the behaviour on a WAM and KUKA robot in two search tasks. Experimental evaluations showed the approach to be superior to greedy opportunistic policies and traditional path planning algorithms. We also provide a review of work related to humans taking decisions under uncertainty in spatial navigation and haptic tasks with an emphasis on works which consider diminished or no visual information.

Chapter 4 - Reinforcement learning in belief space

In this chapter we present a similar approach to the one in chapter 3, "Learning to reason with uncertainty as humans", with the difference that we explicitly encode the task through the introduction of a binary objective function and we consider a peg-in-hole task under high levels of uncertainty. The task requires both high and low levels of precision from the agent to be able to accomplish it, which makes it particularly interesting. We demonstrate the importance of initially provided human data as opposed to using data generated from a myopic policy. We learn a value function approximation of the belief space through locally weighted regression and approximate dynamical programming. By combining a PbD approach in this Actor-critic Reinforcement Learning framework, we demonstrate an improvement upon a purely statistical controller with nearly no additional cost. We refer to this approach as RL-PbD-POMDP.

Chapter 5 - Non-parametric Bayesian state space filter

In this chapter we present an approach to perform a state space estimation of a map and agent given that there is no direct observation between the landmarks and the agent. We demonstrate that by keeping track of the applied measurement functions rather than explicitly parametrizing the full joint distribution of the landmarks and agent we can fully reconstruct the optimal Bayesian state estimation. The advantage of our approach is that the space complexity is linear as opposed to exponential. We validate our approach in 2D search navigation tasks. We also give an overview of the literature of SLAM and emphasis the position of our filter within it.

Chapter 6 - Conclusion

We conclude by providing a holistic summary of our work and achievements. We draw attention to the current open problems and directions for future work in the field of uncertainty and reasoning in Artificial intelligence and robotics.

Chapter 2

Background

Acting under uncertainty is central to AI and robotics and has been an active area of research for decades. It is an umbrella term which encompasses a wide spectrum of fields: *economics*, *psychology*, *cognitive science*, *neuroscience*, *robotics* and *artificial intelligence*. The work in this thesis relies on results from all of the aforementioned fields with varying degree. Cognitive and neuroscience bring justification and biological inspiration in the way we represent our beliefs and how we act accordingly. AI and robotics provide computational models and optimisation methods some of which are biologically inspired to be able to solve decision problems given uncertainty. Because of the vast spectrum of topics we cannot do justice to all them and we will focus on works which are directly relevant to the problems we are addressing in this thesis, *which is how to teach a robotic apprentice to act under uncertainty.* In this chapter we cover the following topics in the presented order: Decision Theory (DT), Markov Decision Process (MDP), Partially Observable Markov Decision Process (POMDP), a literature review and the approach taken in this thesis.

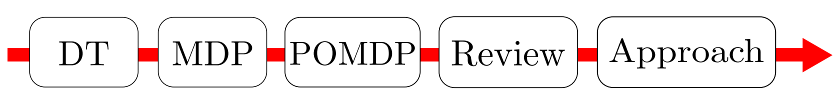


Figure 2.1: Chapter outline.

- **Section 2.1**, introduces what is meant by taking decisions under uncertainty and what are the different sources of uncertainty. We take a historical look at Decision Theory since it is the root node of all subsequent research in reasoning and acting under uncertainty and provides for a good introduction to the topics which will follow.

- **Section 2.2**, mathematically formalises the sequential decision problem under uncertainty and is linked with Decision Theory. We derive from first principal the Bellman optimal equation which is one of the most important result to date in sequential decision processes.

- **Section 2.3**, provides an in depth literature review with the latest results in AI & robotics in the subject of planning and acting under uncertainty.

We draw attention to the different approaches to solving this problem whilst pointing out their advantages and weaknesses. We summaries what has been achieved so fare and what are the open problems.

- **Section** 2.4, the core approach taken by this thesis is detailed. We outline how we teach a robotic apprentice to act under uncertainty.

2.1 Decisions under uncertainty

The main objective of reasoning under uncertainty is to find an action or sequence of actions which will result in the most preferable outcome. There are two key attributes which can render this problem difficult: **stochastic actions** and **latent states**.

Stochastic actions when applied in the same state will not always result in the same outcome. This type of uncertainty can arise from many sources. For instance, the outcome of chaotic systems will always lead to different results when the same action is applied to the same initial conditions, such as the throwing of a dice or the flipping of a coin. In outdoor robotics the terrain might lead to slippage, causing the robot to skid, or in an underwater environment currents might drastically offset the position of an UAV. In articulated robots, the friction in the joints can result in an error in the end-effector position (especially true for cable driven robots).

The second source of uncertainty is when the state space cannot be determined. This arises when the sensors are not able to provide sufficient information to reliably estimate the state. In robotics this uncertainty can arise from inadequate or noisy sensors. In poor environmental conditions such as humidity, lack of light or smoke the robot can experience difficulties in ascertaining its position and thus in planning how to achieve a given objective.

Given these two types of uncertainty, the question is how to represent these uncertainties. The predominant approach is to quantify the uncertainty in terms of probabilities. For instance the application of a forward action to a wheeled robot will result in some probability in a new position further ahead and with a remaining probability distributed to adjacent regions which might have occurred due to slippage.

An observation made through the robot's sensors will result in a probability distribution over the robot's probable location. This quantification of the action and observation uncertainty, in terms of a probability distribution over the state, must be utilised by the agent to plan actions towards accomplishing its goal. In order to take a decision, the agent must assign a utility to each state weighted by the probability of its outcome and act so as to get the highest utility. The utility indicates a preference over the outcomes and when combined with probabilities leads to Decision Theory, which is the topic of the next section.

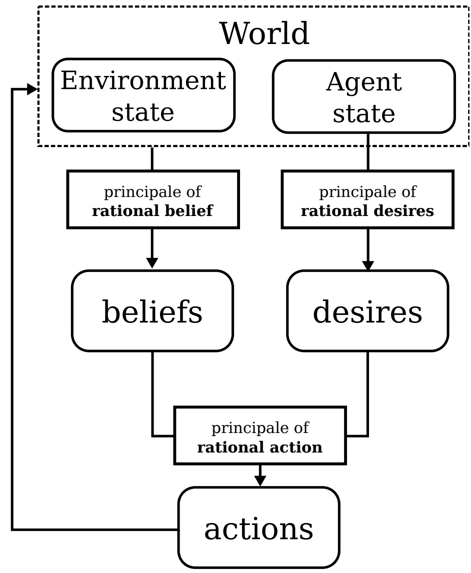


Figure 2.2: Relation between beliefs, desires and actions and are all considered to be rational.

2.1.1 Decision theory

The central question that Decision Theory asks is: *how do we take decisions when faced with uncertain outcomes ?* To answer such a question we need to identify the attributes which are involved when we take a decision, namely our **beliefs** and **desires**. Beliefs reflect a degree of knowledge we have about the world. This degree is ascertained by the amount of evidence we have in support of our beliefs. Epistemology studies in great detail the relationship between truth, beliefs and knowledge. We will not go into a philosophical discussion of their interplay, but make use of the following: if we have sufficient evidence in support of our beliefs and they represent the truth then we consider them to be **rational beliefs**. As for desires, they are linked to our disposition to take upon them. For example if I want to switch off my alarm clock I have to look for it in the last area I believed it to be in. These two attributes, beliefs and desires, are used to frame a decision problem. Early work in decision theory assumed that the problem was well grounded and focused on finding what **rational actions** need to be taken given our beliefs in order to achieve our desires.

Early interest in such questions were typically centred around economics which included deciding an appropriate investment or wager for a particular gamble. It was noted that the expected monetary outcome of a gamble as a means of basing a decision, would often lead to a course of action which contradicts common sense. A famous example of this contradiction is demonstrated in the St. Petersburg paradox. In this paradox a bookmaker proposes the following gamble. An initial pot starts with a content of £2. The bookmaker proceeds to flip a fair coin until the first appearance of a tails which ends the game. Until the occurrence of the first tails the money in the pot doubles after every toss. Once the game ends the player leaves with the contents of the pot. As an avid gambler and **expected value** maximiser how much would one be willing to pay to enter this game ? To access, one would need to know the average payout. The amount of money increases by £2^n, where n is the number of non-final tosses and the probability of reaching n is $1/2^n$. In this case the expected monetary outcome is infinite:

$$\mathop{\mathbb{E}}_{p(\pounds)}\{\pounds\} = \underbrace{\frac{1}{2}\pounds 2}_{\text{first toss}} + \frac{1}{4}\pounds 4 + \cdots = \sum_{n=1}^{\infty} \pounds \frac{2^n}{2^n} = \pounds\infty$$

So the expected gain or return for paying to enter such game is an infinite amount of money. Thus in principal if a player was seeking to maximise his expected return value he would be willing to pay an amount close to infinity to enter the game. This does not seem a good decision rule; no person in the world would be willing to pay such high amounts to enter this game.

Nicolas Bernoulli proposed a solution to the problem which was later published by his brother Daniel (republished Bernoulli (1954)). He introduced the notion of a **utility function**, and he claimed that people should base their decision on the expected utility instead of solely on the monetary outcome.

> "...the value of an item must not be based on its price, but rather on the utility it yields."
>
> — Daniel Bernoulli

The introduction of a utility function takes into account that the net worth of a person will influence their decision since different people (in terms of their monetary worth) will weigh the gain differently. The utility function introduced by Bernoulli was the logarithm of the monetary outcome $x \in X$ weighted by its probability $p(x)$ which results in an expected utility:

$$U(x) = \mathbb{E}\{u(x)\} = \sum_{x \in X} p(x) \underbrace{\log(x)}_{u(x)}$$

It is later in 1944 that von Neumann and Morgenstern (Von Neumann and Morgenstern (1990)) axiomised Bernoulli's utility function and proved that if a decision maker has a preference over a set of lotteries[1] which satisfy four axioms (completeness, transitivity, continuity, independence) then there exists a utility function whose expectation preserves this preference. An agent whose decisions can be shown to maximise the vNM expected utility are said to be **rational** otherwise they are **irrational**.

This is the theoretical basis of most economic theory. It is a **normative** model of how people should behave given uncertainty. It is also the basis of

[1]the term lottery refers to a probability distribution in the original text.

most if not all decision making, cogitative architectures and control policies in AI and robotics (to the best of the author's knowledge).

An aspect to keep in mind regarding the vNM model is that it is normative; it states what should be a rational decision. As a result it is not always consistent with human behaviour. There is great debate regarding the predictions made by vNM models with respect to our behaviour. There have been many studies both demonstrating divergence between the model's predictions and our observed behaviour but also supporting evidence that it does reflect the output of our decision making process. Reasons for divergence have been attributed to how people weigh probabilities and how the decision problem is framed. But probably the most important aspect is that in most decisions we are faced with, the quantification and rationality of our beliefs might not be adequate and limitations of our working memory will come into play in the final decision.

Nevertheless vNM agents are predominantly used in AI and robotics as a means of implementing decision making processes or in control policies. In psychology and cogitative science vNM agents are a used for comparing human behaviour against an optimal strategy (by optimal we mean it is rational in the vNM sense). It is important to remember the origins and assumptions underlying the models that are used to represent control policies or cognitive architectures implemented into robotic systems or software agents.

2.2 Sequential decision making

When Decision Theory is brought up, we are usually referring to a one shot non-temporal decision. However many interesting decision problems are sequential. In such situations, we must consider the effect current decisions will have on future decisions. Expected utility theory (part of Decision Theory) is extendable to a temporal decision problem. There are however two subtle but important differences between the temporal and non-temporal decision problems. The first difference is the utility. In the one time step problem an outcome has one utility assigned to it, $u(x)$. In the temporal decision problem a utility has to be assigned to a sequence of outcomes, $u(x_{0:T})$, where T is the number of sequential decisions taken. The utility of a sequence is the sum of the individual utilities. However if the decision problem is non terminating this will lead to an unbounded utility. To bound the utility a discount factor $\gamma \in [0, 1)$ is introduced and the new temporal utility function becomes:

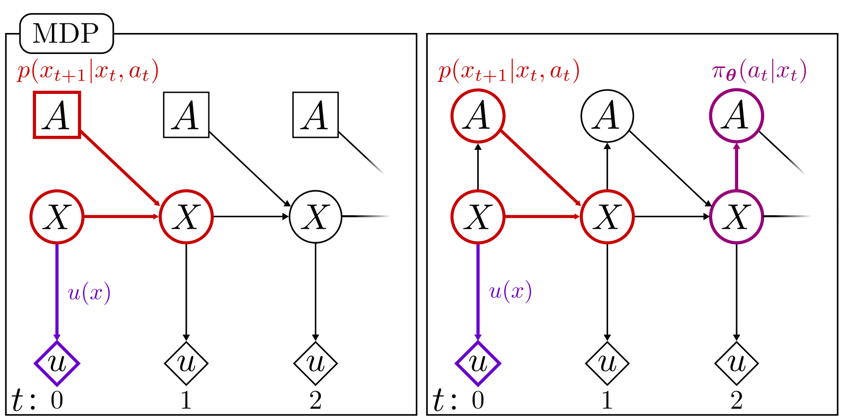
$$u(x_{0:T}) := \sum_{t=0}^{T} \gamma^t u(x_t) \tag{2.1}$$

The discount factor controls the importance that later utilities have on the final utility. If the discount factor is set to zero we obtain the original one shot

Notation	Definitions
$x_t \in \mathbb{R}^3$	Cartesian state space position of the agent end-effector.
$y_t \in \mathbb{R}^M$	Observation/measurement from the agents sensors.
$a_t \in \mathbb{R}^3$	Action, Cartesian velocity of the end-effector of the agent.
X, Y, A	State, observation and action random variables where x, y and a are realisation.
$p(x_t)$	Short hand notation for a probability density function, $p_X(x_t)$.
$x_{0:t}$	$\{x_0, x_1, \cdots, x_{t-1}, x_t\}$, history up to time t.
$p(x_t \mid y_{0:t}, a_{0:t})$	Filtered probability distribution over the state space given the action and observation history.
b_t	Belief state is the filtered state space distribution $b_t = p(x_t \mid y_{0:t}, a_{0:t})$ which will be written as b_t for simplicity.
$\pi_{\boldsymbol{\theta}}(a_t \mid \cdot)$	Parametric probabilistic policy, $a_t \sim \pi_{\boldsymbol{\theta}}(a_t \mid \cdot)$, where $\boldsymbol{\theta}$ is the parameters.
$u(x) \in \mathbb{R}$	Utility function, returns the utility of being in state x. It can also be dependent on the action, $u(x, a)$.
$\gamma \in [0, 1)$	Discount factor, the closer to one the more the later utilities are considered. When set to zero, only immediate utilities are considered which would result in a myopic greedy agent.
$p(x_{t+1} \mid x_t, a_t)$	State transition model, returns the likelihood/probability of reaching state x_{t+1} given that action a_t is applied in state x_t.
$p(y_t \mid x_t)$	Observation/measurement model, returns the likelihood/probability of observing y_t given that the agent is in state x_t.
$\tau(b_t, u_t, y_t)$	Updates a belief given a motion and observation. It makes use of both the motion and observation functions. The state space estimation function, τ, can be any kind of state space filter such as an Extended Kalman Filter (EKF) or a Particle Filter (PF).

Table 2.1: Definition of common variables used.

utility function and if we were to take actions which maximised the expected utility we would not be considering at all the effect current decisions have at future decision points. An agent reasoning in such a way is called **myopic**. The second difference between the temporal and non-temporal decision problem is the way in which probabilities are assigned to outcomes. This was $p(x)$ in the Decision Theory utility function formulation. Now because of the sequential nature of the problem we consider a conditional state transfer probability distribution $p(x_{t+1}|x_t, a_t)$ which models the probability of going from state x_t to x_{t+1} given that action a_t is taken. This particular representation of a sequential decision problem is called a **Markov Decision Process (MDP)** and to be more exact a first order MDP. The necessary models are the state transition and utility functions. The assumption of such a model is that all necessary information to take a decision is encoded in the current state and there is no need to consider the history of state transitions when taking a current decision. In Figure 2.3 we illustrate two graphical representations of a MDP, which are known as **Dynamic Bayesian Networks (DBN)**. A DBN represents the temporal relationship and conditional dependence between random variables, decisions and utilities, which are represented by circles, squares and diamonds. For the MDP to the left the actions are not stochastic, whilst for the MDP on the right the actions taken are governed by a stochastic **policy**, $\pi_{\boldsymbol{\theta}}(a_t|x_t)$. A policy represents the plan of an agent for each state, given a state it will output an action. A policy is considered optimal when it maximises the expected utility function, it is optimal in the vNM sense.



(a) off-policy **(b)** on-policy

Figure 2.3: Dynamical Bayesian Network of a Markov Decision Process; it encodes the temporal relation between the random variables (circles), utilities (diamond) and decisions (squares). The arrows specify conditional distributions. In **(a)** the decision nodes are not considered random variables whilst in **(b)** they are. From these two DBN we can read off two conditional distributions, the state transition distribution (in red) and the action distribution (in purple).

Solving a MDP means finding a policy whose actions in any given state will always maximise the expected utility. Such a policy is usually denoted as π^*, the **optimal policy**. As in decision theory, the expected utility is the utility of a sequence of states $u(x_{0:T})$ weighted by its probability. The graphical representation (Figure 2.3 (a)) allows the probability of a sequence of states and actions, to be read off directly:

$$p(x_{0:T}, a_{0:T-1}) = p(x_0) \prod_{t=0}^{T-1} p(x_{t+1}|x_t, a_t) \tag{2.2}$$

$$u(x_{0:T}) = u(x_0) + \gamma u(x_1) + \cdots + \gamma^{T-1} u(x_{T-1}) + \gamma^T u(x_T) \tag{2.3}$$

We are interested in finding the sequence of actions, $a_{0:T}$, which will maximise the expected utility function:

$$\operatorname*{argmax}_{a_{0:T-1}} U(x_{0:T}, a_{0:T-1}) = \max_{a_0} \sum_{x_1} \cdots \max_{a_{T-1}} \sum_{x_T} \left(p(x_{0:T}, a_{0:T-1}) u(x_{0:T}) \right) \tag{2.4}$$

Solving the above directly in its current form would to lead to an exponential complexity. Making use of the first order Markov assumption and that current utilities do not depend on future utilities, the summations can be re-arranged and a recursive pattern emerges which can be exploited:

$$\begin{aligned} \operatorname*{argmax}_{a_{0:T-1}} U(x_{0:T}, a_{0:T-1}) &= \max_{a_0} \sum_{x_1} \cdots \max_{a_{T-2}} \sum_{x_{T-1}} p(x_{0:T-1}, a_{0:T-2}) \\ &\left(u(x_{0:T-2}) + \gamma^{T-1} \left(u(x_{T-1}) + \gamma \max_{a_{T-1}} \sum_{x_T} p(x_T|x_{T-1}, a_{T-1}) u(x_T) \right) \right) \end{aligned} \tag{2.5}$$

From the rearrangement we notice that Equation 2.5 has the same functional form as Equation 2.4, except that the recursive component can be summarised by Equation 2.6, which is known as the **Bellman** optimal equation (the asterisk indicating that it is optimal),

$$V^*(x_t) := u(x_t) + \gamma \max_{a_t} \sum_{x_{t+1}} p(x_{t+1}|x_t, a_t) V(x_{t+1}) \tag{2.6}$$

where for the terminal state $V_T(x_T) = u(x_T)$. The Bellman equation is a means of solving a sequential decision problem through use of dynamic programming. It shows that the utility of the current state is based on the immediate utility and the discounted maximum utility of the next state. Making use of this recursion reduces the computation complexity which is now quadratic in the number of states, $\mathcal{O}(T\,|A|\,|X|^2)$. To find the optimal value and subsequent policy an approach would be to repeatedly apply the Bellman equation to each state until the value function converges. What makes the problem difficult to solve is maximisation over the actions. This induces two problems, the first is that the optimisation is nonlinear and the second is that if the action space is continuous the maximisation will be expensive to compute. This brings into play the two main approaches to solving a MDP: **off-policy** and **on-policy**. Off-policy methods solve directly for the optimal value function, $V^*(x)$, and perform the maximisation over the actions. **Value-Iteration (VI)** is such a method. On-policy approaches, $V^{\pi}(x)$, find the optimal value and policy through repeating **policy evaluation** and **improvement** steps. In the policy evaluation the value or utility of a policy is found through solving the on-policy version of the Bellman equation:

$$V^{\pi}(x_t) := u(x_t) + \gamma \sum_{a_t} \pi_{\boldsymbol{\theta}}(a_t|x_t) \sum_{x_{t+1}} p(x_{t+1}|x_t, a_t) V(x_{t+1}) \tag{2.7}$$

In the policy improvement step, the policy is made more greedy by maximising the value function. Through the repetition of these two steps both the value function and policy converge to the optimal. On-policy methods are preferred in settings where the action space is highly continuous, such as in robotics. Using dynamic programming is however not the method of choice since it requires multiple passes through the entire state space and for this reason it is necessary to have the model of the state transition a priori. Instead **Reinforcement Learning (RL)** methods are used to find an optimal value and policy. RL is a sample based approach in which an agent interacts with the environment gathering examples of state transitions and the utility and uses them to gradually solve the Bellman equation.

We introduced the formulation of a sequential decision process for the MDP model and showed how an optimal policy and value function are obtained through maximising the expected utility. The re-arrangement of the summations, known as variable elimination, allows to exploits a recursive structure present in the Markov chain. The recursive component turns out to be the Bellman optimal equation, which when solved (via dynamic programming or reinforcement learning) results in an optimal value and policy function. A MDP models the uncertainty inherent in the state transition but not the uncertainty of the state. The MDP assumes that the state space is always fully observable, which is a strong assumption. In robotics, the on board sensors return an estimate of the state with a certain amount of uncertainty associated with it. To take this additional uncertainty into consideration the MDP has to accommodate it. This leads to a Partially Observable Markov Decision Process (POMDP).

2.2.1 POMDP

A POMDP is a popular approach for formulating a sequential decision process in which both motion and observation uncertainty are considered. In this partially observable setting the agent does not know with exactitude the state of the environment, but is able to observe it through his **sensors**. We define a sensor mathematically as being a function of the state space, x_t, relating to an observation, y_t, corrupted by some noise, ϵ_t,

$$y_t = h(x_t) + \epsilon_t \tag{2.8}$$

The sensor function $h(x_t)$ can be linear or non-linear and the additive noise term ϵ_t can be Gaussian (usually the case), non-Gaussian, state dependent or not. The uncertainty of the latent state, x_t, is quantified by a probability distribution, $p(x)$. This probability distribution represents all the hypothetical positions in the world in which the agent can be found. In Figure 2.4 **(a)** an agent is located in a square yard containing a wall. Initially the agent is confident of his position; his state uncertainty $p(x_0)$ is low, represented by the blue probability density. However during a circular displacement the agent skids and the state uncertainty is increased by the state transition function, $p(x_{t+1}|x_t, a_t)$; this step is referred to as **motion update**. To reduce the uncertainty, the agent takes a measurement, y_t, with his sensors which provide range, r, and bearing, ϕ, information with respect to the wall, see Figure 2.4 **(b)**. The agent uses the model of his sensor, known a priori, to deduce all possible locations in the world from where the current measurement could have originated. This model is known as the measurement likelihood function:

$$p(y_t|x_t) = \mathcal{N}(y_t - h(x_t); 0, \Sigma) \tag{2.9}$$

The measurement likelihood function makes use of the measurement function $h(x)$ and it models the noise in the sensor. In this case the noise model, ϵ_t, is Gaussian, paramaterized with mean zero and covariance Σ. Typically the parameters of the measurement likelihood function are learned a priori.

In Figure 2.4 **(c)** the likelihood is illustrated. The dark regions indicate areas of high likelihood, which are possible locations from which the sensor measurement could have originated. The value of the measurement likelihood function is then integrated into the state space probability density function; this step is referred to as **measurement update**.

The two update steps, motion and measurement, are part of a recursive state estimation process called a **Bayesian state space filter**, which we formalise below in Equation 2.10-2.11.

The motion model, Equation 2.10, updates the position of the probability distribution according to the applied action, a_t, and adds uncertainty by increasing the spread of the distribution. The measurement information is then incorporated by Equation 2.11. The measurement likelihood always reduces the uncertainty or leaves it constant. The Bayesian state space filter is such an important component to belief space decision making that we define it by the filter function, $\tau(b_t, a_t, y_t)$, which takes as input the current belief, applied action and sensed measurement and returns the resulting belief b_{t+1}. The state space filter is an essential component to a POMDP which will become apparent later.

With the latent state, its relation to the observation variable and the Bayesian filter defined, we can introduce the POMDP model in Figure 2.6 (*left*). It has

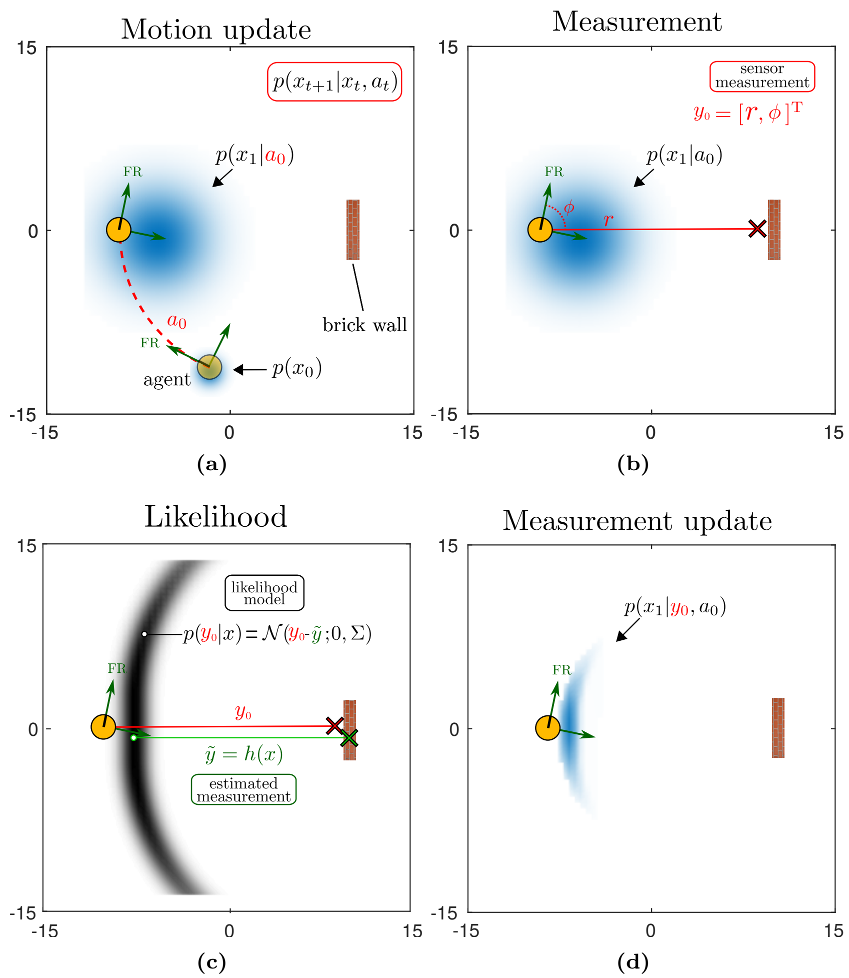


Figure 2.4: **(a)** An agent is located to the south west of a brick wall. It is equipped with a range sensor. The agent takes a forward action but skids, which results in a high increase of the uncertainty. **(b)** The agent takes a measurement, y_0, of this distance to the wall; because his sensor is noisy his estimate is inaccurate. **(c)** The agent uses his measurement model to evaluate the plausibility of all locations in the world which would result in a similar measurement; illustrated by the likelihood function $p(y_0|x_0)$. **(d)** The likelihood is integrated into the probability density function; $p(x_0|y_0) \propto p(y_0|x)p(x_0)$.

Bayesian filter

The Bayesian filter turns a prior probability distribution over the state space, $p(x_{t-1}|y_{0:t-1}, a_{0:t-1})$, to a posterior $p(x_t|y_{0:t}, a_{0:t})$ by incorporating both motion and measurement. Applied recursively it keep a probability distribution over the state space which considers all the past history of actions and observations. We define the application of these two steps by the filter function τ, which takes the current belief, the applied action and measurement, and outputs the next belief, b_{t+1}.

Motion update

$$p(x_t|y_{0:t-1}, a_{0:t}) = \int p(x_t|x_{t-1}, a_t)\, p(x_{t-1}|y_{0:t-1}, a_{0:t-1})\; dx_{t-1} \quad (2.10)$$

Measurement update

$$p(x_t|y_{0:t}, a_{0:t}) = \frac{1}{p(y_t|y_{0:t-1}, a_{0:t})} p(y_t|x_t)\, p(x_t|y_{0:t-1}, a_{0:t}) \quad (2.11)$$

$$p(y_t|y_{0:t-1}, a_{0:t}) = \int p(y_t|x_t)\, p(x_t|y_{0:t-1}, a_{0:t}) dx_t \quad (2.12)$$

Filter function

$$b_{t+1} := \tau(b_t, a_t, y_t) \quad (2.13)$$

Figure 2.5: Bayesian state space filter.

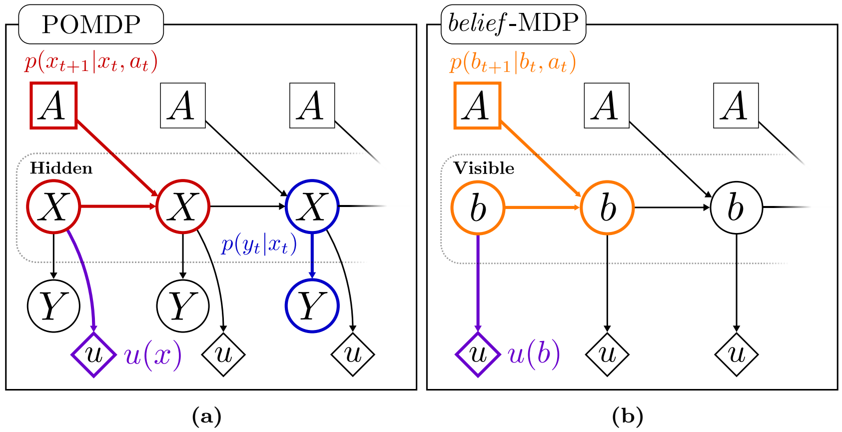


Figure 2.6: **(a)** POMDP graphical model. The state space, X, is hidden, but is still partially observable through a measurement, Y. **(b)** The POMDP is cast into a belief Markov Decision Process, belief-MDP. The state space is a probability distribution, $b(x_t) = p(x_t)$, (known as a belief state) and is no longer considered a latent state. The original state transition function $p(x_{t+1}|x_t, a_t)$ is replaced by a belief state transition, $p(b_{t+1}|b_t, a_t)$. The reward is now a function of the belief.

the same Markov chain structure as the MDP, introduced in the previous section, but the state space X is latent and a new layer of observation variables Y is added.

As the state space is only partially observable the expected utility has to be computed for each possible history of states, actions and observations. All approaches in the literature instead encapsulate all these possible histories into a belief state $b(x_t)$ (for short notation b_t) which is a probability distribution (also referred to as an information state, I-state) over the state space x_t and use this new state description to cast the POMDP into a **belief-MDP** (states are probability distributions, beliefs). By casting a POMDP into a *belief*-MDP the state space is considered observable and we recover the same structure as in the standard MDP problem.

As we are working within a belief space the reward function has to be adapted to:

$$u(b_t) = \sum_{x_t} u(x_t)\, b(x_t) = \mathbb{E}_{b_t}\{u(x_t)\} \tag{2.14}$$

which is an expectation. The goal as before is to find a sequence of actions which will maximise the expected utility. Since our *belief*-MDP has the same structural form as the MDP, the solution to the problem is the same Bellman equation derived previously. We just substitute the new belief transition function and we get the corresponding belief Bellman Equation, 2.15.

$$V^*(b_t) = u(b_t) + \gamma \max_{a_t} \sum_{b_{t+1}} p(b_{t+1}|b_t, a_t)\, V^*(b_{t+1}) \tag{2.15}$$

However, using this equation in this form is problematic, as we are summing over the space of beliefs (which is high dimensional and infinite for the continuous case) and the transition function is a probability distribution over beliefs. The key to overcome this problem is to realise that if we know what the current measurement and applied action are, there is only one valid possible belief, b_{t+1}, and the summation over beliefs vanishes. This can be seen by substituting the belief transition function, Equation 2.16, into the Bellman equation Equation 2.15.

$$p(b_{t+1}|b_t, a_t) = \sum_{y_t} p(b_{t+1}|b_t, a_t, y_t)\, p(y_t|y_{0:t-1}, a_{0:t}) \tag{2.16}$$

After the substitution and re-arrangement of the summation we get Equation 2.17. Since the observation is known (because the outer summation is over y_t), the summation over the beliefs vanishes since there is only one possible future belief which is given by the Bayesian filter function $b_{t+1} = \tau(b_t, a_t, y_t)$,

$$u(b_t) + \gamma \max_{a_t} \sum_{y_t} \underbrace{\left(\sum_{b_{t+1}} p(b_{t+1}|b_t, a_t, y_t)\, V^*(b_{t+1}) \right)}_{1 \cdot V^*(\tau(b_t, a_t, y_t))} p(y_t|y_{0:t-1}, a_{0:t}) \tag{2.17}$$

which simplifies to:

$$\begin{aligned} V^*(b_t) &= u(b_t) + \gamma \max_{a_t} \sum_{y_t} p(y_t|y_{0:t-1}, a_{0:t})\, V^*(\tau(b_t, a_t, y_t)) \\ &= u(b_t) + \gamma \max_{a_t} \mathop{\mathbb{E}}_{y_t}\{V^*(b_{t+1})\} \end{aligned} \tag{2.18}$$

The belief Bellman equation is intuitive. The value of the current belief is the immediate utility plus the value of the future belief states weighted by the probability of a measurement which would result in these future belief states. An exact solution exists only when considering a finite state, action and observation space and a finite planning horizon T, Richard D. Smallwood (1973). The belief-MDP can be solved with value iteration but each backup operation (application of the bellman equation) results in an exponential growth in the number of parameters needed to represent the value function, which is computationally intractable.

Most early techniques for solving POMDPs used value iteration. The preference for persisting in doing this, given the computational burden, is that since the utility function uses a linear operator (the expectation) and that the Bellman backup operation (applying the Bellman equation to the current value function) preservers the linearity, the value function after each updates is Piece Wise Linear and Convex (PWLC). A good text on the implementation of exact value iteration for POMDPs can be found in (Thrun et al., 2005, Chap. 15) and Kaelbling et al. (1998).

In summary there are two problems in solving a POMDP:

- **curse of dimensionality**: A discrete state space of size N will result in a belief space of dimension $N-1$. The discretization choice will greatly impact the computational cost of Value Iteration.

- **curse of history**: The space and computational complexity in the worst case is exponential with respect to the planning horizon, T, Du et al. (2010).

Given such complexity it is hard to see POMDPs being actually usable for real world scenarios. As a result many approximate techniques have emerged with some being very successful. In the next section, we survey the literature and the developments of approximate POMDP algorithms and their applications.

2.3 Literature review

We review the latest methods on *Acting under uncertainty*. This is an extremely dense and spread out area of research, no doubt because of its importance. If uncertainty is not considered adequately, the control policy risks being suboptimal or lead to drastic failure. We will focus the review four subsections in the following order:

- 2.3.1 Value Iteration (VI)
- 2.3.2 Policy Search
- 2.3.3 Planning
- 2.3.4 Heuristics

with an emphasis on robotic applications. In Figure 2.7 we illustrate graphically these four topics which their associated sub-fields.

2.3.1 Value Iteration

The POMDP formulation introduced previously is the main theoretical starting point of policies which consider uncertainty optimally. However solving an exact POMDP through dynamic programming (value iteration) is computationally intractable and an exact solution only exists for discrete state, action and observation space (Thrun et al., 2005, Chap. 15). This intractability, in which only problems with a few states could be solved has inhibited the application of the POMDP framework to robotics.

Point-base Value Iteration

The first breakthrough of the application of VI in belief space to a robotic application was Point-Base Value Iteration (PBVI) Pineau et al. (2003). It

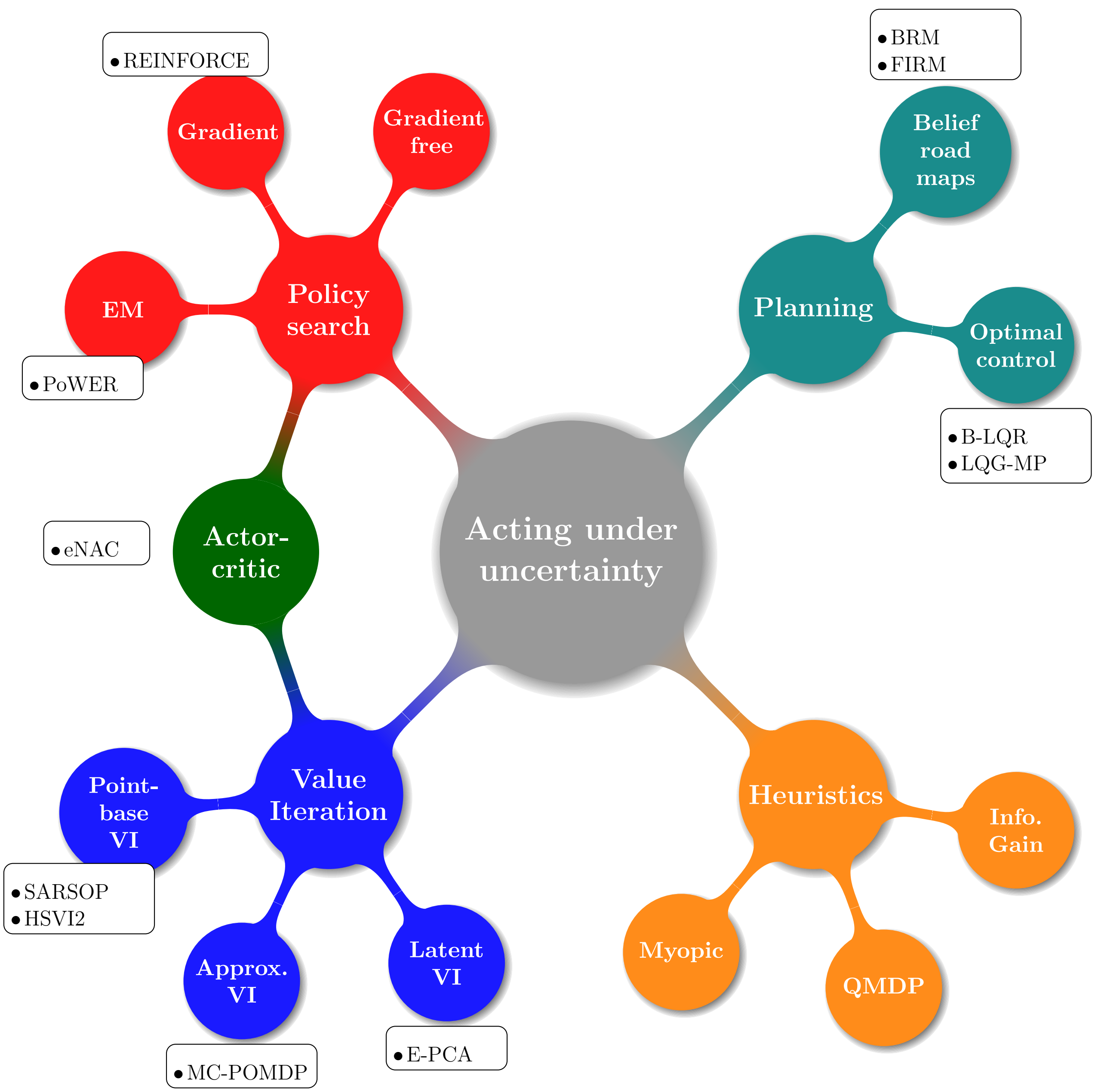


Figure 2.7: Mind-map of AI and robotic methods for acting under uncertainty.

allowed VI to be applied to a robotic navigation problem consisting of 626 states in a hospital patient search task. The key insight to scale VI was to only consider a subset of belief states which were reachable and relevant to the problem. This is achieved by smart sampling techniques and only performing VI backups on beliefs states which are relevant. From this point most research has focused on determining efficient strategies to sample belief points and on which to apply VI. Heuristic Search Value Iteration (HSVI1) (Smith and Simmons (2004)) and HSVI2 (Smith and Simmons (2012)) use forward search heuristics to find relevant beliefs by keeping a lower and upper bound on the current estimated value function. The belief tree is expanded by choosing an action and observation with relation to the potential future effect on the value of the bounds, which are being minimised. HSVI has a comparable performance with respect to classical PBVI except in the game of tag (a benchmark problem), in which it fairs significantly better. A method developed after HSVI, named Forward Search Value Iteration (FSVI) (Veloso (2007)) takes an alternative approach to keeping an upper and lower bound on the value function, as in HSVI, since doing so results in a drastic increase in the computation time necessary to find a solution. Instead FSVI assumes that the state space is fully observable and first solves the MDP for this case. The MDP is then used to generate a set of belief points for the PBVI solver. This is achieved by taking the Most Likely State (MLS) and to follow the MDP policy accordingly. It is orders of magnitude faster than HSVI and results in comparable polices. FSVI fairs badly however when information gathering actions are necessary. Since it is essentially using a myopic policy to generate its samples, these will be insufficient to find the global optimal policy when the solution requires information gathering actions. The very last sampling generation technique to date, which is considered to be the most efficient, is SARSOP (Kurniawati et al. (2008)). It uses aspects from both HSVI and FSVI. It keeps upper and lower bounds on the value function and also uses the MDP solution to generate samples. The key idea of SARSOP is to sample belief points which will contain the optimal set of samples necessary to achieve an optimal policy. Both SARSOP and HSVI2 are considered state of the art in PBVI value approximation techniques. See Du et al. (2010) for a review and comparison of both techniques on problems with thousands of states including simulation examples in grasping, tracking and UAV navigation.

These methods are well suited to addressing problems which are easily expressed in a discrete state space. All considered problems are simulation based and no physical interaction problems are considered. Besides the belief set generation problem, interest has also been poised on porting the PBVI to a continuous state space. An example of a continuous action space PBVI method is Perseus Spaan and Vlassis (2005), in which the authors replace the maximisation over the action by sampling the actions from a parametric continuous representation. In Porta et al. (2006) the state space, transition and observation model are represented by Gaussian Mixtures and the authors consider a particle set or Gaussian mixture representation of the belief. The authors show that a continuous representation of the state space preserves the PWLC property of the value function. They extend their method to continuous action and observations through sampling instead of discretising. Results are shown in a 1D continuous corridor setting. In a more recent approach Brechtel et al. (2013) a discrete state presentation of a continuous state space is learned and is combined with sampling techniques to solve the continuous integrals present in the Bellman equation. The explicit learning of the state representation leads to an increased performance when compared to the other continuous state PBVI methods.

PBVI techniques have come fare since their first application to robotic navigation back in 2003 and have lead to a rapid increase of interest. Initially only a few hundred states could be considered and now problems with over tens of thousand of states are being solved in seconds (very problem specific of course). Most of the research has focused on how to gather a good set of sample beliefs efficiently. Later efforts focused on adapting PBVI to continuous state spaces more suited to robotic applications. The main approach consists of using sampling techniques to overcome the maximisation over the actions (when considering continuous actions) or to choose a suitable parametric representation of the transition, observation and utility model so that the Bellman equation can be solved in closed form. Most evaluations of have focused on simulated and simplified robotic navigation problems in 1D and 2D. We have not discussed online POMDP-solvers since they are also based on VI and sampling techniques and thus share a lot of similarities with PBVI. We refer the reader to Ross et al. (2008) for a detailed review. In summary, trying to preserver the PWLC property of the value functions leads to complicated VI methods which are difficult to port to fully continuous state, action and observation space. Efforts which have attempted to do this have not yet be shown to scale. As a result of this difficulty, of making this transition to a fully continuous space, approximate value iteration methods have been explored as an alternative. In approximate value iteration the PWLC property of the value function is dropped and is represented by a regression function.

Approximate Value Iteration

Point-based Value Iteration techniques try to preserve the PWLC property of the value function. This directly leads to a discretization of the state space which if continuous by nature, is prone to the curse of dimensionality. An alternative approach is to represent the value function by a non-parametric function, parameterize the belief space and perform approximate dynamic programming.

A very first successful example of this approach is Monte Carlo POMDP (MC-POMDP) Thrun (2000) in which a continuous state, action and observation version of the Heaven & Hell benchmark problem was solved successfully with a working implementation on a non-simulated mobile base. The belief was represented by a particle filter and the policy by a Q-value function, whose functional form was a non-parametric regressor (k-nearest neighbour) of the particle filter. The distance metric was the sample KL divergence between two particle sets. The POMDP was solved through Reinforcement Learning (interaction with the environment) and approximated dynamic programming also known as experience replay, batch RL or Fitted Q-Iteration (FQI) Ernst et al. (2005b). Although highly computationally demanding the method was successful.

This inspired many similar approaches such as Brooks and Williams (2011) where the belief state filter was an Extended Kalman Filter (EKF), the value function was also non-parametric and the POMDP was solved via FQI. When compared with Perseus in a discretized 2D localisation task both approaches reached equivalent policies but the authors method achieved it far faster than Perseus, a PBVI method.

An alternative approach is to represented the history of the previous states or observations in an augmented state space and the treat the problem as a standard MDP. In this way the partial observability is directly encoded in the state representation. The motivation is that in contrast to POMDPs there has been fare more research focused on MDPs and much work has been done on the application of non-linear function approximators for representing the value function in combination with reinforcement learning optimisation techniques to solving them. A successful example was the usage of a multi-layer perceptron as a Q-value function approximator, Neural Fitted Q-Iteration (NFQ) Riedmiller (2005b). This approach was successfully applied to the standard RL benchmarking problems (carte pole, acrobat, mountain care), but no partially observable setting was considered. Later in Hausknecht and Stone (2015) the authors applied a Deep Recurrent Q-Network (DRQN) (extension to the work in Mnih et al. (2015)) to capture the history of states in a game of Pong where the state space was occluded half the time. By introducing a long term memory component the POMDP in effect is turned into a MDP and the authors apply an optimisation approach similar to FQI.

The advantage of these approaches is that problem with very large state spaces or continuous state spaces can be solved by using standard machine learning function approximation methods. These methods are easier to understand and implement and adapting POMDP methods to them is relatively straight forward. This is one particular way of dealing with the curse of dimensionality but not the only way. An alternative is to find a latent belief space which is of a much lower dimension than the original and perform value iteration in that space.

Latent Value Iteration

Latent belief space or belief space compression is a way of addressing the curse of dimensionality. The assumption is that although the belief space is of considerable size (thousands of dimensions) a latent belief space exists which is considerable smaller in terms of dimensions (a dozen). A first approach of compressing the belief is to transform it into a set of sufficient statistics (first and second moment for example) and treat the problem as a fully observable MDP in which the states are sufficient statistics of the beliefs. In Roy and Thrun (1999) the authors do just this, they compressed the filtered belief to its mean and entropy and performed VI on this augmented state space in a navigation task in which the goal was to reach a location with a minimum amount of uncertainty. This approach, called Augmented MDP (AMDP), brings a great simplification to solving the POMDP but at the cost of a lossy belief compression.

In further developments Roy (2005) compared both PCA and exponential-PCA (E-PCA) Roy and Gordon (2003), as a means of belief compression technique to find a low dimensional belief space. The authors showed that an original belief of thousands of dimensions could be compressed to a 10 dimensional belief space whilst retaining most of the information. This approach was shown to be superior to AMDP. It requires however computationally expensive transitions back and forth between the low and high dimensional belief states, a necessary step for the the application of VI. The latest work in this area is Li et al. (2010) which investigates the use of non-negative matrix factorisation in combination with k-means clustering as a way of compressing the belief. There method showed some improvement over the E-PCA approach but was only evaluated on discrete benchmark problems.

Belief compression as a means of reducing the curse of dimensionality is an interesting approach. The caveat is that it requires discretising the belief to a fixed grid, collecting many samples and learning an appropriate set of belief-basis eigenvectors. As such, the larger the state space, the larger the dimensionality and thus more samples are required to find a suitable set of basis belief-eigenvectors. Surprisingly, belief space compression methods have not had wide attention although they shown promising results.

Summary: Value Iteration

Value Iteration seeks to find an optimal policy directly through applying the Bellman equation to a belief-MDP (POMDP) and most of the research has focused on finding ways to alleviated the curse of dimensionality so that VI remains tractable in belief space. The first approach, PBVI, considers a relevant subset of the belief space. Because of the complexity involved in keeping the PWLC property of the value function which restricts its use in large state spaces, alternative approaches discard this property in favour of approximating the value function through machine learning regression techniques. These approaches are considerably more simple to implement than PBVI solvers which require heuristic pruning techniques and are difficult to port to continuous state spaces in general. Alternative approaches have considered finding a latent belief state and perform value iteration in this space. There has however been relatively little work in the latent belief space approach.

Overall, the above approaches consider mostly discrete actions even for the large state (history states) MDPs which have been gaining recent attraction. There are only a few exceptions and these resort to sampling strategies or the usage of paramaterized high level actions. The next approach we consider addresses the problem of continuous actions directly and are termed policy search methods.

2.3.2 Policy search

The approaches seen so far use a value function to encode the problem. When the optimal value function is solved, a policy can be derived from it by taking an action which maximises the value function at each time step, a process known as making the policy greedy with respect to the value function. This requires learning a high dimensional value function of the belief space and the resulting policies are not necessarily smooth, as small changes in the value function can lead to drastic changes in the policy. Even small approximation errors in the value function can lead to very bad greedy policies Baxter and Bartlett (2000). There is no doubt that deriving a policy from a generic value function for highly continuous policy, such as in the case of controlling an articulated robotic arm, is not easy.

This has lead to an alternate approach in which a policy is learned directly without a value function. An initial policy is defined in terms of a paramaterized function, $\pi_{\boldsymbol{\theta}}$, and the utility is a function of the policy parameters, $u(\boldsymbol{\theta})$. The optimal policy is found by searching for the parameters $\boldsymbol{\theta}$ which will maximise the utility function. This is can be accomplished through various optimisation methods: gradient descent, gradient free, expectation-maximisation, etc...

Gradient: policy search

A very early type of policy search was REINFORCE (likelihood ratio) algorithms first introduced by Williams (1992). From a set of task executions, also called roll-outs, the gradient of the utility function is estimated and used to improve the policy through gradient ascent. The key aspect of this approach is that the derivative of the cost function is independent of the state transition model and as a result the gradient can be estimated by Monte Carlo methods. Application of this methodology to a partially observable setting lead to Gradient POMDP, GPOMDP Baxter and Bartlett (2000) in which the authors developed a conjugate stochastic gradient ascent algorithm to optimise a policy as a function of the current observations. To be optimal, the whole history should be considered or some sort of memory (compressed history) should be introduced. In an extension to this method Aberdeen and Baxter (2002), the authors used a HMM to represent the POMDP which they learned the parameters in conjunction with those of the policy. These are early examples of policy search approaches which are able to fair well on the early POMDP benchmark problems (Heaven & Hell). The main difficulty is to reduce the bias and variance of the gradient estimate which preoccupies most gradient based approaches. Optimising the utility function via stochastic gradient ascent typically needs thousands of gradient estimates such that in expectation terms the parameters are maximising the cost function. An approach which mitigates this problem, coined Pegasus Ng and Jordan (2000), removes the stochasticity from the optimisation by setting the seed of the random number generator constant. A policy evaluation becomes deterministic and by repeating this process many times (different random seeds) the stochasticity is present between the different evaluations and not within them. The end result is the same as stochastic gradient ascent (if repeated sufficient times) but is far easier to optimise individual non-stochastic problems. This policy search method was used to learn a set of controllers for a radio controlled helicopter Kim et al. (2004), which is considered to be one of the very first successful applications of RL to a MDP/POMDP problem. Recent approaches to gradient based methods include grasping objects under Gaussian position uncertainty Stulp et al. (2011), Stulp et al. (2012).

Expectation-Maximisation: policy search

One drawback of gradient based optimisation is that the learning rate plays a significant role on the speed of convergence. An alternative approach consists of using Expectation-Maximisation (EM) methods Kober and Peters (2009) which do not require a learning rate. Successful applications include: ball-in-a-cup, a humanoid learning the skill of archery Kormushev et al. (2010b), learning how to flip a pancake Kormushev et al. (2010a) and keeping balance on a two-wheeled robot Wang et al. (2016). These are just some examples of the application of RL to continuous action and state space problems. When uncertainty is present, the maximum likelihood state estimate is typically taken and is treated as the true state. A good surveys on policy gradient search methods can be found in: Deisenroth et al. (2011), Kober et al. (2013).

Actor-critic: policy search

Gradient and EM methods only optimise the parameters of the policy, also known as actor only methods. An alternative is to have a separate parameterization of the value and policy functions. This approach is known as an **Actor-Critic**, in which the gradient of the utility function is used both to update the value and policy functions. It has been shown that this approach reduces the variance of the gradient estimate and allows to smoothly change the policy which is desirable when controlling a robot for instance, see Grondman et al. (2012) for a survey highlighting differences and advantages of policy search vs actor-critic methods. A successful application of actor-critic is (episodic) Natural Actor Critic (eNAC) Vijayakumar et al. (2003), Peters and Schaal (2008b), a method which uses the *natural gradient* of the value function to update the parameters of a policy. The advantage of using the natural gradient is that it guarantees small changes in the distance between the successive roll-out trajectory distributions. Previous policy gradient methods did not have such guarantees, since small parameter changes of the policy could lead to large changes in the roll-out distributions, which is undesirable. In terms of performance NAC converges faster than GPOMDP and has been applied to learn Dynamic Motor Primitives (DMPs) to control a humanoid robot.

Summary: policy search

For problems in which the state and action space are continuous, policy search is preferred to pure value iteration based methods, which is the case for articulated robotics. In this case, the policies are only guaranteed to be **locally optimal** as oppose to the VI methods which can find **global optimal** policies. However if the parameters of the policy are initialised such that it is in the vicinity of the global optimum of the utility function, then the local optimal will be global. A lot resides on the initialisation and dimensionality of the parameter space of the policy. In terms of using them to solve POMDPs, most examples, at least for robotic applications, act according to the Most Likely State (MLS) or are a function of a history of observations. In such a way the partial observability is **implicitly** encoded into the policy as opposed to explicitly as was the case for PBVI methods.

Policy gradient methods are iterative and generally require a lot of data to be able to achieve a good policy. Also often the policies learned are not transferable between different tasks and have to be completely relearned. This of course depends on the representation of the state space which if task invariant causes no problem, but unfortunately this is not the case. The next approach to treating uncertainty is more aligned with addressing this last issue of re-usability. These are the **planning** methods.

2.3.3 Planning

Belief space planners leverage the power of traditional planning and optimal control techniques such as: A*, D*, RRT, Dijkstra and LQR to the belief state space. In most of the following techniques (with a few exceptions), a fundamental assumption made is that the motion and measurement models are Gaussian and as a result, a point in the belief space can be represented by the first and second moment: the mean and covariance. An important distinction with VI and Policy search methods is that planners do not solve for a policy. These are online methods in the sense that they have to re-compute a set of actions at every time step as oppose to a policy which can directly query which action should be applied given the current state. The generic objective function used in belief space planning penalises for the amount of uncertainty at the goal and a cost is incurred for every step taken. The planned path will be a compromise between the exploitation actions, which seek to go directly to the goal, and information gathering actions, which seek to reduce the uncertainty.

Belief space road maps

An example of belief space planning is the application of Probabilist Road Maps (PMR) to a belief state space, Prentice and Roy (2009), referred to as Belief Road Maps (BRM). By taking advantage of the linear structure of the Kalman Filter update the authors show that the covariance matrix can be factorised such that a sequence of motion and measurement updates between two belief points in the BRM can be computed by a single linear operation parametrised by the current belief. The key advantage of this approach is that it allows for rapid replanning and is able to scale to large state spaces. The authors evaluated their planner in the MIT campus (simulated). Applications of this methodology include the control of an indoor quadrotor helicopter He et al. (2008) and indoor navigation (a. Agha-mohammadi et al. (2011), a. Agha-mohammadi et al. (2014)) (based on Feedback-based Information Road Maps FIRM , a similar approach in spirit to BRM).

Optimal control

Another main approach is based on optimal control theory, from which Linear Quadratic Controllers (LQG) have been adapted to a belief state space. In this setting the dynamics are considered linear (or linearizable) and the motion and measurement processes are Gaussian. The main difficulty of applying LQG to a belief space is that future observations are unknown, which implies that an expensive marginalisation of the observations has to be done. In Platt et al. (2010) the authors assume instead that at each time step the measurement obtained would be the **maximum-likelihood observation**. This assumption removes the stochasticity from the belief update (since the observations are considered known) and receding horizon optimisation techniques can be applied. These optimisation methods require a nominal trajectory which is generally generated assuming a fully observable state space with standard planning algorithms like RRT Van Den Berg et al. (2011), and subsequently refined by dynamical programming methods until a local optimal solution in attained. In Erez and Smart (2010), the authors parametrized the belief by a mixture of two Gaussians to to tackle unilateral constraints and applied their planner to a 16 dimensional attention allocation problem. The optimisation method used was Differential Dynamic Programming (DDP) and maximum likelihood observations were assumed. For implementations based on this approach, when the planned belief trajectory deviates from the observed belief, replanning takes place. In recent improvements, van den Berg et al. (2012), the assumption of maximum-likelihood observation was removed successfully and has been applied in a simulated surgery problem, Sun and Alterovitz (2014), in which a needle has to be navigated through a body without entering into contact with vital organs.

Most optimal control methods assume that the belief space can be parametrized by a single Gaussian function, which can be restrictive. There have been a few approaches which consider **non-Gaussian** belief state spaces. In Platt et al. (2012) the authors introduce a non-Gaussian belief. The approach initially finds the Most Likely State (MLS) and then samples a set of hypothesis states from the belief. The cost function, with respect to the ML and sampled hypothesis, results in a sequence of actions which will seek to generate measurements which will prove or disprove the hypothetical states with respect to the ML state whilst also trying to reach the goal. Recent work Zito et al. (2013) incorporates this optimisation method into a grasping problem under non-Gaussian pose uncertainty. The method in question is able to perform well with only a few drawn samples from the belief. However the object was not picked up and as a result the stability of the grasp was not evaluated.

Summary: planning

Most advances in planning methods in belief space have been in optimal control and were able to show applicability to high-dimensional belief state spaces in a variety of applications. To be fast these methods have to make assumptions with respect to the shape of the belief (Gaussian) and the type of future observations which are available. These can be restrictive but in many applications (such as those which use vision) the uncertainty of objects in the world are often parametrized by Gaussians. The main difference between optimal control approaches and policy search methods is that the computational burden is shifted to online resolution of actions as oppose to constructing a policy offline through repeated interactions with the environment which can be very time consuming. The advantage of planning methods is that they are more flexible than parametric policies in the sense that they are more generic. They solve the objective function online and can be used in different environments, as oppose to a policy which would have to be re-learned.

2.3.4 Heuristics

The methods discussed so far can be considered computationally expensive and/or constraining in the type of belief which can be used (typically a uni-modal Gaussian). If the problem domain is more complex or an expensive optimisation problem is not necessarily required, simple heuristic methods can achieve a satisfying solution and in some cases the equivalent of a full blown POMDP solver. Heuristic methods for dealing with uncertainty are widespread in robotics due to the high dimensionality and continuity of the state space. We consider here two heuristic approaches, myopic and information gain. Myopic ignores most of the variance in the uncertainty and considers only the Most Likely State (MLS) whist information gain considers actions in terms of their uncertainty reduction.

Myopic & Q-MDP

Myopic policies consider only the most likely states, which in the case of a Gaussian belief is the mean, and act accordingly. These types of approaches ignore the variance in the uncertainty and risk to fail catastrophically or result in sub-optimal behaviour. MLS is typically used in complicated domains such as grasping, especially when the actual shape of the object is considered to be unknown. A successful approach to this problem is to have a prior non-parametric regressor function representing the shape. As contacts are made with the object more points are added to the regressor improving the shape constructed by exploring the unknown object and gradually acquiring points. The uncertainty of the shape in a region is typically a function of the number samples. At this point either an exploratory movement is done to move a finger towards a region of high uncertainty (the MLS region) or a grasping attempt is carried out. In Hollinger et al. (2012) an AUV maps the hull of a ship by constructing a mesh and encoding the uncertainty of the mesh with a Gaussian Process (GP). A set of viewing locations, where there is uncertainty (MLS), are computed and a trajectory is obtained by solving a *travelling salesman* problem whilst seeking to maximise coverage of areas with high mesh uncertainty. In Chen and von Wichert (2015) a grasping controller uses the uncertainty, encoded by GP, to guide an exploration process. The fingers would move towards regions of high uncertainty whilst keeping contact with the object. For a good review on related methods for grasping objects under shape uncertainty consult Li et al. (2016), where the authors also use a GP based method to encode the shape uncertainty. The exploration methods for all these methods are in the same in spirit; move towards regions which have high uncertainty (exploration) and when the uncertainty is sufficiently low perform a grasp (exploitation).

An improvement is to consider the variance in the uncertainty and not just the MLS. Such an approach is a called Q-MDP Littman et al. (1995), Nowé et al. (2012) in which the underlying MDP is first solved assuming the state space to be fully observable. Then an action is taken which maximised the expected MDP value function weighted by the belief. This approach only considers uncertainty for one time step but it has been shown to be efficient in some domains (Thrun et al., 2005, Chap. 16). The negative aspect of this approach is that no information gathering actions emerge and the method will fail in problems where this is necessary (Heaven & Hell benchmark problem for instance). Most PBVI based research compare their algorithms against a Q-MDP agent and PBVI always fairs better. For a comparison of different heuristics such as Q-MDP and MLS consult Cassandra et al. (1996b) and for a more recent comparison Lin et al. (2014). A recent application of this method include gaze allocation problems Nunez-Varela et al. (2012) where the uncertainty originates from the limited field of view. In Hauser (2011), Q-MDP is used to evaluate nominal trajectories generated from RRT where starting positions were sampled from the initial belief. A recent follow up on this idea, Vien and Toussaint (2015), considers a task in which a robot has to localise itself with respect to a table. A set of macro actions are evaluated in a Q-MDP framework to achieve this task in which each macro action is solved by an optimal control method.

Both MLS and Q-MDP do not fully consider the uncertainty. This of course leads to great computational gain but at the expense of the quality of the policies, which can be very sub-optimal in some cases. It is known that for increasing the chance of success, a policy which deals with uncertainty needs both **goal orientated** and **information gathering** actions. The next heuristic approach, which we call **information gain**, is based on this concept.

Information gain

Information gain is the decrease or increase of uncertainty resulting from the application of an action. It is obtained by forward simulating the belief and computing the difference between the current entropy and resulting entropy of the simulated action. The vast majority of applications consider a set of marco/parametrised actions. In this set there are typically goal orientation actions which will act as if the state space was fully observable (MLS move) and information gathering actions, whose goal is to reduce the amount of uncertainty such that the goal orientated actions have a higher chance to succeed. The cost function which is optimised is typically a compromise between the distance/time taken to reach the goal and the amount of information gained while executing the task. An early example considered path planning problem for a robot in the National museum of American history Roy et al. (1999). An information gain map was first computed off-line in which a map cell gave an estimate of how much information would be acquired at this location. This was incorporated into an objective function which optimised the information gain along a route with respect to the time taken to reach the goal. The path was given by solving the objective function using dynamic programming. In this case no explicit actions where defined, but the uncertainty was taken into account by weighting informative regions more than open space. The result was trajectories which stayed close to walls. Information gain methods are often used in SLAM applications because of the extremely high dimensionality of the belief space which is of the map and robot position. In Stachniss et al. (2005) a mobile robot is exploring and building a map of an office floor and a set of macro actions are available. A portion of the actions are exploratory and lead the robot to unexplored areas which results in an increase of uncertainty in the overall map whilst the other actions brings the robot back to already explored areas resulting in an improved estimate of the map. For each action the information gain is computed and incorporated in an cost function. A one time step look head is done for each action, which potentially implies an expensive forward simulation, and the action giving the maximum information gain is chosen. This approach has been shown to be effective for large state space problems, notably in Active SLAM navigation Vallve and Andrade-Cetto (2014).

Information gain maximisation is not only restricted to navigation, there are many examples in grasping where this approach is used. Examples include tactile driven exploration such as in Hsiao et al. (2010) where a parametrised set of goal orientated and information gathering actions are used in the context of estimating the pose parameters (6D) of a power drill. The information gain of each action is incorporated into a cost function and the best action is chosen accordingly. The authors report a breadth first search depth of one action to achieve a good performance for the task. Later grasping approaches have built on this with different modifications to the information gain metric Javdani et al. (2012) and there have been successful applications such as finding a door handle Hebert et al. (2013) and opening a door.

Summary: heuristic

Heuristic methods make strong assumptions which alleviates both the curse of dimensionality and the curse of history associated with POMDP problems. Either the MLS is considered (curse of dimensionality) or the planning horizon is restricted to one time step look ahead (curse of history) as it is the case for Information Gain methods. Heuristic methods in robotics have been regaining traction. In the early days of robotics methods such as Q-MDP, MLS, information gain maps and "best fields of view" were the predominant methods for considering uncertainty in policies and planning algorithms. This was simply due to the computational limitations of the time and POMDP solvers could only handle a few states before the arrival of PBVI methods. Since more sensory information is available and used in robotic systems it is again computationally expensive to compute optimal policies. In many cases spending large computational resources does not result in policies which are obviously superior to simple and intuitive heuristics. Lately many DARPA[2] teams when faces with state uncertainty resort to information gain heuristics, for instance.

2.3.5 Summary: literature

In the literature we characterised four approaches of how artificial agents have been programmed to reason under uncertainty.

When control algorithms were first being applied to mobile robotics uncertainty was handled with heuristics: MLS, Q-MDP and other techniques not fully discussed such as *next best view* methods. Practically speaking the computational resources at the time were too limited and it was unfeasible to solve optimally for a POMDP problem. Also it is not clear at what time the robotic community started to apply results from operational research to robotics in the case of partial observability. Certainly it is not until the advent of the first point-based value iteration methods that there was a shift of interest towards improving POMDP solvers such that they could be applied to robotic domains (navigation & manipulation). When evaluated against heuristics methods it was clear that in some scenarios (Heaven & Hell problem) the POMDP solvers did far better. Value Iteration methods have not been widely used in cases where the action space is continuous. There have been efforts to adapt them to continuous actions space, however there is yet no concrete evidence that these methods scale. If the robotic domain requires continuous actions then either policy search or optimal control methods are preferred. Policy search methods were first considered since they are part of the markov decision process family which is within the POMDP framework.

Policy search methods consider the uncertainty implicitly. These methods work well when there is relatively few control parameters and behaviour to be learned are either reflexes (like in the case of the autonomous helicopter) or primitive actions such as picking up an object. The uncertainty considered in the reviewed literature on policy search methods is predominantly characterised by a Gaussian function. It is not clear how well policy search methods would scale to situations in which there is a lot of uncertainty and so fare there has not been a lot of emphasis on comparing policy search methods with heuristics.

Optimal control methods came later, after policy search, and have recently started to gain traction since the adaptation of LQR to belief state spaces. As for policy search methods the uncertainty is considered Gaussian although recent research has been addressing this. The advantage of optimal control with respect to policy search methods is that they are more flexible since the objective function is resolved online. But at the expense of an increased computational cost.

[2] https://www.youtube.com/watch?v=9Oav3JajR7Q

Heuristics are still actively being used in research and very successful applications in the DARPA robotic challenge use heuristic approaches. The probable reason is the volume of sensory information and the size of the control architecture in robotic platforms competing does not leave room for anything else. Especially when considering project management constraints and reliability. That said, maybe there is no reason to use complicated methods. We note that there has been a significant absence of comparison between optimal control, policy search and heuristic problems on the same set of benchmark problems.

In Figure 2.8, we summarise attributes we consider important in the four approaches we reviewed. We bring attention to the typical type of actions and problems which these methods address. Note that we consider both Policy search and Value Iteration methods as being **off-line**. Although many authors say that the policy can be executed at any time, the optimal solution is not attained until after many interactions with the environment. This is not the case for Optimal Control and most Heuristic methods which give a solution on the spot, which we consider to be **on-line** methods.

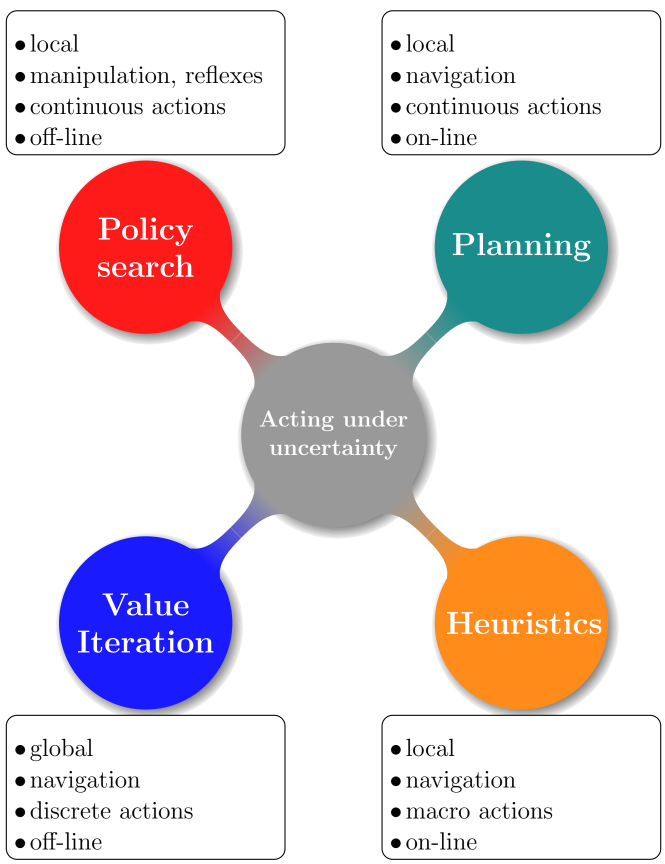


Figure 2.8: Summary of the aspects of the reviewed methods. Local refers to the optimality of the solution, on/off-line refers to if the solution is computed on the stop (on-line) or many simulations are required to obtain the solution (off-line).

The performance of all the methods mentioned in the literature review crucially depend on the quality of **exemplary demonstrations**. For instance, PBVI require search heuristics to find an optimal set of belief points, the quality of the optimal policy of policy search methods depend on the exploration-exploitation trade-off and optimal control methods strongly depend on the initial nominal trajectory. In a way this is intuitive, if you initialise your search method or algorithm with an initial solution which is of high quality (close to optimality) then which ever optimisation method used PBVI, Policy Search, Planning,... a solution should be obtained with computational ease. The question is then: *how to generate such exemplary demonstrations ?*

2.4 Approach

As discussed in the literature summary, the initial data provided to the solvers plays an important role in the optimisation time and quality of the final policy or plan. A popular approach known as Programming by Demonstration (PbD) is a way to provide initial exemplary data. PbD is a methodology whose aim is to achieve the transfer of knowledge and behaviour from a teacher to an apprentice. The teacher is usually a human expert (this is not a constraint) who demonstrates to an apprentice how to accomplish a task. In the case of articulated robots, kinesthetic teaching is often preferred. The teacher would hold the robot, which is back drivable, and demonstrate to it trajectories. From the trajectories the states and actions, at each time step, are recorded and stored in dataset $D = \{(x, a)\}$ which is then used to learn a a policy $\pi_{\boldsymbol{\theta}}(x, a)$, usually a regressor function, which encapsulates the taught behaviour. Other ways are possible such as using vision or a wearable interfaces which are common to both teacher and expert. We will not go into a great detailed review of PbD, for an in depth review the reader is referred to Billard et al. (2008a), Billard and Grollman (2013). PbD has had many successful applications when the state space is considered observable but for the latent state case there are very few examples.

In this thesis we apply the PbD framework to a partially observable setting; we want humans to teach robots how to act under uncertainty. We know that generally speaking we are better at handling uncertainty than artificial agents, especially in haptic and tactile tasks. A hypothesis for this observation is probably that our perception capabilities are much higher and acute than current robotic software and hardware systems. To be able to study the ability of humans as teachers in a POMDP setting, we chose tasks in which a high level of uncertainty is present. For this reason we restrict ourselves to tasks in which the subjects can only use, their sense of touch. We namely consider **search tasks** in which a human is searching for an object whilst **blindfolded**. In summary we seek to learn control policies for robots in tasks which have the following problem specific attributes:

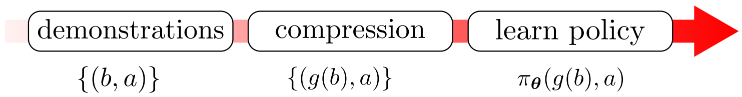


Figure 2.9: Three steps in learning a POMDP policy from human demonstrations: First gather the belief-action dataset, second compress the beliefs and third learn a generative policy.

Problem attributes

- Blindfolded search tasks, no vision.
- Sparse haptic and tactile information.
- Continuous action and state space.
- High amount of non-Gaussian uncertainty.

In our approach, the robot apprentice observes the human teacher demonstrate a search task. As the human teacher searches, he makes contact with various aspects of the environment trying to localise himself whilst looking for the object in question. During the demonstration the apprentice infers the humans beliefs by observing his actions and stores them into a dataset $D = \{(b, a)\}$. Given this belief-action dataset we learn a generative distribution $\pi_{\boldsymbol{\theta}}(b, a)$ of the behaviour exhibited during the search which is then transfer to the robot apprentice. In Figure 2.9 we illustrate the PbD-POMDP data pipeline.

This is the general concept but there are a few caveats which make this task not as straight forward as it seems.

- **The belief state is unknown**: When the robot apprentice is watching the human perform a search task under state uncertainty, it is unable to observe the belief state of the human. All that the agent can observe directly are the actions of the teacher. We make **two assumptions**, the first is that the apprentice can infer the observations of the teacher by examining the teachers relation with the environment and secondly the initial uncertainty of the teacher is assumed to be known. From these two assumptions, the sequence of belief states can be inferred via a Bayesian filter. This implies that the mental belief state of the human teacher is in fact known given the assumptions. We give more details in Chapter 3 on the validity of these assumptions and discuss their relation to Bayesian Theory of Mind (BoTM).

- **Learning a policy as a function of non-parametric beliefs:** Given that we are considering high levels of uncertainty and the observations are sparse, in the form of contacts, no parametrisation of the belief in terms of a Gaussian function would be adequate. In this thesis all the considered beliefs will be from the non-parametric Bayesian filter family,

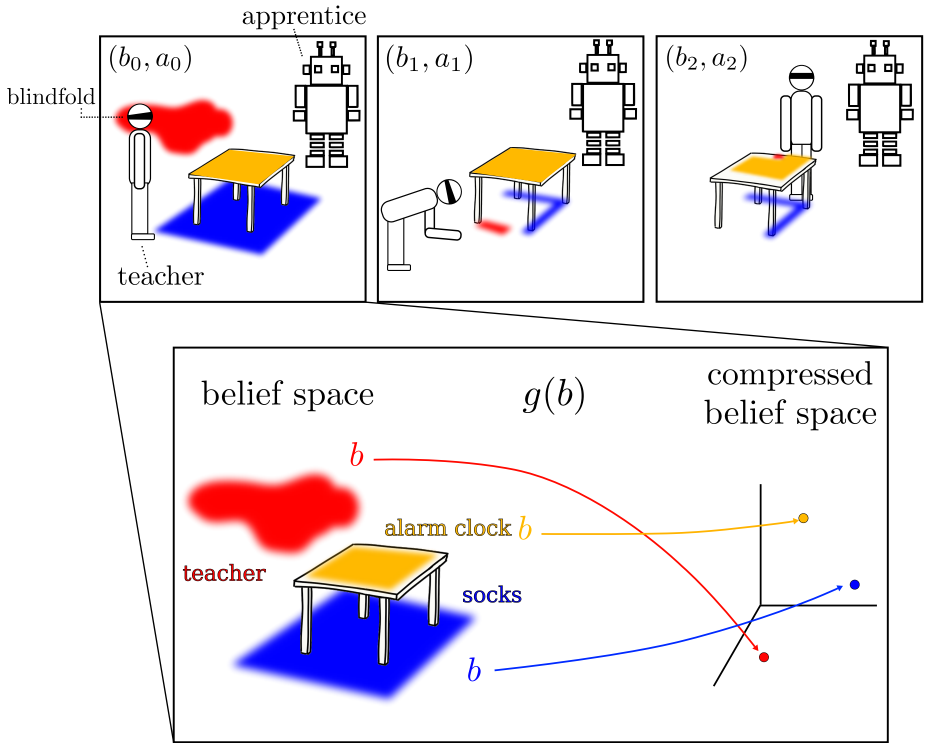


Figure 2.10: Demonstrations: An apprentice is looking at a human teacher who is searching for the alarm clock's button and his pair of socks. The apprentice assumes the structure of the original beliefs the human teacher has with respect to his position and that of the alarm clock and socks, these are represented by the red, yellow and blue density functions. **Compression:** Given the data set of beliefs and actions obtained from the demonstrations, the beliefs is compressed to a fixed parametrisation. **Learn policy:** A generative policy, $\pi_{\boldsymbol{\theta}}(g(b), a)$ is learned from the actions and compressed beliefs and can be executed according the schematic on the right. SE represents any Bayesian state space estimator, which takes as input, the current observation, belief and action and outputs the next belief state.

such as particle filters, which allows for a lot of flexibility. Learning a policy directly as a function of a particle filter is intractable. First in non-parametric filters there is typically thousands of states and in efficient particle filters the number of parameters varies over time. We **compress the belief** into the most likely state and the entropy. In this way the size of the belief state is fixed and low dimensional.

In Figure 2.10 we illustrate an example of human teaching an apprentice robot how to search for objects (alarm clock and socks) in a state of high uncertainty, the human is blindfolded. Given what the apprentice can observe he must infer the beliefs of the teacher (red, blue and orange probability density functions).

- **Reactive policy**: The control loop cycle, which computes the belief state, compresses it, and computes the resulting action to take, should happen at around 10 to 100 Hz. This range may seem arbitrary but is in fact based

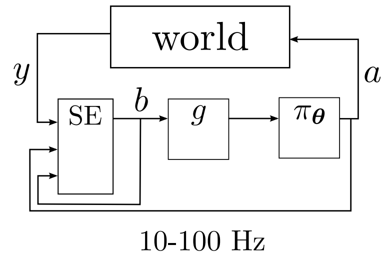


Figure 2.11: Control architecture of the apprentice robot. The control loop should run between 10-100Hz. Given an applied action, the world returns an observation which is integrated by the State Estimator (SE) to give the current belief. The belief is the compressed and given as input to the policy.

on the humans control ability which at the highest cognitive level a delay in response is around 100ms and at the lowest reflex level at around 10-20 ms Winter (2009). This is to draw attention that the full control loop, belief filtering, compression and action prediction should all happen within this range. See Figure 2.11 for an illustration of the control architecture used.

- **Scalable belief filter**: In scenarios in which there are multiple objects being searched for by a human teacher, the joint belief distribution of a non-parametric Bayesian state space filter will become quickly computationally intractable. This motivates the development of a new type of SLAM filter methods which can scale in situations in which observations are very sparse.

All of the above points are the motivation behind many of the decision choices we take and use in the subsequent chapters. They are necessary such to be able to successfully teach robotic systems to act as humans in partial observable states.

Chapter 3

Learning to reason with uncertainty as humans

The conclusions drawn from the literature survey in Chapter 2 are that non-heuristic methods for planning and control rely heavily on the initial data provided to their respective optimisers. An ideal initial set of behaviour should be comprised of explorative and exploitative actions so that a final optimal policy can quickly achieve the balance between minimising uncertainty and solving the task at hand. This is especially true for Reinforcement Learning (RL) methods which make use of explorative actions to be able to find an optimal policy. In many RL applications random exploration or Gaussian noise perturbation is sufficient to find an optimal policy. This is the case when either an exhaustive search of the action space is possible (mountain cart, inverted pendulum, etc...) or in policy search methods where the policy is parametrised by a few parameters. In continuous action-state space POMDPs, when a generic non-parametric policy is desired this is not feasible, especially when the decision horizon is long. Continuous action-state space POMDPs applications have predominantly focused on cases in which the uncertainty can be quantized by a single Gaussian parametrisation. This representation can be constraining since it requires the observation likelihood to be Gaussian as well. This assumption is restrictive and ill-suited for haptic search tasks in which observations are discontinuous and occur as impulses.

In this chapter, we demonstrate that human foresight and intuition can be leveraged as a means of solving the exploration/exploitation dilemma under partial observable conditions. Human beings are versatile in their ability to accomplish tasks which are considered to be complex by current robotic standards. This perceived ability which we have over current robotic systems, due to our prior domain knoweldge and experience, can be extracted, encapsulated and transferred to a robot apprentice.

To demonstrate the application of the transfer of behaviour from a human teacher to a robotic apprentice we apply the framework outlined in Chapter 2, Section 2.4 (PbD-POMDP) to a blindfolded haptic search task. In our blindfolded search task, both a robot and a human must search for an object on a table whilst deprived of vision and hearing, illustrated in Figure 3.1. The robot and human both have prior knowledge of the environmental setup making this a specific search problem with no required mapping of the environment, also known as active localisation. In Figure 3.1, a human has his sense of vision and hearing impeded, making the perception of the environment partially observable and leaving only the sense of touch available for solving the task. The hearing sense is also impeded since it can facilitate localisation when no visual information is available and the robot has no equivalent giving an unfair advantage to the human. By impeding hearing we align the perception correspondence between the human and robot.

By representing the belief of the human's position in the environment by a Particle Filter (PF) and learning a mapping from this belief to hand actions (velocities) with a Gaussian Mixture Model (GMM), we can model the human's search process and reproduce it for any agent. We further categorize the type of behaviours demonstrated by humans as being either risk-prone or risk-averse and find that more than 70% of the human searches were considered to be risk-averse. We contrast the performance of this human-inspired search model with respect to Greedy and Coastal Navigation search methods. Our evaluation metric is the distance taken to reach the goal and how each method minimises the uncertainty. We further analyse the control policy of the Coastal Navigation and GMM search models and argue that taking uncertainty into account is more efficient with respect to distance travelled to reach the goal.

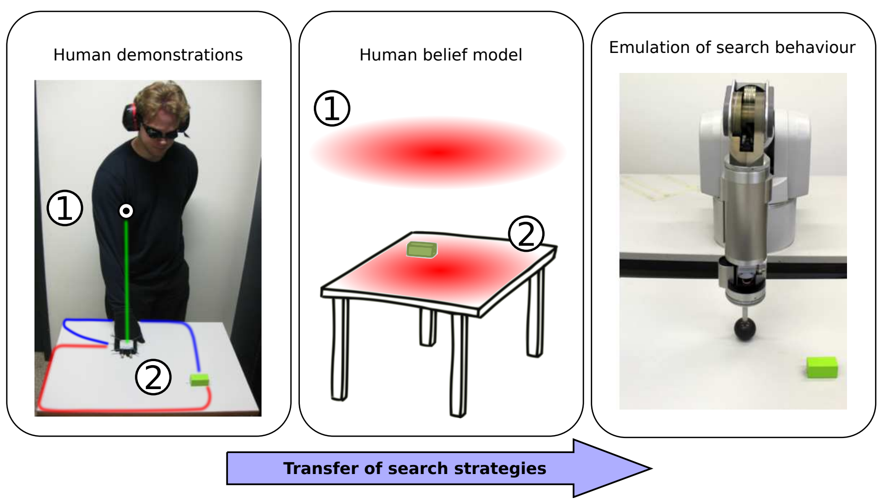


Figure 3.1: Blindfolded search task *Left:* Search task, a human demonstrator searching for the green wooden block on the table given that both his hearing and vision senses have been impeded. He starts (hand) at the white spot near position (1). The the red and blue trajectories are examples of possible searches. *Middle:* Inferred belief the human might have with respect to his position. If the human always starts at (1) and his belief is known, all following beliefs (2) can be inferred from Bayes rule. *Right:* WAM Robot 7 DOF reproduces the search strategies demonstrated by humans to find the object.

There are **two assumptions** we make when applying Programming by Demonstration, PbD (also known as Imitation Learning), to the POMDP task described above. The first assumption is that the human teacher's *spatial cognitive* abilities are good enough to accomplish the task in a consistent fashion.

In other words demonstrations should not be random and a pattern exists. The second assumption is that human's beliefs inferred by the apprentice are close to the actual belief of the human.

3.1 Outline

- 3.2 Background
  We review aspects of the literature in robotics and cognitive science which are related to spatial navigation which consider scenarios with limited perceptual information. We review related literature from *Spatial Navigation*, *Theory of Mind* and *Programming by Demonstration*.

- 3.3 Experiment
  The table search experiment protocols are described and we detail how to learn and transfer search strategies from human teachers to a robot apprentice. A total of 15 human teachers participated and each gave 10 demonstrations, giving a total of 150 searches.

- 3.4 Formulation
  We detail the implementation of the human belief in terms of a Particle Filter (PF). This includes the measurement and motion models. We describe how we compress the belief particle filter in terms of the most likely state and differential entropy.

- 3.5 Policies

  – 3.5.1 Modelling human search strategies
    We detail the implementation and parametrisation of a Gaussian Mixture Model (GMM) policy encapsulating the human search strategies and how it synthesises new searches.

  – 3.5.2 Coastal Navigation
    We detail the implementation of a Coastal Navigation policy, used as a comparison with the GMM policy.

- 3.6 Results
  We conduct three types of analysis: we quantify the behaviour present in humans and policies in terms of riskiness; we qualitatively evaluate the differences between the GMM policy learned from human demonstrations and the Coastal Navigation policy; we evaluate the distance taken to find the goal for a set of four search policies, including the GMM.

3.2 Background

3.2.1 Spatial navigation

Spatial navigation, Wang (2007), Wolbers and Hegarty (2010), focuses on the role that sensory perception (vision, vestibular, proprioception ...), motor control and mental cognition have on the navigational ability of humans, animals and insects. A central aspect of spatial navigation is the way in which we mentally represent the geographical world, known as a *cognitive map* (mental representation of environment first proposed by Tolman, 1948) and how we update our pose estimation in this map. The aspects of both construction and correction of a cognitive map have been studied in great depth, Wolbers et al. (2008). There is reported evidence that we use both vestibular and proprioception in inferring self-motion in order to update our position through dead reckoning (also known as path integration). Given the estimated position we then use external cues such as geometric (the shape of a room) and features (the colour of the walls), to correct our position. The actual representation of our position and environment in our cognitive map has been proposed, Burgess (2006), to be either encoded in our own frame of reference (egocentric) or in a frame of reference which is independent to us (allocentric) and acts like a standard paper map or both. This cognitive map enables us to reason about the relations between our own position and that of other items and landmarks present. This representation also facilitates our ability to localise ourselves and plan novel routes when needed.

In Wang and Spelke (2000), the authors studied the effect that disorientation has on blindfolded subjects' ability to recover their heading, which is necessary for re-localisation. Through eight different experiments they concluded that humans have an egocentric cognitive map.

Studies have also looked at the difference between congenitally blind, late blind and sighted people in their ability to encode ego-allocentric cognitive maps. In Pasqualotto et al. (2013), the authors dispose a set of seven objects (brush, slipper, pan, dish, book, spoon, bottle) in the form of an array in a 12.5m $\times$ 9m room. The objects are positioned on top of stools. During a training phase, ten congenitally blind, ten late blind and ten blindfolded sighted people were taken through the setup and touched all objects present. This guided exploration (the experimenter leading the subject through the object array) was repeated until the participants could correctly recall all the objects' locations twice consecutively without help. In a testing room (no objects present) the participants were asked "Judgement of Relative Direction" questions and the accuracy and response time were recorded. From the results the authors concluded that blindfolded and late blinded participants used a allocentric representation of the object array, whilst the congenital blind subjects use an egocentric model. The cause of this difference is attributed to the role played by vision in the development of the multisensory brain area, in which vision is necessary for the development of an allocentric model.

Many similar experiments have been conducted and a summary can be found in the following review Burgess (2006), where the authors explicitly state that a consensus has formed; both egocentric and allocentric representations of the environment are working in parallel. Current questions ponder whether allocentric models are part of the semantic memory as opposed to the procedural memory used by the egocentric model.

Spatial cognition and memory

The quality of the human teacher in search tasks, which are partially observable in the terms of absence of vision, will strongly depend on the teacher's ability to maintain an accurate cognitive map of his environment. This implies that the size of the environment and search task will have an effect on the teacher's ability to provide near optimal demonstrations. Early and influential research into human's short term memory was presented in 1956 by George Miller in a seminal work, Miller (1956) (22'780 citations), in which he described the "so called" magical number of our short term memory as being 7 ± 2 items, known as *Miller's Law*. This research was conducted on a one dimensional task in which no spatial navigation was required. Since then there have been many studies investigating the limits of short term memory.

In Lavenexa et al. a set of subjects had to find either 1, 3, 5 or 7 goal pads, among a grid array of 23 pads in a 4m $\times$ 4m room, within a 1 minute interval. They measured the subjects' error in terms of the number of locations visited before finding the goals. They found that on average the subjects had to visit "$1.6 \times \#num\_goals$" pads before achieving the task. The authors concluded that in this spatial navigation task there was no magical number which represents the limit of short term memory. In another spatial navigation experiment, Iachini et al. (2014), the effect that the scale of the environment has on the ego-allocentric representation is studied in blindfolded, late and early blind subjects. The main findings were that cognitively blind people have more difficulty in developing an allocentric representation of the world.

In Stankiewicz et al. (2006), a search task in a virtual maze is conducted by a set of human subjects. The aim is to investigate the limitations that perception, memory and uncertainty have on human decisions in comparison with an ideal agent (POMDP solution). The authors' main findings were that as the size of the maze increases the performance of the human subject decreases with respect to the ideal agent, as human subjects are limited by the uncertainty in their location and have difficulty in maintaining multiple hypotheses.

Summary: spatial cognition

The studies detailed above reported that if the environment is not overly large and complex our cognitive model is sufficient to produce policies which are on par with an optimal POMDP agent.

Our study seeks to transfer exploratory behaviour from human teachers to a robot apprentice in a partially observable setting. In our search scenario the environment is less than 3 meters in length and 2 meters in depth with a single goal object to be found. Given this setup and the evidence from previous studies, humans should be able to achieve this task with a high level of proficiency.

This is beneficiary since currently both humans and animals are better at spatial navigation than robots Stankiewicz et al. (2006) especially when uncertainty is present. The quality of the demonstrations will strongly depend on the teacher's short term memory in retaining a sufficiently accurate cognitive map of the environment.

3.2.2 Human beliefs

A crucial aspect for the success of PbD-POMDP learning is that the apprentice be able to infer the human's belief of his location whilst he is searching. In others words the apprentice (human or robotic) has to infer the cognitive map of the teacher.

The study of inference of another's mental state is part of Theory of Mind (ToM) Sodian and Kristen (2010), which is concerned with our ability to infer beliefs, desires, intentions, perception, goals and current knowledge. In this study, the apprentice will have to infer the teacher's beliefs which we assume are **rational**. A rational belief is a belief for which observations bring supportive evidence and gradually increase the certainty of the belief. In a recursive formulation this known as Bayesian Theory of Mind (BToM), where the Bayesian component highlights the hypothesis that humans integrate information and update their beliefs in a similar fashion to Bayes rule.

Due to the complexity in the number of sensory sub-components, such as gaze following, and their interplay, required as a precursor to the development of a ToM, much effort has been focused their development. Early work in implementing a ToM in a humanoid robot was introduced in Scassellati (2002) and is based on ToM models of Leslie (1994) and Baron-Cohen (1995). The author focused on building basic skills such as face finding and distinguishing animate and inanimate stimuli but left open the problem of the final interaction between all the components.

In Butterfield et al. (2009), the authors model ToM as a Markov Random Field which defines a joint probability distribution over a set of hidden actions and observation variables. The functions of these variables are hand-crafted for each experiment. The authors demonstrate that through a suitable parameterisation of the MRF they achieve results comparable to the predictions of

ToM. Recently in Devin and Alami (2016), ToM and planning architecture have been integrated in a joint action collaborative human robot task, in which position, goal and action state of the human partner is maintained by his robotic assistant.

Work on modelling human beliefs and intentions has been undertaken in cognitive science, Bake et al. (2011), Richardson et al. (2012). In Baker et al. (2006), the authors present a Bayesian framework for modelling the way humans reason and predict actions of an intentional agent. The comparison between a generic Bayesian model and the humans' predictions yielded similar inference capabilities. This when asked to guess the intentions of a goal oriented agent in a 2D world, which both the Bayesian model and the humans were observing. This provided evidence supporting the hypothesis that humans integrate information using Bayes rule. Further, in Bake et al. (2011), a similar experiment was performed in which the inference capabilities of humans, with regard to both belief and desire of an agent, were comparable to that of their Bayesian model. Again they found the human's inference was comparable to that of the Bayesian model.

In our PbD-POMDP framework we make a similar hypothesis that humans integrate information in a Bayesian way, however in a continuous domain. We infer the belief that humans have of their location in the world during search tasks.

3.2.3 Programming by demonstration & uncertainty

Programming by demonstration (PbD) is advantageous in the POMDP and MDP contexts since it removes the need to perform the time consuming exploration of the state-action tree to discover an optimal policy and does not rely on any exploration heuristics to gather a sufficient set of belief points (as in point based value iteration methods discussed in Chapter 2).

We expect humans to perform an informed search. In contrast to stochastic sampling methods, humans utilise past experience to evaluate the costs of their actions in the future and to guide their search. This foresight and experience are implicitly encoded in the parameters of the model we learn from the demonstrated searches.

PbD has a long history in the autonomous navigation community. In Kasper et al. (2001), behaviour primitives of the PHOENIX robot control architecture are incrementally learned from demonstrations. Two types of behaviour namely *reactive* and *history-dependent* are learned and are encoded by radial basis functions. The uncertainty is implicitly handled by directly learning the mapping between stimulus and response. In Hamner et al. (2006) the parameters of a controller which performs obstacle avoidance are learned from human demonstrations. The uncertainty is inherently handled by learning the relation between sensor input and control output. In Silver et al. (2010) the objective function of a path planner is learned from human demonstrations. The objective function is a weighted sum of features corresponding to raw sensor measurements. This is another example where the partial information of the state is taken into account at the perception-action level, with the difference that instead of a policy being learned the objective function from which it is generated is learned. In Nicolescu and Mataric (2001), the authors learn how to combine low level pre-acquired action primitives to achieve more complex tasks from human demonstrations, but they do not consider the effect of uncertainty.

Much work has been undertaken in learning reactive-behaviour, history dependent behaviour and combining multiple behaviour primitives to achieve complex behaviour. However very few have studied the effect of uncertainty in the decision process and do not consider it during the learning or assume that it is implicitly handled. A noticeable exception is Lidoris (2011), in which a human expert guides the exploration of a robot in an indoor environment. The high level actions (*Explore*, *Loop Closure*, *Reach goal*) taken by the human are recorded along with three different features related to the uncertainty in the map. Using SVM classification a model is learned which indicates which type of action to take given a particular set of features. The difference with our approach is that we perform the learning in continuous action space at trajectory level and multiple actions are possible given the same state, which cannot be handled by a classifier.

3.3 Experiment: table search

In our search task setup, Figure 3.2 and Figure 3.3 (*top left*), a group of 15 human volunteers were asked to search for a wooden green block located at a fixed position on a bare table. Each participant repeated the experiment 10 times from each of 4 mean starting points with an associated small variance. The starting positions were given with respect to the location of the human's hand (all participants where right handed). The humans were always facing the table with their right arm stretched out in front of them. The position of their hand was then either in front, to the left, to the right, or in contact with the table itself.

As covered in the background section, previous work has taken a probabilistic Bayesian approach to model the beliefs and intent of humans. A key finding was that humans update their beliefs using Bayes rule (shown so far in the discrete case). We make a similar assumption and represent the human's location belief (where he thinks he is) by a particle filter which is a point mass representation of a probability density function. There is no way of knowing the human's belief with certainty. We make the critical assumption that the belief is observable in the first time step of the search and all following beliefs

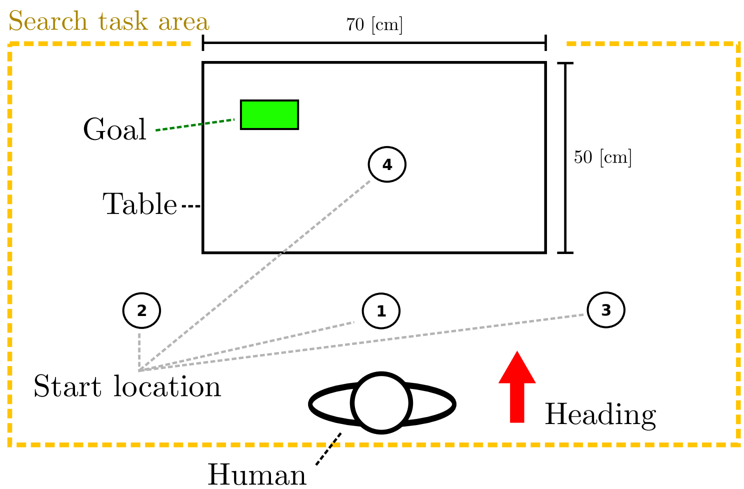


Figure 3.2: Table search task. Blindfolded human subjects after a disorientation step are placed in one of the four starting locations. The heading of the subject is always kept the same. The human's objective is to locate the green block on the table. Throughout all experiments the green wooden block is kept in the same location.

are assumed correct through applying Bayes integration. The belief is always initialized to be uniformly distributed on top of the table, see Figure 3.3 (*top right*), and the starting position of the human's hand is always in this area.

Before each trial the participant was told that he/she would always be facing the same direction with respect to the table (so always facing the goal, like in the case of a door) but his/her translational starting position would vary. For instance, the table might not be always directly in front of the person and his/her distance to the edge or corner could be varied. In Figure 3.3 *bottom left*, we illustrate four representative recorded searches whilst in the *bottom right*, we illustrate a set of trajectories which all started from the same region. One interesting aspect is the diversity present, demonstrating clearly that humans behave differently given the same situation.

It is non-trivial to have a robot learn the behaviour exhibited by humans performing this task. As we cannot encapsulate the true complexity of human thinking, we model the human's state through two variables, namely, the human's uncertainty about his current location and the human's belief of his position. The various strategies adopted by humans are modelled by building a mapping from the state variables to actions, which are the motion of the human arm. Aside from the problem of correctly approximating the belief and its evolution over time, the model needs to take into consideration that people behave very differently given the same situation. As a result it is not just a single strategy that will be transferred but rather a mixture of strategies.

3.4 Formulation

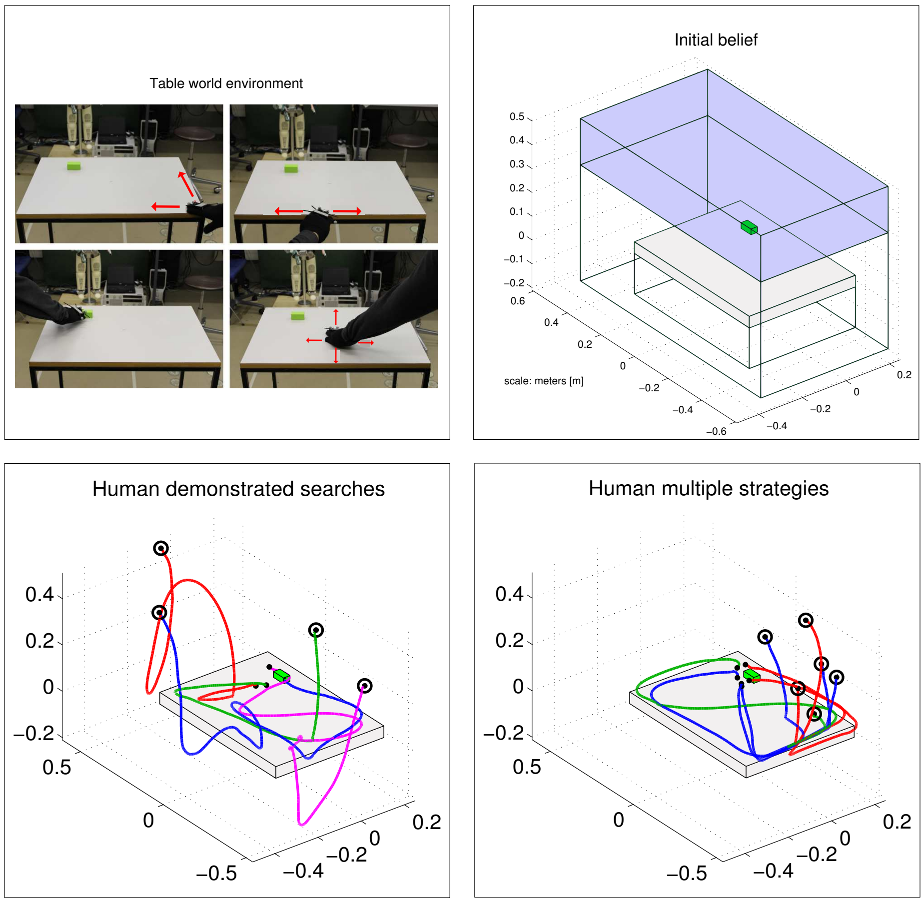


Figure 3.3: *Top left*: A participant is trying to locate the green wooden block on the table given that both vision and hearing senses have been inhibited. The location of his hand is being tracked by the OptiTrack® system. *Top right:* Initial distribution of the uncertainty or belief we assume the human has with respect to his position. *Bottom left:* Set of recorded searches, the trajectories are with respect to the hand. *Bottom right:* Trajectories starting from same area but have different search patterns, the red trajectories all navigate to the goal via the top right corner as opposed to the blue which go by the bottom left and right corner. Among these two groups there are trajectories which seem to minimize the distance taken to reach the goal as opposed to some which seek to stay close to the edge and corners.

In the standard PbD formulation of this problem, a parametrised function is learned, mapping from state x_t, which denotes the current position of the demonstrator's hand, to the hand's displacement $\dot{x}_t$. In our case since the environment is partially observable we have a belief or probability density function, $p(x_t|y_{0:t}, \dot{x}_{0:t})$, which is conditioned on all sensing information, $y_{0:t}$, (the subscript, $0:t$, indicates the time slice which ranges from, $t = 0$, to the current time, $t = t$) over the state space at any given point in time and the history of applied actions, $a_{0:t}$. We seek to learn this mapping, $f : p(x_t|y_{0:t}, \dot{x}_{0:t}) \mapsto \dot{x}_{t+1}$, from demonstrations. During each demonstration we record a set of variables consisting of the following:

- $\dot{x}_t \in \mathbb{R}^3$, velocity of the hand in Cartesian space, which is normalised.
- $\hat{x}_t = \arg\max_{x_t} p(x_t|y_{0:t}, \dot{x}_{0:t})$, the most likely position of the end-effector, or believed position.
- $U \in \mathbb{R}$, the level of uncertainty which is the entropy of the belief: $H\left(p(x_t|y_{0:t}, \dot{x}_{0:t})\right)$.

A statistical controller was learned from the tuple dataset: $\{(\dot{x}, \hat{x}, U)\}$ recorded during the search trials of the human subjects. Having described the experiment we proceed to give an in-depth description of the mathematical representation of the belief, sensing and motion models and the uncertainty.

Belief model

A human's belief of his location in an environment can be multi-modal or uni-modal, Gaussian or non-Gaussian and may change from one distribution to another. We chose a particle filter to be able to represent such a wide range of probability distributions. A particle filter is a Bayesian probabilistic method which recursively integrates dynamics and sensing to estimate a posterior from a prior probability density. The particle filter has two elements. The first estimates a distribution over the possible next state given dynamics and the second corrects it through integrating sensing. Given a *motion model* $p(x_t|x_{t-1}, \dot{x}_t)$, and a *sensing model* $p(y_t|x_t)$, we recursively apply a prediction phase where we incorporate motion to update the state, and an update phase where the sensing data is used to compute the state's posterior distribution. The two steps are depicted below.

$$p(x_t|y_{0:t-1}, \dot{x}_{0:t}) = \int p(x_t|x_{t-1}, \dot{x}_t)\, p(x_{t-1}|y_{0:t-1}, \dot{x}_{0:t-1})\, dx_{t-1} \quad (3.1)$$

$$p(x_t|y_{0:t}, \dot{x}_{0:t}) = \frac{p(y_t|x_t)p(x_t|y_{0:t-1}, \dot{x}_{0:t})}{p(y_t|y_{0:t-1})} \quad (3.2)$$

The probability distribution over the state $p(x_t|y_{0:t}, \dot{x}_{0:t})$ is represented by a set of weighted particles which represent hypothetical locations of the end-effector and their density which is proportional to the likelihood. The particular particle filter used was the *Regularised Sequential Importance Sampling* (Arulampalam et al., 2002, p.182). From previous literature Bake et al. (2011) it has been shown that there is a similarity between Bayes update rule and the way humans integrate information over time. Under this assumption we hypothesise that if the initial belief of the human is known then the successive update steps of the particle filter should correspond to a good approximation of the next beliefs.

Sensing model

The sensing model tells us the likelihood, $p(y_t|x_t)$, of a particular sensation y_t given a position $x_t \in \mathbb{R}^3$. In a human's case, the sensation of a curvature indicates the likelihood of being near an edge or a corner. However the likelihood cannot be modelled using the human's sensing information. Direct access to pressure, temperature and such salient information is not available. Real sensory information needs to be matched against virtual sensation at each hypothetical location x_t of a particle. Additionally, for the transfer of behaviour from human to robot to be successful, the robot should be able to perceive the same information as the human, given the same situation. An approximation of what a human or robot senses can be inferred, based on the end-effector's distance to particular features in the environment. In our case four main features are present, namely corners, edges, surfaces and an additional dummy feature defining no contact, air. The choice of these features is prior knowledge given to our system and not extracted through statistical analysis of recorded trajectories. The sensing vector is $y_t = [p_c, p_e, p_s, p_a]$, where p refers to probability and the subscript corresponds to the first letter of the feature it is associated with. In Equation 3.3, the sensing function, $h(x_t, x_c)$, returns the probability of sensing a corner, where $x_c \in \mathbb{R}^3$ is the Cartesian position of the corner which is the closest to x_t.

$$p_c = h(x_t, x_c; \beta) = \exp\left(-\left(\beta \cdot \|x_t - x_c\|\right)^2\right) \tag{3.3}$$

The exponential form of the function, h, allows the range of the sensor to be reduced. We set $\beta > 0$ such that any feature which is more than 1cm way from the end effector or hand has a probability close to zero of being sensed. The same sensing function is repeated for all feature types.

The sensing model takes into account the inherent uncertainty of the sensing function 3.3, and gives the likelihood, $p(y_t|x_t)$ of a position. Since the range of sensing is extremely small and entries are probabilistic we assume no noise in the sensor measurement. The likelihood of a hypothetical location, x_t, is related to Jensen-Shannon divergence (JSD), $p(y_t|x_t) = 1 - JSD(y_t||\hat{y}_t)$, between true sensing vector, z_t, obtained by the agent and that of the hypothetical sensation

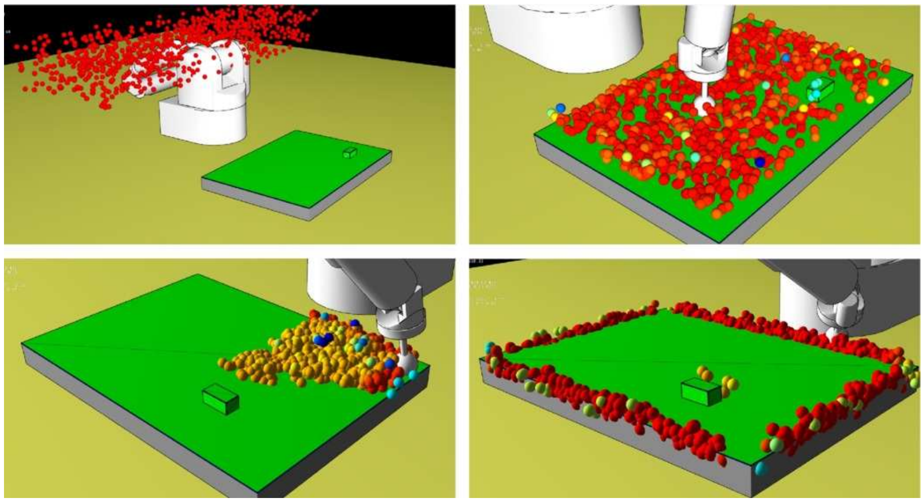


Figure 3.4: Four different time frames of the evolution of the belief particle filter. *Top left*: Initial belief distribution; a lot of uncertainty. *Top right:* First contact is made with the table, the measurement likelihood restrains the samples to be on the table's surface. *Bottom right:* First contact is an edge. *Bottom left:* Gradual localisation.

$\hat{y}_t$ generated at the location of a particle. In Figure 3.4, four different beliefs are shown.

Motion model

The motion model is straight forward compared with the sensing model. In the robot's case the Jacobian gives the next Cartesian position given the current joint angles and angular velocity of the robot's joints. From this the motion model is given by $p(x_t|x_{t-1}, \dot{x}_t) = \mathrm{J}(q)\dot{q} + \epsilon$ where q is the angular position of the robot's joints, $J(q)$ is the Jacobian and $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ is white noise. The robot's motion is very precise and its noise variance is very low. For humans, the motion model is the velocity of the hand movement provided by the tracking system. In our experiment we consider the noise from motion to be negligible. An increase in uncertainty already results from the re-sampling stage of Sampling Importance Resampling (SIR) particle filter and we found no need to add additional motion noise. The particles' positions were updated by applying the measured velocity obtained from either the visual tracking system (when recording the human demonstrations) or the robot's forward kinematics.

Uncertainty

In a probability distribution framework, entropy is used to represent uncertainty. It is the expectation of a random variable's total amount of unpredictability. The higher the entropy the more the uncertainty, likewise the lower the entropy, the less the uncertainty. In our context, a set of weighted

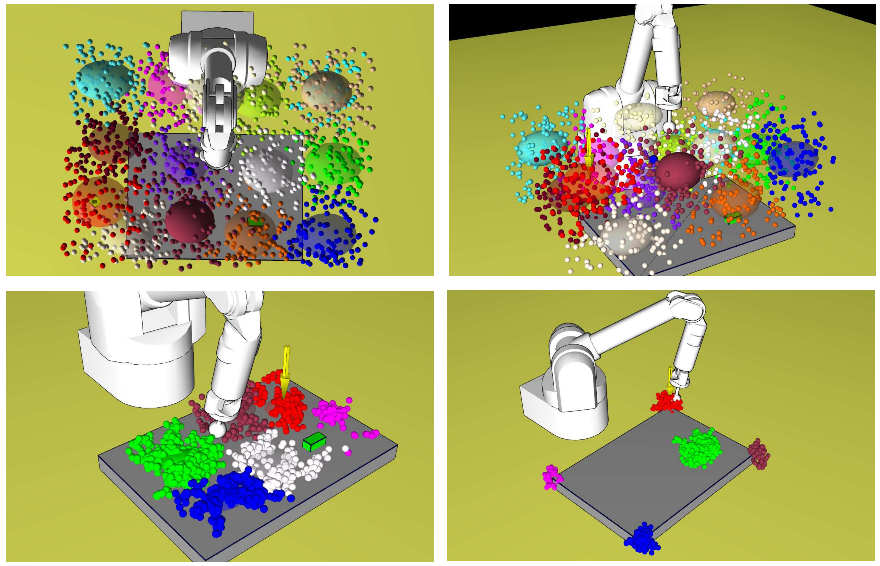


Figure 3.5: Representation of the estimated density function. *Top Left and Right:* Initial starting point, all Gaussian functions are uniformly distributed with uniform priors. The red cluster always has the highest likelihood which is taken to be the believed location of the robot's/human's end-effector. *Bottom Left:* Contact with the table has been established, the robot location differers from his belief. *Bottom Right:* Contact has been made with a corner, the clusters reflect that the robot could be at any corner (note that weights are not depicted, only cluster assignment).

samples $\{w_i, x_i\}^{i=1\ldots N}$ replaces the true probability density function of the belief, $p_u(x_t|y_{0:t}, \dot{x}_{0:t})$. A reconstruction of the underlying probability density is achieved by fitting a Gaussian Mixture Model (GMM), Equation 3.4, to the particles,

$$p_u(x_t|y_{0:t}, \dot{x}_{0:t}\,; \{\pi, \mu, \Sigma\}) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(x_t\,; \mu_k, \Sigma_k) \tag{3.4}$$

where K is the number of Gaussian components, the scalar π_k represents the weight associated to mixture component k (indicating the component's overall contribution to the distribution) and $\sum_{k=1}^{K} \pi_k = 1$. The parameters μ_k and Σ_k are the mean and covariance of the normal distribution k.

The main difficulty here is determining the number of parameters of the density function in a computationally efficient manner. We approach this problem by finding all the modes in the particle set via mean-shift hill climbing and set these as the means of the Gaussian functions. Their covariances are determined by maximizing the likelihood of the density function via Expectation-Maximization (EM).

Given the estimated density we can compute the upper bound of the differential entropy Huber et al. (2008), H,

$$H\left(p_u(x_t|y_{0:t}, \dot{x}_{0:t}\,; \{\pi, \mu, \Sigma\})\right) = \sum_{k=1}^{K} \pi_k \left(-\log(\pi_k) + \frac{1}{2}\log((2\pi e)^D |\Sigma_k|) \right) \tag{3.5}$$

where e is the base of the natural logarithm and D the dimension (being 3 in our case).

The reason for using the upper bound is that the exact differential entropy of a mixture of Gaussian functions has no analytical solution. When computing both the upper and lower bounds it was found that the difference between the two was insignificant, making any bound a good approximation of the true entropy. The choice of the believed location of the robot/human end-effector is taken to be the mean of the Gaussian function with the highest weighted π.

$$\hat{x}_t = \arg\max_{x_t} p_u(x_t|z_{0:t}\,; \{\pi, \mu, \Sigma\}) = \mu_{(k=\max(\pi))} \tag{3.6}$$

Figure 3.5 depicts different configurations of the modes (clusters) and believed position of the end-effector (indicated by a yellow arrow).

3.5 Policies

3.5.1 Modelling human search strategies

During the experiments, the recorded trajectories show that different actions are present for the same belief and uncertainty making the data multi-modal (for a particular position and uncertainty different velocities are present). That is multiple actions are possible given a specific belief. This results in a one-to-many mapping which is not a valid function, eliminating any regression technique which directly learns a non-linear function. To accommodate this fact we use a GMM to model the human's demonstrated searches, $\{(x, \dot{x}, U)\}$. Using statistical models to encode control policies in robotics is quite common, see Billard et al. (2008b).

By normalising the velocity the amount of information to be learned was reduced. We also took into consideration that velocity is more specific to embodiment capabilities: the robot might not be able to reproduce safely some of the velocity profiles demonstrated.

The training data set comprised a total of 20'000 tuples $(\dot{x}, \hat{x}, U)$, from the 150 trajectories gathered from the demonstrators. The fitted GMM $p_s(\dot{x}, \hat{x}, U)$ had a total of 7 dimensions, 3 for direction, 3 for position and 1 scalar for uncertainty. The definition of the GMM is presented below in equation 3.7.

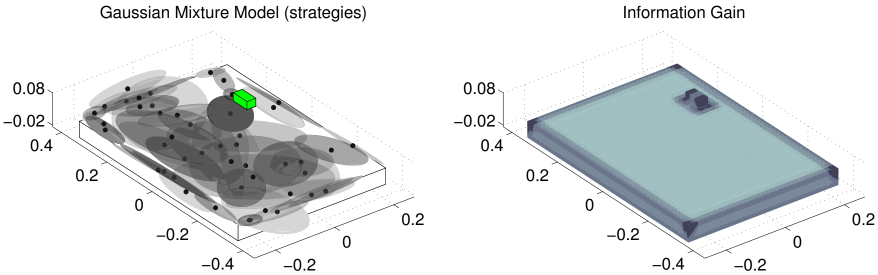


Figure 3.6: *Left:* Resulting search GMM, a total of 67 Gaussian mixture components are present. We note the many overlapping Gaussians: this results from the level of uncertainty over the different choices taken. For example humans follow along the edge of the table in different directions and might leave the edge once they are confident with respect to their location. *Right:* Information Gain map of the table environment, dark regions indicate high information gain as oppose to lighter ones. Not surprisingly, the highest are the corners, followed by the edges.

$$p_s(\dot{x}, \hat{x}, U\,; \{\pi, \mu, \Sigma\}) = \sum_{k=1}^{K} \pi_k \cdot \mathcal{N}(\dot{x}, \hat{x}, U\,; \mu_k, \Sigma_k) \tag{3.7}$$

$$\mu_k = \begin{bmatrix} \mu_{\dot{x}} \\ \mu_{\hat{x}} \\ \mu_U \end{bmatrix} \Sigma_k = \begin{bmatrix} \Sigma_{\dot{x}\dot{x}} & \Sigma_{\dot{x}\hat{x}} & \Sigma_{\dot{x}U} \\ \Sigma_{\hat{x}\dot{x}} & \Sigma_{\hat{x}\hat{x}} & \Sigma_{\hat{x}U} \\ \Sigma_{U\dot{x}} & \Sigma_{U\hat{x}} & \Sigma_{UU} \end{bmatrix}$$

Given this generative representation of the humans' demonstrated searches we proceeded to select the necessary parameters to correctly represent the data. This step is know as model selection and we used Bayesian Information Criterion (BIC) to evaluate each set of parameters which were optimised via Expectation-Maximisation (EM).

A total of 83 Gaussian functions were used in the final model, 67 for trajectories on the table and 15 for those in the air. In Figure 3.6 *(left)* we illustrate the model learned from human demonstrations where we plot the 3 dimensional slice (the position) of the 7 dimensional GMM to give a sense of the size of the model.

3.5.2 Coastal Navigation

Coastal navigation Roy et al. (1999) is a path planning method in which the objective function, Equation 3.8, is composed of two terms.

$$f(x_{0:T}) = \sum_{t=0}^{T} \lambda_1 \cdot c(x_t) + \lambda_2 \cdot I(x_t) \tag{3.8}$$

The first term, $c(x_t)$, is the traditional "cost to go" which penalizes every step taken so as to ensure that the optimal path is the shortest. The value was simply set to 1 for all discrete states in our case. The second term, $I(x_t)$, is the information gain of a state. The information gain, I, of a particular state is related to how much the entropy of a probability density function (pdf), being the location's uncertainty in our case, can be reduced. The two λ's are scalars which weigh the influence of each term.

In our table environment we discretised the state space, $\mathbb{R}^3$, into bins so as to have a resolution of approximately, $1cm^3$, giving us a total of a 125'000 states. The action space was discretised to 6 actions, two for each dimension meaning that all motion is parallel to the axis. For each state, x_t, an $I(x_t)$ value is computed by evaluating Equation 3.9,

$$I(x_t) = \mathbb{E}_{p(y_t|x_t)}\{H(p_u(x_t|y_{0:t}, \dot{x}_{0:t})\} - H(p_u(x_t|y_{0:t-1}, \dot{x}_{0:t})) \tag{3.9}$$

which is essentially the difference between the entropy of a prior pdf to that of a posterior pdf. We set our initial pdf to be uniformly distributed and we computed the maximum likelihood sensation for each discrete state x_t which is akin to the expected sensation or assuming that there is no uncertainty in sensor measurement (an assumption often made throughout the literature to avoid carrying out the integral of the expectation in Equation 3.9). The result is the difference between the posterior pdf, given that the sensation occurred in x_t, and the prior pdf. The resulting cost map is illustrated in Figure 3.6. As expected, corners have the highest information gain followed by edges and surfaces. We do not show the values of the table since they provided much less information gain.

The optimization of the objective function is accomplished by running the Dijkstra's algorithm. This algorithm, given a cost map, computes the shortest path to a specific target from all the states. This results in a policy.

3.5.3 Control

The standard approach to control with a GMM is to condition on the state, $\hat{x}_t$ and U_t in our case, and perform inference on the resulting conditional GMM, Equation 3.10, which is a distribution over velocities or directions.

$$p_s(\dot{x}|\hat{x}, U) = \sum_{k=1}^{K} \pi^k_{\dot{x}|\hat{x},U} \cdot \mathcal{N}\left(\dot{x}\,; \mu^k_{\dot{x}|\hat{x},U}, \Sigma^k_{\dot{x}|\hat{x},U}\right) \tag{3.10}$$

The new distribution is of the dimension of the output variable, the velocity (dimension 3). The variable $\dot{x}$ in $\dot{x}|\hat{x}, U$ indicates the predictor variable and the variables $\hat{x}, U$ have been conditioned. A common approach in statistical PbD methods using GMMs is to take the expectation of the conditional (known as Gaussian Mixture Regression), equation 3.11

$$\dot{x} = \mathbb{E}\{p_s(\dot{x}|\hat{x}, U)\} = \sum_{k=1}^{K} \pi^k_{\dot{x}|\hat{x},U} \cdot \mu^k_{\dot{x}|\hat{x},U} \tag{3.11}$$

The problem with this expectation approach, is that it averages out opposing directions or strategies and may leave a net velocity of zero. One possibility would be to sample from the conditional, however this can lead to non-smooth behaviour and flipping back and forth between modes resulting in no displacement. To maintain consistency between the choices and avoid random switching we perform a weighted expectation on the means so that directions (modes) similar to the current direction of the end-effector receive a higher weight than opposing directions. For every mixture component k, a weight α_k is computed based on the distance between the current direction and itself. If the current direction agrees with the mode then the weight remains unchanged but if it is in disagreement a lower weight is calculated according to the equation below.

$$\alpha_k(\dot{x}) = \pi^k_{\dot{x}|\hat{x},U} \cdot \exp(-\cos^{-1}(< \dot{x}, \mu^k_{\dot{x}|\hat{x},U} >)) \tag{3.12}$$

Gaussian Mixture Regression is then performed with the normalised weights α instead of π (the initial weight obtained when conditioning).

$$\dot{x} = \mathbb{E}_\alpha\{p_s(\dot{x}|\hat{x}, U)\} = \sum_{k=1}^{K} \alpha_k(\dot{x})\, \mu^k_{\dot{x}|\hat{x},u} \tag{3.13}$$

The final output of equation 3.13 gives the desired direction ($\dot{x}$ is re-normalised). In the case when the mode suddenly disappears (because of sudden change of the level of uncertainty caused by the appearance or disappearance of a feature) another present mode is selected at random. For example, when the robot has reached a corner, the level of uncertainty for this feature drops to zero. A new mode, and hence new direction of motion, will then be computed. However this is not enough to be able to safely control the robot. One needs to control the amplitude of the velocity and ensure compliant control of the end-effector when in contact with the table. This behaviour is not learned here, as this is specific to the embodiment of the robot and unrelated to the search strategy. The amplitude of the velocity is computed by a proportional controller based on the believed distance to the goal,

$$\nu = \max(\min(\beta_1, K_p(x_{\mathrm{g}} - \hat{x}), \beta_2) \tag{3.14}$$

where the β's are lower and upper amplitude limits, x_g is the position of the goal, and K_p the proportional gain which was tuned through trials.

As mentioned previously, compliance is the other important aspect when having the robot duplicate the search strategies. Collisions with the environment occur as a result of the uncertainty. To avoid risks of breaking the table or the robot sensors we have an impedance controller at the lowest level which outputs appropriate joint torques τ. The overall control loop is depicted in Figure 3.7.

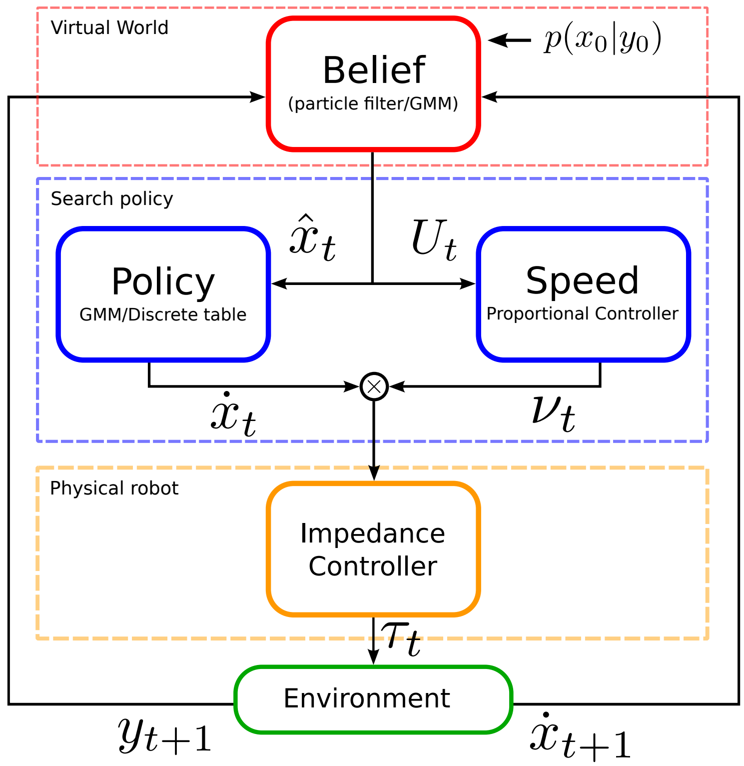


Figure 3.7: Overview of the decision loop. At the top a strategy is chosen given an initial belief $p(x_0|y_0)$ of the location of the end-effector (initially through sampling the conditional). A speed is applied to the given direction based on the believed distance to the goal. This velocity is passed onwards to a low level impedance controller which sends out the required torques. The resulting sensation, encoded through the Multinomial distribution over the environment features, and actual displacement are sent back to update the belief.

3.6 Results and discussion

Throughout our evaluation of our GMM PbD-POMDP control policy we will be considering four search policies: Greedy, GMM, Hybrid and Coastal. We evaluate behaviour present in the human demonstrations, and the four above mentioned policies in terms of their riskiness. We qualitatively compare the policies of the GMM model and the Coastal Navigation algorithm and highlight the effect of uncertainty. We finish with a quantitative evaluation of search efficiency in terms of distance travelled until the goal is found. The layout of this section follows as:

- Section 3.6.1, we analyse the types of behaviour present in the human demonstration as well as in four different search algorithms: Greedy, GMM, Hybrid and Coastal.

- Section 3.6.2, we qualitatively analyse the GMM search policy (namely the different modes/decisions present) with respect to the Coastal navigation policy.

- Section 3.6.3, we evaluate the search performance, with respect to the distance taken to reach the goal and the uncertainty profiles towards the end of the searches in 5 different experiments (different types of initializations).

3.6.1 Search & behaviour analysis

For each method (Greedy, GMM, Hybrid, Coastal) 70 searches were performed with all starting positions drawn from the uniform distribution used during the teaching stage (depicted in Figure 3.3 *top right*, page 52). In Figure 3.8 we illustrate the expected sensation $\mathbb{E}\{y\}$ and variance $\mathrm{Var}\{y\}$ for each trajectory with respect to the edge and corner of the table.

The selection of edges and corners as features as a means of classifying the type of behaviours present is not solely restricted to our search task. Salient landmarks will result in a high level of information gain, which is the case for the edge and corner (see Figure 3.6 *right*, page 58). Other tasks can use such features or variants in which the curvature is considered for representing the task space. These features are present in most settings and high level features can use these easily as their building blocks.

We note that the Greedy search approach seeks to go directly to the goal without taking into account the uncertainty. The GMM models human search strategies. The Hybrid is a combination of both the Greedy and GMM method where once the uncertainty has been sufficiently minimised, the policy switches (threshold) to the Greedy method for the rest of the search. The Coastal navigation algorithm finds the optimal path to the goal based on an objective function

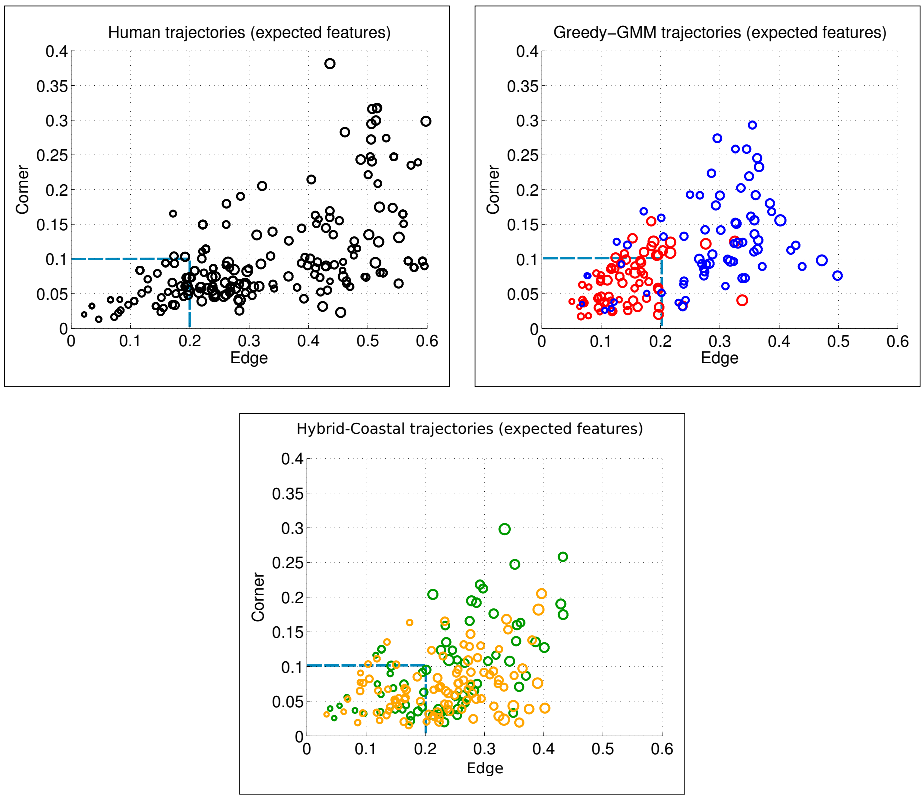


Figure 3.8: Expected sensation. Plots of the expected sensation of the edge and corner feature for all trajectories. The axes are associated with the sensor measurements, 0 means that the corresponding feature is not sensed and 1 the feature is fully sensed. A point in the plots summarises a whole trajectory by the mean and variance of the probability of sensing a corner or edge. The radius of the circles are proportional to the variance. The doted blue rectangle represents the decision boundary for classifying a trajectory as being either risk-prone or risk-averse. A point which lies inside the rectangle is risk-prone. *Left:* Human trajectories demonstrate a wide variety of behaviours ranging from those remaining close to features to those preferring more risk. *Right:* Red points show Greedy and blue points the GMM model. *Bottom:* Green circles are associated with the Hybrid method whilst orange are those of the Coastal navigation method. The Hybrid method is a skewed version of the GMM which tends towards risky behaviour and exhibits the same kind of behaviour as the Coastal algorithm.

Criteria	**Greedy**	**GMM**	**Hybrid**	**Coastal**	**Human**
risk-prone (f)	77 %	11 %	30 %	46 %	26 %
risk-prone (r)	78 %	12 %	24 %	45 %	7 %

Table 3.1: Percentage of risk-prone trajectories based on two decision criteria, the feature (f) and the risk (r) (information gain) metrics discussed above.

which consists of a trade-off between time taken to reach the goal and the minimisation of the uncertainty.

It can be seen that the human demonstrations have a much wider spread than those of the search algorithms. We suggest that this is due to human behaviours being optimal with respect to their own criteria as opposed to the algorithms which usually tend to only maximise a single objective function. The trajectories of the Greedy and GMM methods represented by their expected features demonstrate two distinctive behaviours (in terms of expected sensation), risk-prone for the Greedy and risk-adverse for the GMM.

We make **the assumption** that Greedy trajectories are risk-prone by nature. We performed a SVM classification on the Greedy-GMM expected features (Figure 3.8 *right*) and used the result to construct a decision boundary as a means of classifying a trajectory as being either risk-prone or risk-averse. Table 3.1 *first row* shows that the GMM and Human search trajectories are mostly risk-averse. Surprisingly the Coastal policy seems to be very risk-prone given that it seeks paths close to highly informative areas. We use a second metric based on the information gain, which we call the Risk factor, to classify trajectories as being either risk-prone or risk-averse.

The Risk factor of each individual trajectory is inversely proportional to its accumulated information gain. Figure 3.9 (*left*) shows the kernel density estimation distribution of the risk for each search method. Two trajectories per search type corresponding to a supposed risk-prone and risk-averse search are plotted in the expected feature space in Figure 3.9 (*right*). As expected, risk-prone strategies for which the risk tends to 1 have a low expectation of sensing edges and corners and produce trajectories with a low information gain while those with a high expectation of sensing features have a high information gain. Since the metric lies exclusively in the range [0,1] we define that every trajectory which has a Risk factor lower than than 0.5 will be considered risk-averse whilst those above are risk-prone. Table 3.1 *second row* illustrates the riskiness of each search method. It is evident that humans are risk-averse in general followed by GMM which is a smoothing of the human data, then Hybrid which as expected should be more risk-prone since it is a linear interpolation between the GMM and Greedy search policies and finally Coastal and Greedy.

Figure 3.10 (*top left & right*), shows risk-prone (red) and averse (green) trajectories produced by human demonstrations and by the Greedy search. Both these extremes correspond to our intuition that risk-averse trajectories tend to remain closer to features or areas of high information gain as oppose to risk-prone

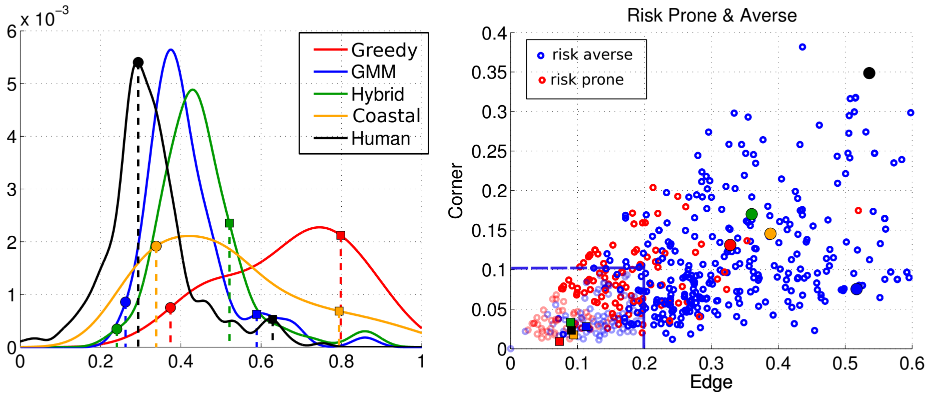


Figure 3.9: Risk of searches. Illustration of risk-prone and risk-averse searches in terms of a Risk factor (*left*) and expected sensation (*right*). *Left:* Each trajectory was reduced to a single scalar, which we call the Risk factor, quantifying the risk of a trajectory. The Risk factor is inversely proportional to the sum of the information gain of a particular trajectory. The colour paired dots (risk averse) and squares (risk prone) represent trajectories which are plotted in Figure 3.10, to illustrate that these correspond to risk averse and prone searches. *Right:* Corresponding trajectories chosen in the Risk factor space but represented in the feature space. As expected, trajectories with a high risk map to regions of low expected feature. However the transition from the Risk space to feature space is non-linear and will result in a different risk-level classification than the feature metric previously discussed.

searches. However to stress the case that humans have multiple search strategies present, we performed 40 GMM searches (model of the human behaviour) which all started under the same initial conditions (same belief distribution, true position and believed position). Figure 3.10 shows the resulting trajectories and expected features for each trajectory. It is clear that multiple searches occur which is reflected in the plot of the expected features. All of the search strategies generated by the GMM for this initial condition produced risk-averse trajectories.

We conclude that there is a strong inclination towards inferring that indeed multiple search strategies do arise in the human searches since they were extracted and encoded in the GMM model. From the risk distribution, humans have a tendency to be risk-averse.

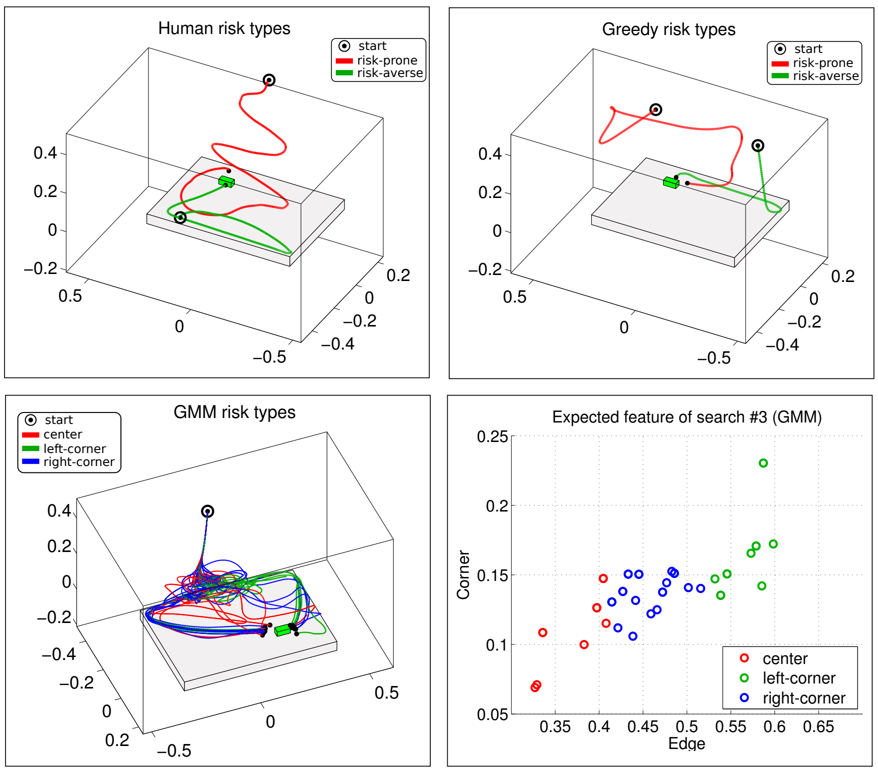


Figure 3.10: Risk prone & averse searches (red & green trajectories). *Top left:* Two human trajectories taken from data shown in Figure 3.9. *Top right:* Two Greedy trajectories. *Bottom left:* GMM trajectories, all starting from the same location, the colour coding is to illustrate the different policies which were encoded and emerge given the same initial conditions. *Bottom right:* Corresponding expected features of each trajectory, the colour coding matches the trajectories to the "GMM risk types" sub-figure. All the searches which were generated by the GMM for this initialisation produced risk-averse searches (based on the feature metric discussed previous).

3.6.2 GMM & Coastal Navigation policy analysis

We next illustrate some of the modes (action choices) present during simulation and evaluate their plausibility. Figure 3.11 shows that multiple decision points have been correctly embedded in the GMM model. All arrows (red) indicate directions that reduce the level of uncertainty.

Figure 3.12 depicts the vector fields of both Coastal and GMM models where, as expected, the Coastal navigation trajectories tend to stay close to edges and corners until they are sufficiently close to the goal. This is achieved by weighting the information gain term $I(x_t)$ in the objective function sufficiently (λ_2). If λ_2=0 the Coastal policy is the same Greedy algorithm.

It can be further seen that when the uncertainty tends towards it's maximum value ($U \to 1$) all behaviour tends to go towards the edges and corners. As the uncertainty reduces ($U \to 0$) the vector field tends directly towards the goal. However even at a low level of uncertainty, the behaviour at the edges and corners remains multi-modal and tends to favour remaining close to the edges and corners. This is an advantage of the GMM model. If the uncertainty has been sufficiently reduced and the true position of the end-effector or hand is not near an edge the policy dictates to go straight to the goal. This is not the case for the Coastal algorithm which ignores the uncertainty and strives to remain in the proximity of corners and edges until sufficiently close. This approach could potentially lead to unnecessary travel cost which could otherwise have been avoided.

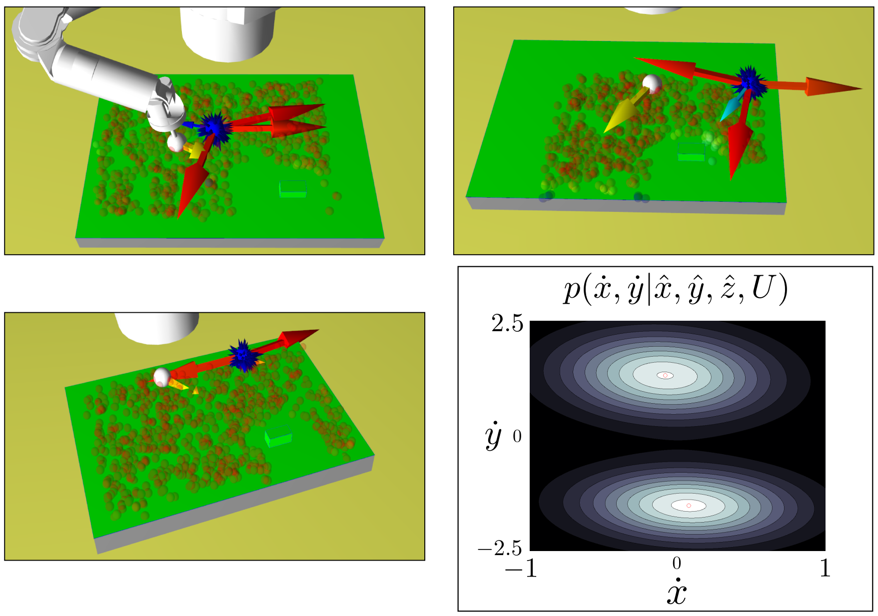


Figure 3.11: Illustration of three different types of modes present during the execution of the task where the robot is being controlled by the learned GMM model. The white ball represents the actual position of the robot's end-effector. The blue ball represents the believed position of the robot's end-effector and the robot is acting according to it. The blue ball arrows represent modes. Colours encode the mode's weights given by the priors π_k after conditioning (but not re-weighted as previously described). The spectrum ranges from red (high weight) to blue (low weight). *Top left:* Three modes are present, but two agree with each other. *Top right:* Three modes are again present indicating appropriate ways to reduce the uncertainty. *Lower left:* Two modes are in opposing directions. No flipping behaviour between modes occurs since preference is given to the modes pointing in the same direction as the robot's current trajectory. *Lower right:* GMM modes when conditioned on the state represented in the lower left figure. The two modes represent the possible directions (un-normalised).

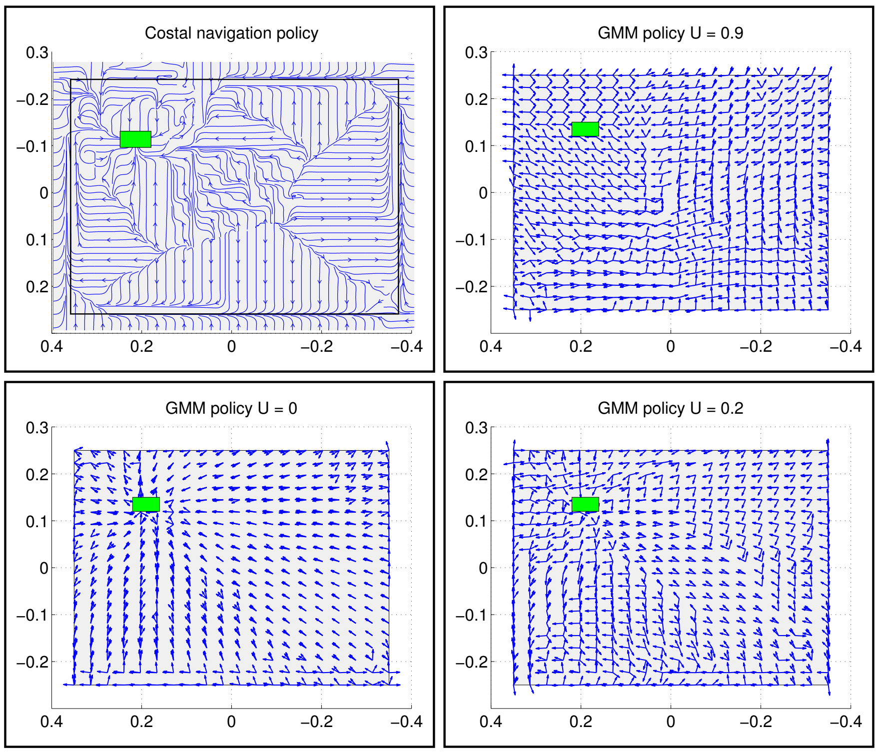


Figure 3.12: Illustration of the vector field for the Coastal and GMM policy. *Top Left* Coastal policy, there is only one possible direction for every state at any time, the values of λ_2 in the cost function were set experimentally. *Others:* The GMM policy for three different levels of uncertainty. For each point multiple actions are possible which is reflected by the number of arrows (only the first three most likely actions). As the uncertainty decreases the policy becomes less multi-model, but remains around the edges and corners. Note that once certain of being close to an edge there is a possibility to go either straight to the goal or stay close to the edge and corners.

3.6.3 Distance efficiency & Uncertainty

We seek to distinguish the most efficient method in terms of two metrics, the distance (in meters) taken to reach the goal and the level of uncertainty upon arriving at the goal. We report results on 5 different search experiments in which we compare the Greedy, GMM and Coastal Navigation algorithms. The Hybrid was not fully considered since it is a heuristic combination of the Greedy and GMM methods.

In the first experiment, the true and believed locations of the end-effector were drawn uniformly from the original start distribution (Figure 3.3, *top right*) reflecting the default setting. The initializations (both real and believed end-effector locations) for the remaining 4 experiments were chosen in order to reflect particular situations which highlight the differences and drawbacks between each respective search method. For the first experiment (Uniform search experiment), a 100 trials were carried out in which the end-effector position and belief were initialized uniformly. As for the other 4 search experiments, 40 separate runs were carried for each of the three algorithms.

Table 3.2 reports the mean and variance of the distance taken (in meters) to reach the goal for each search method for all 5 experiments. We report on an Analysis of Variance (ANOVA) to test that all experiments were significantly different from one another as were the searches. We test the null hypothesis, H_o, that there is no statistical difference between the 5 search experiments. Before performing the ANOVA, we verified that our dependent variable, distance [m] taken to reach the goal, follows a normal distribution for all methods and all experiments (a total of $5 \times 3 = 15$ tests), an assumption which is required by an ANOVA analysis. A Kolmogorov-Smirnov test was performed on each experiment and associated search method. A total of 11/15 searches rejected the null hypothesis with a significance level of less than 5% (p-value < 0.05).

In Table 3.3 we report the p-values and F-statistics for an ANOVA on the 5 different experiments where our null hypothesis is that all experiments produce statistically the same type of search. For all experiment types the p-value is extremely small, below a significance value of 1% (p-value < 0.01) which indicates that we can safely reject the null hypothesis and accept that all experiments

Experiment	**Greedy**	**GMM**	**Coastal**
Uniform	1.54 (0.46)	0.99 (0.14)	1.13 (0.57)
#1	3.02 (0.36)	1.82 (0.23)	3.44 (1.50)
#2	0.80 (0.01)	1.41 (0.14)	0.94 (0.01)
#3	1.14 (0.08)	1.80 (0.17)	2.14 (0.81)
#4	0.75 (0.04)	1.34 (0.07)	0.68 (0.01)

Table 3.2: Mean distance and (variance) taken to reach the goal for 3 methods in 5 experiments. The grey shaded entries correspond to the results of the search algorithm which obtained the fastest time to reach the goal in each type of experiment/search.

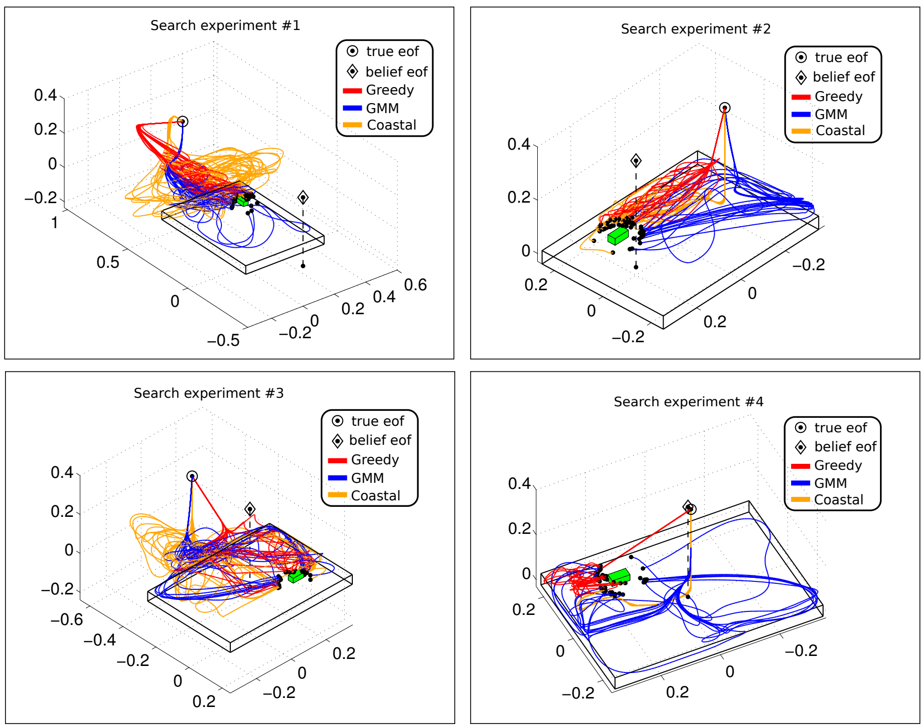


Figure 3.13: Four search initializations, from *top left* to *bottom right* we refer to them as #1-4. The circle indicates the true starting point of the end-effector (eof), whilst the triangle is the initial believed location of the eof. The initialisation in #1 was chosen such that the true and believed eof locations were at opposite sides of the table. This setting was selected to highlight the draw back in methods which do not take into account uncertainty. The second initialisation #2, reflects the situation where once again there is a large distance between true and believed location of the eof. However this time both are above the table. The starting points in #3 are a variant on #1 with the difference being that the believed eof position is above the table whilst the true eof location is not. The last experiment #4 was a setup which would be favourable to algorithms that are inclined to be greedy. Both true and believed eof locations are close to one another.

search method	Uniform	#1	#2	#3	#4
p-value (F)	2e-06 (14)	5e-07 (19)	7e-11 (36)	4-06 (15)	4e-16 (67)

Table 3.3: ANOVA tests the null hypothesis that all search experiments produced the same type of search with respect to the distance taken to reach the goal. All the p-values are extremely small which indicate that the null hypothesis can safely be rejected.

p-value (F)	Greedy vs GMM	Greedy vs Coastal	GMM vs Coastal
Uniform	3.59e-08 (30)	3.32e-04 (13)	1.90e-01 (2)
#1	5.80e-08 (46)	1.88e-01 (2)	4.58e-06 (28)
#2	3.60e-08 (47)	4.68e-04 (14)	4.54e-06 (28)
#3	3.57e-07 (37)	2.07e-05 (23)	1.25e-01 (2)
#4	6.70e-10 (64)	1.58e-01 (2)	6.34e-13 (107)

Table 3.4: ANOVA between paired search methods. The first column gives an indication of the probability that both the Greedy and GMM searches are statistically the same (the null hypothesis). This was rejected with a tolerance of below %1. In the second column, Greedy vs Coastal searches #1 and #4 are statistically closer than the rest with a p-value threshold of 10% required to be able to reject the null hypothesis. In the third column the uniform and #3 are not statistically different and would require a higher threshold on the p-value to be so.

produced very different searches, which is important for a comparative study.

As the first ANOVA only indicated that the experiments produced different searches, we also performed a second ANOVA test between the paired search methods to confirm that the methods themselves are statistically different. Table 3.4 illustrates the difference between the individual search methods for each experiment. It was found that most search algorithms produced significantly different searches (p-value < 0.01) with the exception of the GMM and Coastal algorithm for the Uniform and #3 experiment (p-value < 0.1). However the GMM and Coastal trajectories for the #3 experiment appear to be quite different when the trajectories are off the table's surface, see Figure 3.13 *(Bottom left)*, but share similar characteristics such as edge following behaviour.

From our ANOVA analysis we conclude that the behaviour exhibited by the three search strategies is significantly different. This is certainly the case for the Greedy and GMM methods, even though in certain situations the Greedy and Coastal policies display similar behaviour such as in experiment #1. The reason for this is that both the Greedy and Coastal policies start in a situation where there are no salient features available and their polices take the true end-effector location to an even more feature deprived region. In this situation the GMM policy is the clear winner with respect to the distance taken to reach the goal.

In experiment #2, both Greedy and Coastal policies perform equally well and will usually perform faster than the GMM model if the true and believed locations of the end-effector remain on the surface of the table. Otherwise if this is not the case, they will both reduce the uncertainty in a very inefficient way as the modes will often change during the search. This leads to the believed position (most likely state, $\hat{x}_t$) varying greatly, resulting in an increased time before the uncertainty has been narrowed down sufficiently for a contact to occur

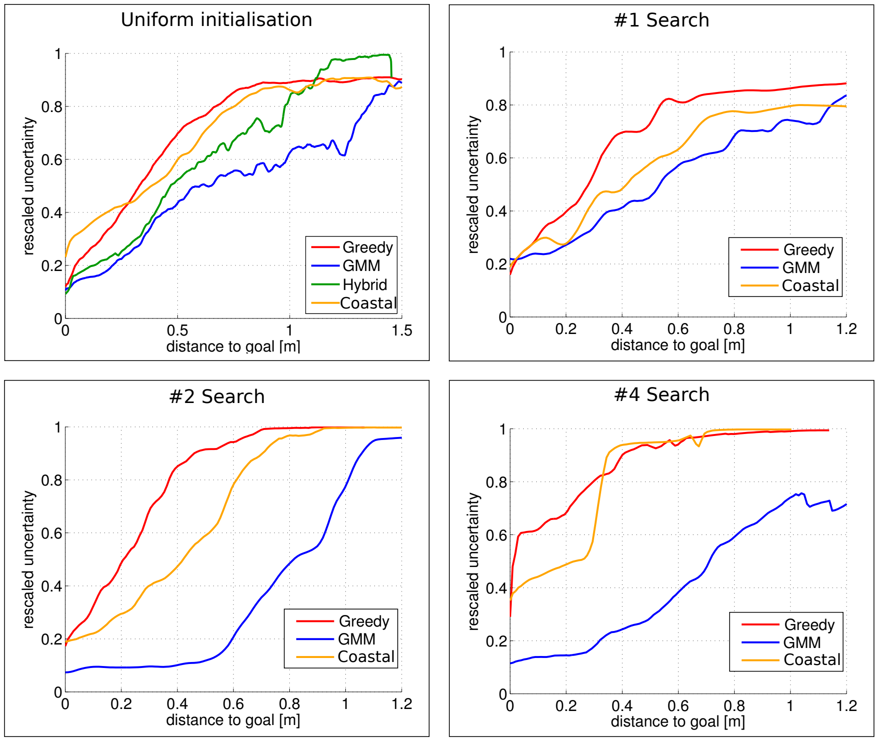


Figure 3.14: Reduction of the uncertainty for the Uniform, #1, #2 and #4 experiment, the expected value is reported *Top left*: Uniform initialisation, expected uncertainty for the Greedy (red), GMM (blue), Hybrid (green) & Coastal (orange) search strategies. *Top right:* Experiment #1. *Bottom left:* Experiment #2. *Bottom right:* Experiment #4.

with the table (or simply by chance).

Figure 3.14 shows the normalised uncertainty with respect to the distance remaining to the goal for all experiments, (#3 is excluded being similar to the #2).

The results show which methods actively minimise the uncertainty and which methods find the goal whilst being more dependent on chance. For all the reported experiments the GMM (learned from human searches) reaches a lower expected uncertainty than all other search algorithms. For the Uniform and #1 search experiment, all methods reach the same final uncertainty level. However, for the #2 and #4 experiments, the GMM reaches the goal with significantly lower uncertainty. It is inferred that the GMM model actively minimises the uncertainty which is also reflected in the distance it takes reach the goal in comparison with the other methods.

While the Greedy (#2) and Coastal (#4) are faster than the GMM method, Table 3.2, both have a far higher level of uncertainty at the arrival which leads to the assumption that chance has a non-negligible effect on their success.

3.7 Conclusions

In this work we have shown a novel approach in teaching a robot to act in a partially observable environment. Through having human volunteers demonstrate the task of finding an object on a table, we recorded both the inferred believed position of their hand and associated action (normalised velocity). A generative model mapping the believed end-effector position to actions was learned, encapsulating this relationship. As speculated and observed, multiple strategies are present given a specific belief. This can be interpreted as the fact that humans act differently given the same situation.

The behaviour recorded from the human demonstrations, encoded as set of expected sensations, showed the presence of trajectories which both remained near to the edge and corner features but also trajectories which remained at a distance. Risk-prone and risk-averse behaviour was further confirmed by the overlap of the risk factor of Human and GMM generated trajectories with that of the Greedy risk factor. According to the feature-based factor, more than 70% of the human search trajectories were considered to be risk-averse whilst 93% according to the Risk factor. Similarly the GMM search trajectories showed to be 89-88% risk-averse.

In terms of the comparative study, the GMM controller is more adapted to dealing with situations of high uncertainty and accounts for it better than Greedy or Coastal planning approaches. This is evident in the experiment where the believed position and true position of the end-effector were significantly far apart and distant from salient areas. Future questions of scientific value to be addressed are to which extent do humans follow the reasoning of a Markov Decision Process in a partially observable situation where the state space is continuous (the problem has been partially addressed in Bake et al. (2011) for discrete states and actions).

A drawback of the PbD-POMDP approach is that the quality of the learned policy is dependent on the abilities of the human teacher. If the teacher is good (on average) then the transferred policy will be adequate, if however the human is suboptimal at performing the task, then the resulting policy will be poor. An autonomous way of evaluating the quality of the demonstrations whilst learning a policy is necessary. In the next chapter, "Chapter 4", we demonstrate that by introducing a cost function and using a Reinforcement Learning approach we can account for poor demonstrations and increase the quality of the policy.

Chapter 4

Peg in hole

In Chapter 3, we demonstrated that we could learn a search policy from demonstrations of human teachers for a task consisting of locating a wooden object on a table and successfully transferring it to a robot apprentice. The motivation behind our approach is that the intuition and knowledge exhibited by the human teachers, during the search, consists of a good balance between exploration and exploitation actions which then can be encapsulated in a generative Gaussian Mixture Model (GMM) and subsequently used as a control policy. The approach is satisfactory if we are interested in extracting different behaviours present across the human teachers and reproducing them. If our objective is however to learn a policy with a unique behaviour which is optimal or at least close to optimal, then using the approach detailed in Chapter 3, as it is, will not necessarily result in a efficient policy. For the GMM will model both good patterns exhibited by the human teachers and their mistakes. If the task is difficult and many possible solutions exist, such as was the case in the blindfolded search task of the previous Chapter, many demonstrations will be necessary for search patters to be present and encoded in the GMM. Otherwise it would have to be combined with another policy as showed in Chapter 3, for our Hybrid GMM-Greedy policy. There the task undertaken is implicitly encoded, as there is no cost function which is optimised, in the GMM and as a result the taught behaviour has to be goal oriented and consistent, which is not always the case in a blindfolded search task.

To overcome the above mentioned limitations, in this Chapter we use a binary cost function as means of ranking demonstrations provided by the human teachers. We combine our PbD-POMDP approach with an Actor-Critic Reinforcement Learning (RL) framework which is close to Fitted-Value Iteration(FVI) and other experience replay methods, which we will refer to as RL-PbD-POMDP. Our objective is to avoid noisy explorative rollouts, common to all RL approaches and is their Achille's heel, and only rely on the data provided by the human teachers. We want to avoid any autonomous exploration common in RL for three reasons. Firstly it is time consuming and is typically only applicable to RL problems in which an exhaustive exploration of the entire state or parameter space is feasible, such as in traditional RL problems like the inverted pendulum or mountain cart. The universal exploration method, used throughout RL, is state independent (sometimes state dependent) white noise which results in an entire exploration of the state space. This is neither practical or feasible for the type of search problems we are considering. The second reason is that the exploration cannot be random as we are using a physical robotic system, and this would be dangerous. The third and most important reason is that we want to use the same amount of information for our RL-PbD-POMDP as we used for the PbD-POMDP approach. This is to strongly highlight the fact that we can obtain an improved policy without the need of any additional information

We analyse our RL-PbD-POMDP approach on a power-socket Peg-in-Hole (PiH) search task. In this task, human teachers must demonstrate to a robot apprentice how to search for and connect a plug to a power socket. The first component of the task, the search for the socket, is similar to the table-wooden block setup in the previous Chapter. Here the connection of the plug to the power socket, the PiH component, which requires a higher level of precision to be able to achieve.

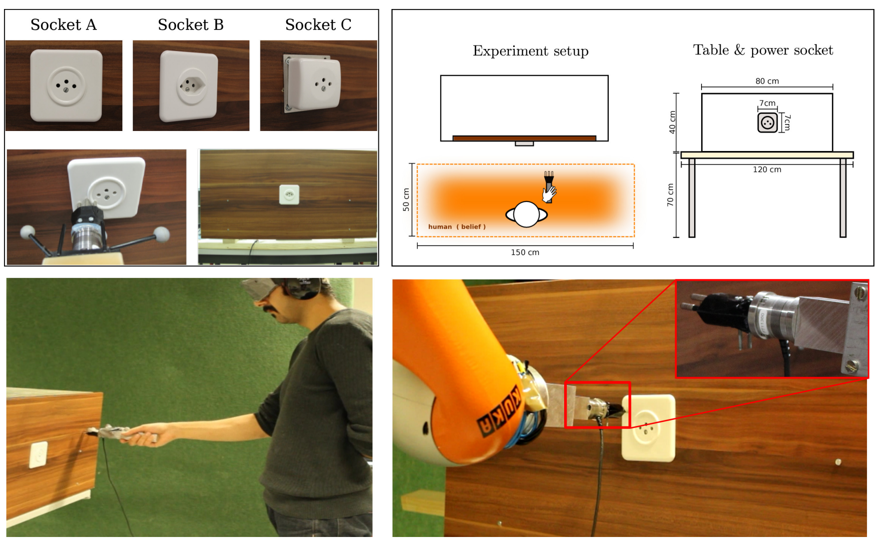


Figure 4.1: *Top-right:* The experimental setup. Blindfolded human teachers stand in the orange rectangle always facing the wall. Human teachers are informed of their starting location and told that they would always be facing the wall. *Top-left:* Three different power sockets. The plug is connected to a cylinder, to make it easy to hold for the human teachers. An ATI 6-axis force torque sensor (Nano25 Series) is between the cylinder and the plug. *Bottom-left* Human teacher performing the PiH search task. The human teachers wear goggles to remove sight and ear defenders to greatly diminish hearing.*Bottom-right:* The KUKA LWR robot equipped with the same force torque sensor and plug used by the human teachers.

In Figure 4.1 (*Top-right*), we illustrate the setup of the search task. The diagram shows the distribution of the initial starting position (in orange) of the human teachers. As was the case in our previous search experiment in Chapter 3, the teacher is disoriented before each trial but knowns that his initial heading will be the same, facing the wall. The human teachers are deprived of visual information (they are blindfolded) and their hearing sense is significantly reduced (by wearing ear defenders). We consider one type of plug, Type J[1], and three different power sockets. Power *socket A*, has a ring around its holes, *socket B* has a funnel, which we hypothesize should make it easier to connect, and *socket C* has a flat elevated surface. See Figure 4.1 (*Top-left*) for an illustration. The plug held by the human teachers is attached to a cylindrical handle with an ATI 6 axis force torque sensor (Nano25 [2]) fixed between the two. On top of the cylinder is a set of markers from which a motion capture system (OptiTrack[3]) provides both position and orientation, see Figure 4.1 (*Top-left*). In the *Bottom-left* quadrant we can see an example of a human teacher trying to accomplish the search and connection task, and to the *Bottom-right* the KUKA LWR robot apprentice is reproducing a search policy learned from the teacher.

We found that by learning a value function over the belief space using approximate dynamic programming (part of FVI) and using this as a Critic to update the parameters of our GMM policy (Actor) we were able to achieve an important improvement over our previous PbD-POMDP approach. We performed evaluations both in simulation and on the KUKA LWR robot where we tested policies ability to generalise to sockets for which no training data was provided and different socket locations. In all our evaluations the RL-PbD-POMDP approach proved to be always better. More importantly we demonstrate that the RL-PbD-POMDP approach performs significantly better when we use the training data from the worst teacher, which mitigates the **original assumption** that the teachers have to be consistently efficient at the task.

4.1 Background

4.1.1 Peg-in-hole

The Peg-in-Hole (PiH) task is one of the most widespread steps in industrial assembly and manipulations processes, with examples including the assembly of vehicular transmission components Chhatpar and Branicky (2001) and valves Cheng and Chen (2014). To be successful, the estimated position of the robot's end-effector and workpiece must be precise. Typically, the clearance the between and plug and the workpiece's hole is very small leaving little room for error. As a result, variations in the assembly's components in combination with position uncertainty can result in either jamming during the insertion process or in failure of the plug finding the hole. This created a need for adaptive search and insertion policies for PiH, which has been driving research in this area.

[1] http://www.iec.ch/worldplugs/typeJ.htm
[2] http://www.ati-ia.com/products/ft/sensors.aspx
[3] http://www.optitrack.com/

From the literature, we identified the different components in PiH solutions. All approaches use to some extent a vision system to estimate the position of the workpiece. Given an estimate of the workpiece's position, a common approach is to follow either a blind increasing spiral Cartesian trajectory or parametrised policies which guarantee that all positions on the workpiece have been visited. To increase the chances of a connection these approaches use a compliant controller which usually includes a hybrid force/position controller. The second predominant approach (which has been confined to academic circles) follows the data driven Programming by Demonstration (PbD) framework. Teleoperated or kinesthetic demonstrations of a human teacher are recorded and a policy is learned and fine-tuned so as to reproduce the same (F)orce/(T)orque profile as that demonstrated by the human teacher. The first approach does not consider reproducing the F/T profile but rather follows a position trajectory whilst being compliant.

In Meeussen et al. (2010) a PR2 executes a parameterised policy designed to connect a plug to a power outlet in order for the PR2 to recharge itself. The plug is equipped with a checkerboard to facilitate pose estimation of the plug with respect to power outlet whose position is extracted through a vision processing pipeline. An initial connection is attempted by visual servoing which is successful 10% of the time. When unsuccessful a spiralling outward motion is carried out with 2mm increments. This method achieved an overall success rate of 95%. The hybrid control paradigm Fisher and Mujtaba (1992) was used throughout the execution of the task.

In Yang et al. (2014) the authors learn a PiH policy for the Cranfield benchmark object. A vision system obtains the pose parameters of the object whilst a human teacher demonstrates trajectories, through teleoperation, in the frame of reference of the object. A time-dependent policy represented with Dynamic Movement Primitives (DMP) Schaal et al. (2004) encodes the recorded Cartesian end-effector pose. A F/T profile is encoded separately by a regressor parameterised by radial basis functions. Successive refinements of the DMP policy are achieved through using force feedback to adapt the parameters of an admittance controller. This results in the policy having similar force profiles to the human teachers. Such an approach was first proposed by Nemec et al. (2013) and further applications based on this method have been done Abu-Dakka et al. (2014) with the incorporation of a disturbance rejection policy. Reinforcement learning has also been used in combination with DMP to learn PiH policies. In Kalakrishnan et al. (2011) an DMP policy is initialised with kinesthetic demonstrations of a door opening and pen pick up task. The recorded Cartesian trajectories are encoded in parameterised DMP policy and augmented with a F/T regressor profile. A reward function is designed, encoding desirable properties of the F/T profile such as smoothness and continuity, and after 110 trials a policy was found to be a 100% successful. In Gullapalli et al. (1994) a 18 dimensional input (sensed position, previous position and force) and 6 dimensional output (linear and angular velocity) neural network is learned by associative reinforcement learning. During the learning process the plug is randomly positioned within the vicinity of the hole. After a 100 executions and updates, the policy is successful and was able to generalise across different geometries and clearances.

The above policies were learned from human demonstrations and encoded by a regressor function and optimised to reproduce a desired F/T profile. Further approaches to the PiH problem are predominantly based on heuristic search mechanisms and compliant controllers.

In Chhatpar and Branicky (2001) different blind search policies for the insertion of a spline toothed hub into a forward clutch are analysed. The state space is discretised into points so that the distance between two neighbours is smaller than the clearance of the hole, which is known as a spray point coverage. Different search strategies which ensure that all the points are visited are evaluated. It is found that paths following concentric circles gradually spiralling inwards were the most effective method in finding the hole. The concentric circle search strategy has been applied in many PiH tasks. For instance in M. Bdiwi (2015), a PiH heuristic policy was developed to connect a 5-pin waterproof industrial charger to an electric socket. The authors estimated the pose of the socket through a vision system and used a force controller in combination with a spiral search policy to achieve a connection and demonstrated their approach to be reliable.

In Park et al. (2013) the authors make the observation that humans lack the precision and sensing accuracy of robotic systems. But nevertheless, are more proficient than robots at PiH. The authors observe that when humans try to connect a square plug to a socket, they rub the plug against the socket's outlet without looking. It is thought that the inherent compliance in humans' motor control is the key to our success at PiH tasks kook Yun (2008). The authors introduce an Intuitive Assembly Strategy (IAS) inspired by the above observation which does not require the hole to be precisely localised. The IAS search strategy is based on compliant spiral motion and the execution of the search trajectory is performed with a hybrid force/position controller.

The spiral strategy is widely used in industrial applications due to its simplicity, however, it is a blind search method. Another approach to the assembly process consists of fine-tuning parameters of predefined policies. In Cheng and Chen (2014) the authors develop an online Gaussian Process policy optimisation of an assembly task. They demonstrate that by learning the dynamical model of the task during execution, it can be used to learn the parameters of the policy faster than offline methods, such as Design of Experiment (DOE) or Genetic Algorithms.

4.2 Experiment

The sockets are positioned at the center of a fake wall clamped to a table, see Figure 4.1 (*Top-left*) for an illustration of the environment. Each teacher is given the opportunity to familiarise himself with the environment. Before each trial the human teacher is placed on a chair and disoriented by the experimenter. Once disoriented, the teacher is allowed to stand and is signalled to start the search task by a light touch to the shoulder. Figure 4.1 (*Bottom-left*) illustrates a human teacher performing the task. The disorientation step is to induce the effect of an uniform prior over the teachers believed location. We make the assumption that the human's believed location can be presented by a probability density function, which is assumed to be known. All subsequent distributions can be obtained through the application of a Bayesian filter given that both the sensing and motion information are provided by the force torque and motion capture sensors.

In Figure 4.1 (*Top-right*) we illustrate the experimental setup. The orange area represents the teachers starting location and is assumed prior knowledge. The teachers are told and shown before hand the environmental setup and it is made explicitly clear to them that they will always be starting in the orange area facing the wall.

A group of 10 human teachers participated in the plug-socket experiment. Each teacher performed the search and PiH for sockets A and B. A teacher of group A (starting with socket A), would start by performing 15 times the search and connection with socket A and after a short break, during which socket A was replaced by socket B, would carry on to perform 15 trials with socket B.

Before the actual recording of the task, the teachers had a training period to familiarise themselves with environment and become comfortable in wearing the sensor deprivation apparatus. After each teacher felt sufficiently ready to carry out the task to the best of his ability, the experimenter proceeded to disoriented him through the usage of a chair as mentioned previously. The teachers were reminded that they were facing the direction of the fake wall and that the starting location would be always within the orange rectangular area. In Figure 4.2 we can see the time taken by the teachers to accomplish the task.

Both groups A and B took 9 ± 10s to find their socket. This was expected since the sockets are at the same location. However after the socket was found it took a further 8 ± 7s on average for group A to connect socket B and 12 ± 10s on average for group B to connect socket A. As we can see this is not a straight forward task when considering the sensory deprivation. See Figure 4.2 (*Right*) the time taken to connect the plug to the socket.

The location belief of the humans was represented by a probability density function. We made the assumption that after the disorientation step the human's believed location would be uniform and spread across the rectangular starting area. Although the mental state of the human remains unobservable, there is sufficient evidence to support our assumption that his belief state gets updated in a Bayesian fashion [citation].

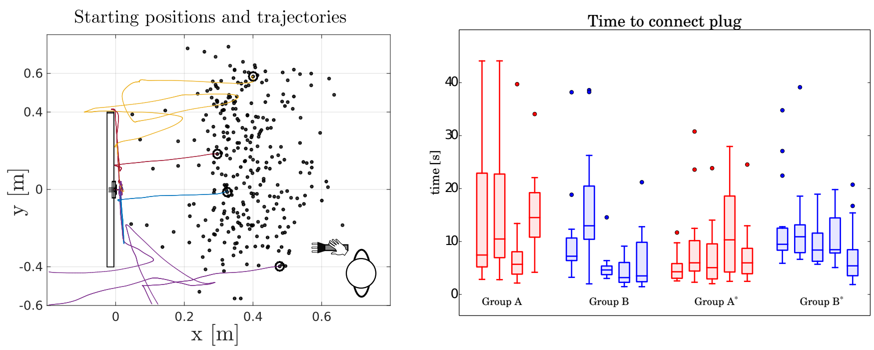


Figure 4.2: *Left*: Black points represent the starting position of the end-effector for all the demonstrations. Four trajectories are illustrated. *Right:* Time taken for the teachers to accomplish the PiH once the socket is localised. Group A and B are depicted in red and blue. The asterisk indicates that the group has changed sockets, so Group A* means that Group A is now performing the task with socket B and Group B* means that group B is now performing the task with socket A.

During each trial we recorded the position and orientation of the plug provided by the motion capture system, and the sensed force and torque, given by the ATI sensor.

4.3 Formulation

4.3.1 Belief probability density function

In our setting the belief probability density function is a Point Mass Filter (PMF) (Bergman and Bergman, 1999, p.87), which is a Bayesian filter. It is parametrised by a set of grid cells which contain valid probabilities. Our choice of a PMF, as means to represent the believed location of the plug, is motivated by the fact that the sensing likelihoods are non-gaussian and lead to multi-modal distributions. A PMF is able to capture such non-gaussianity whilst remaining fully deterministic (which is not the case for a particle filter). The PMF gives a probability density, $p(x_t|y_{0:t}, \dot{x}_{0:t})$, which is recursively updated through the application of a **motion**, $p(x_t|x_{t-1}, \dot{x}_t)$ and **measurement**, $p(y_t|x_t)$ model. The motion model updates the position of the probability density function and subsequently increases the uncertainty of the position. This step essentially consists of applying a convolution kernel to the PMF where the covariance is proportional to the measured velocity. The measurement model indicates areas of the state space from which a measurement $\tilde{y}_t$ could have originated. Both the human teachers and the robot apprentice use the same sensor interface. This is a plug holder equipped with a 6-axis force torque sensor which provides a sensed wrench, $\phi \in \mathbb{R}^6$, which we call the **raw** measurement. We define the **actual** measurement to be a function of the sensed wrench, $\tilde{y}_t = h(\phi_t)$, which

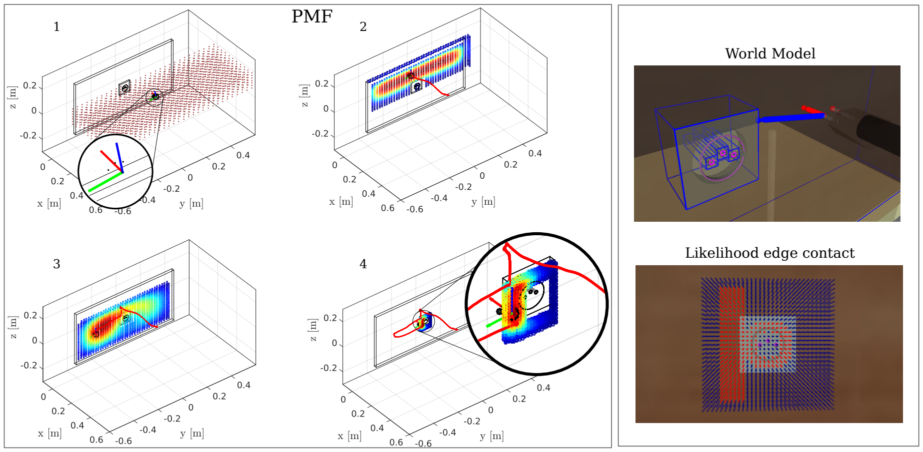


Figure 4.3: *Left:* Point Mass Filter (PMF) update of a particular human demonstration. (1) Initial uniform distribution spread over the starting region. Each grid cell represents a hypothetical position of the plug, the orientation is assumed to be known. (2) First contact, the distribution is spread across the surface of the wall. The red trace is the trajectory history. (3) motion noise increases the uncertainty. (4) The plug is in contact with a socket edge. *Right*: **World model**: The plug is presented by its three plug tips and the wall and sockets are fitted with bounding boxes. **Likelihood**: The plug enters in contact with the left edge of the socket. As a result, the value of the likelihood in all the regions, x_t, close the left edge take a value of one (red points) whilst the others have a value zero (blue points) and areas around the socket's central ring have a value of one.

is a binary feature vector. The feature vector encodes whether a contact is present and the direction in which it occurs, which we discretized to the four cardinalities. In Figure 4.3 (*Right-bottom*) we illustrated the likelihood when an edge is sensed.

4.3.2 Belief compression

The probability density function $p(x_t|y_{0:t}, \dot{x}_{0:t})$ is high dimensional and it is impractical to directly learn a statistical policy $\pi_\theta : p(x_t|y_{0:t}, \dot{x}_{0:t}) \rightarrow \dot{x}_t$; therefore some form of compression is necessary. One possibility would be E-PCA [cite] which finds a set of representative basis functions (which are probability distributions). Although elegant this method requires a discretisation of the belief space which is computationally expensive. Instead we chose to compress the pdf to a belief space vector composed of the maximum a posteriori, $\hat{x}_t^{\text{MAP}} = \text{argmax}_{x_t}\, p(x_t|y_{0:t}, \dot{x}_{0:t}) \in \mathbb{R}^3$, and the differentiation entropy, $U = H\{p(x_t|y_{0:t}, \dot{x}_{0:t})\} \in \mathbb{R}$. All pdfs in our recorded data set D are transformed to a belief space feature vector, $b_t = [\hat{x}_t^{\text{MAP}}, U]^{\text{T}}$.

From the demonstrations we obtained a dataset $D = \{\dot{x}_{1:T}^{[i]}, \omega_{1:T}^{[i]}, \phi_{1:T}^{[i]}, b_{1:T}^{[i]}\}$, where the upper index $[i]$ references the ith trajectory and subscript $1:T$ denotes the time steps during the trajectory from initialisation $t = 1$ until the end $t = T$. The data consisted of the plug's linear velocity, $\dot{x} \in \mathbb{R}^3$, angular velocity $\omega \in \mathbb{R}^3$, the sensed wrench $\phi \in \mathbb{R}^6$ (force-torque), and the belief state,

b, over the plug's location.

4.4 Learning Actor and Critic

In our approach we learn two data driven policies. The first policy maps from belief space to linear velocity $\pi_{\boldsymbol{\theta}_1} : b_t \mapsto \dot{x}_t$ and the second from angular sensed wrench to angular velocity, $\pi_{\boldsymbol{\theta}_2} : \phi_t \mapsto \omega_t$. We chose to learn the belief policy $\pi_{\boldsymbol{\theta}_1}$ in a Actor-Critic RL framework and the wrench policy $\pi_{\boldsymbol{\theta}_2}$ directly from the demonstrated data as was done in [cite], which proved to be efficient in overcoming jamming during the PiH. A POMDP solver's objective is to find a policy (Actor), $\pi_{\boldsymbol{\theta}_1} : b \mapsto u$, which maximises the value function (Critic) $V^{\pi_{\boldsymbol{\theta}_1}} : b \mapsto \mathbb{R}$ for an initial belief, b_0. The value function is the expected reward over an infinite time horizon.

$$V^{\pi_{\boldsymbol{\theta}_1}}(b_t) = \mathbb{E}\left\{ \sum_{t=0}^{\infty} \gamma^t r_{t+1} | b_0 = b, \pi_{\boldsymbol{\theta}_1} \right\} \tag{4.1}$$

In an Actor-Critic setting, the temporal difference error, $\delta_t \in \mathbb{R}$, of the value function (the Critic) is used both as a learning signal to update simultaneously itself and the actor (the policy). In our setting we will learn two separate policies, one for the linear velocity and the other for the angular velocity, as the orientation remains the same during most of the search until it is time to connect the plug to the socket. When the plug has to be connected it is necessary to control for orientation to avoid jamming.

4.4.1 Actor

Both actors/policies are parametrised by a Gaussian Mixture Model (GMM), Equation 4.2.

$$\pi_{\boldsymbol{\theta}}(\dot{x}, b) = \sum_{k=1}^{K} w^{[k]} \cdot g(\dot{x}, b; \boldsymbol{\mu}^{[k]}, \boldsymbol{\Sigma}^{[k]}) \tag{4.2}$$

The parameters $\boldsymbol{\theta} = \{w^{[k]}, \boldsymbol{\mu}^{[k]}, \boldsymbol{\Sigma}^{[k]}\}_{1,\ldots,K}$, are the weights, means and covariances of the individual Gaussian functions, $g(.)$,

$$\boldsymbol{\mu}^{[k]} = \begin{bmatrix} \boldsymbol{\mu}_{\dot{x}}^{[k]} \\ \boldsymbol{\mu}_{b}^{[k]} \end{bmatrix}, \; \boldsymbol{\Sigma}^{[k]} = \begin{bmatrix} \boldsymbol{\Sigma}_{\dot{x}\dot{x}}^{[k]} & \boldsymbol{\Sigma}_{\dot{x}b}^{[k]} \\ \boldsymbol{\Sigma}_{b\dot{x}}^{[k]} & \boldsymbol{\Sigma}_{bb}^{[k]} \end{bmatrix}$$

where $\sum_k w^{[k]} = 1$, $\boldsymbol{\mu}_{\dot{x}}^{[k]} \in \mathbb{R}^3$ and $\boldsymbol{\mu}_{b}^{[k]} \in \mathbb{R}^4$.

A generative model of the angular velocity and wrench $\pi_{\boldsymbol{\theta}_2}(\omega, \theta)$ and a generative model of the linear velocity and belief state $\pi_{\boldsymbol{\theta}_1}(\dot{x}, b)$ are learned. In both cases we use the Bayesian Information Criterion to determine the number of Gaussian functions. In the next section, we will show how the parameters of $\pi_{\boldsymbol{\theta}_1}$ can be adapted by the value function of the Critic.

4.4.2 Critic

The Critic (the value function, Eq. 4.1) evaluates the performance of the current policy. It is the expected future reward given the current belief state and policy. In our setting a reward of $r = 0$ is received at each time step until the goal (plug-socket connection) is achieved, where a reward of 100 is given, $r_T = 100$. Given the continuous nature and dimensionality of the belief space we use locally weighted regression Atkeson et al. (1997) (LWR) as a function approximator of the value function, $V^\pi(b)$. LWR is a memory-based non-parametric function approximator. It keeps a set of input-target pairs $\{(b, r)\}$ as parameters. When a value, b, is queried, a set of p neighbouring points are chosen from the input space and are weighted according to a distance metric. The predicted output is then the result of a weighted least square of the p points. Equation 4.3 is the distance function used where D is a diagonal matrix.

$$W_{i,i} = \exp\left(-\frac{1}{2}(b - b_i)^{\mathrm{T}} D^{-1} (b - b_i)\right) \tag{4.3}$$

A new value is queried according to Equation 4.4,

$$V^\pi(b) = b\,(B^{\mathrm{T}} W B)^{-1} B^{\mathrm{T}} W \mathbf{r} \tag{4.4}$$

where $B = (b_1, \ldots, b_p)^{\mathrm{T}} \in \mathbb{R}^{(D \times p)}$, $W \in \mathbb{R}^{(p \times p)}$ is a diagonal matrix, $\mathbf{r} = (r_1, \cdots, r_p)^{\mathrm{T}} \in \mathbb{R}^{(p \times 1)}$

Fitted Policy Evaluation

To learn the value function we apply batch reinforcement learning Ernst et al. (2005a), also known as experience replay. This is an offline method which applies multiple sweeps of the Bellman backup operator over a dataset of tuples $\{(b_t^{[i]}, \dot{x}_t^{[i]}, r_t^{[i]}, b_{t+1}^{[i]})\}_{i=1,\cdots,M}$ until the Bellman residual, $||V_{k+1}^\pi(b) - V_k^\pi(b)||$, converges.

Algorithm 4.1 Fitted Policy Evaluation

1: **Inputs:**
  K, ϵ, $\{(b_t^{[i]}, r^{[i]}, b_{t+1}^{[i]})\}_{i=1,\cdots,M}$
2: Initialise $\hat{V}_0^\pi = 0$
3: **for** $k = 0$ in K **do**
4: $\hat{V}_{k+1}^\pi(b_t) = \text{Regress}(b,\, r_t + \gamma \hat{V}_k^\pi(b_{t+1}))$
5: **if** $||\hat{V}_{k+1}^\pi(b) - \hat{V}_k^\pi(b)|| < \epsilon$ **then**
6:   **return** $\hat{V}_{k+1}^\pi(b)$
7: **end if**
8: **end for**

A wide spectrum of research has made use of batch RL methods to learn policies. Most of them have focused on learning the Q-value function directly

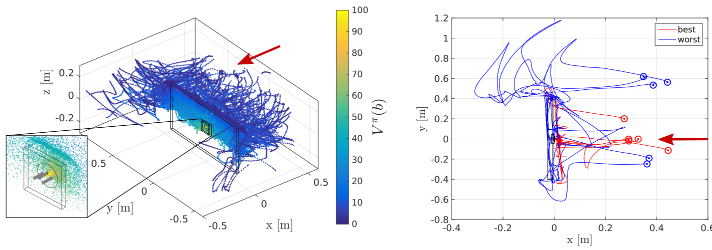


Figure 4.4: *Left:* LWR approximate value function $\hat{V}^\pi(b)$. *Right:* first five best and worst trajectories in terms of the accumulated value.

(Fitted Q-Iteration) Neumann and Peters (2009); Ernst et al. (2005a); Riedmiller (2005a). Although learning the Q-value function directly solves the control problem it often requires discretisation of the action space or assumes quantifiable actions. The reason is that doing Q-Bellman backups, $\hat{Q}(b_t, \dot{x}_t) \leftarrow \gamma \max_{\dot{x}_{t+1}} \hat{Q}(\dot{x}_{t+1}, b_{t+1})$, requires an optimisation over the actions space, $\dot{x}_{t+1}$, to find the best applicable action. Given the dimensionality and continuity of our problem we opted for an on-policy evaluation method which does not require an optimisation, but does require multiple *policy evaluation* and *policy improvements* iterations to achieve an optimal policy. We applied Algorithm 4.1 on our dataset until convergence.

Figure 4.4 (*Left*) shows the belief space with respect to the value function. As expected, the value function is high closest to the socket and low further away. Figure 4.4 (*Right*) shows the best and worst trajectories in terms of the accumulated value function. There is a close relationship between the best trajectories and the time taken to successfully connect the plug to the socket. We can see that the five best trajectories (red) tend to be aligned with the socket (star position in front of socket), whilst the five worst are towards the edges of the wall.

4.4.3 Actor update

The Temporal Difference (TD) error, is used to update the actor $\delta_t^\pi = r_{t+1} + \gamma V^\pi(b_{t+1}) - V^\pi(b_t)$, given by the critic(Sutton and Barto, 1998, Chap. 6). In our offline approach we first computed the value function of the belief state, $V^\pi(b)$, until convergence and then used the estimated value function to update the actor. This offline batch method has the advantage that no divergence will occur during the learning process.

We proceed to update the Actor given the Critic through a modification of the Maximisation step in Expectation-Maximisation (EM) for Gaussian Mixture Models. We refer to this modification as Q-EM which is strongly related to a Monte-Carlo EM-based policy search approach (Deisenroth et al., 2013b, p.50).

The reward of a demonstrated trajectory (one episode) is given by the discounted return, Equation 4.5.

$$R(b^{[i]}, \dot{x}^{[i]}) = \sum_{t=0}^{\mathrm{T}^{[\mathrm{i}]}} \gamma^t \, r(b_t^{[i]}, \dot{x}_t^{[i]}) \tag{4.5}$$

All policy gradient approaches seek to find a set of parameters, $\boldsymbol{\theta}$, of the Actor, which will maximise the expected reward, equivalent to maximising Equation 4.6.

$$\begin{aligned} J(\boldsymbol{\theta}) &= \mathbb{E}_{\pi_{\boldsymbol{\theta}}(\dot{x},b)}\{R(b,\dot{x})\} \\ &= \sum_{i=1}^{N} \underbrace{\left(\prod_{t=0}^{T^{[i]}} \pi_{\boldsymbol{\theta}}(\dot{x}_t^{[i]}, b_t^{[i]}) \right)}_{\pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]})} R(b^{[i]}, \dot{x}^{[i]}) \end{aligned} \tag{4.6}$$

To find the parameters which maximise the cost function, $\mathrm{argmax}_{\boldsymbol{\theta}'} J(\boldsymbol{\theta}')$, the derivative is taken and set to zero. As this cannot be done directly, we maximise the logarithmic lower bound of the cost function. This results in Equation 4.7,

$$\nabla_{\boldsymbol{\theta}'} \log(J(\boldsymbol{\theta}')) = \sum_{i=1}^{N} \sum_{t=0}^{T^{[i]}} \nabla_{\boldsymbol{\theta}'} \log \pi_{\boldsymbol{\theta}'}(\dot{x}_t^{[i]}, b_t^{[i]}) \, Q^{\pi}(\dot{x}_t^{[i]}, b_t^{[i]}) \tag{4.7}$$

Setting the derivative of Equation 4.7 to zero and solving for the parameters $\boldsymbol{\theta} = \{w, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ leads to a Maximisation update step of EM which is weighted by Q^{π}, see Appendix 4.8.1 for a more complete derivation. We use the TD error of the *Critic* as a substitute for Q^{π}. Assuming that our estimated value function, $\hat{V}^{\pi_{\boldsymbol{\theta}}}$, is close to the true value function $V^{\pi_{\boldsymbol{\theta}}}$, the TD error δ^{π} is an unbiased estimate of the advantage function, Equation 4.8 (see Appendix 4.8.3).

$$A^{\pi}(b_t, u_t) = Q^{\pi}(b_t, u_t) - V^{\pi}(b_t) = \delta_t^{\pi} \tag{4.8}$$

Using the advantage function as means of policy search is popular with examples such as Natural Actor Critic (NAC) Peters and Schaal (2008a).

Each data point has an associated weight, $\delta \in \mathbb{R}$, where $\delta^{[m]} \geq 0$ means that the state action-pair $x^{[m]}$ leads to an increase in the value function and $\delta^{[m]} \leq 0$ leads to a decrease in the value function. The likelihood is re-weighted accordingly, giving more importance to data points which lead to a gain. Since the Q-EM update steps cannot allow negative weights, we rescale the TD error to be between 0 and 1.

4.5 Control architecture

We learned both a Gaussian Mixture Model for both the linear and angular velocity. For most of the search and up until the plug is within the socket's hole, only the linear control policy is active. The orientation is kept constant. The direction to search is given by conditional, Equation 4.9,

$$\pi_{\boldsymbol{\theta}}(\dot{x}|b) = \sum_{k=1}^{K} w_{\dot{x}|b}^{[k]} \cdot g(\dot{x}; \boldsymbol{\mu}_{\dot{x}|b}^{[k]}, \boldsymbol{\Sigma}_{\dot{x}|b}^{[k]}) \tag{4.9}$$

which is a distribution over the possible normalised velocities. The subscript $\dot{x}|b$ indicates that the parameters are the result of the conditional. The reader is referred to Calinon et al. (2010),Sung (2004) for a detailed derivation of the conditional of a GMM. In autonomous dynamical systems control, the velocity to is taken from the expectation of Equation 4.9. The model learned is multi-modal, as different search velocities are possible in the same belief state. Taking the expectation, which is weighted linear combination of the modes would result in unobserved behaviour or no movement if the velocities cancel out. In Figure 4.6 we illustrate the multi-modal vector fields of the conditional, Equation 4.9. As a result we use a modified version of the expectation operator which favours the current direction, Equation 4.10 - 4.11.

$$\alpha(\dot{x}) = w_{\dot{x}|b}^{[k]} \cdot \exp(-\cos^{-1}(<\dot{x}, \boldsymbol{\mu}_{\dot{x}|b}^{[k]}>)) \tag{4.10}$$

$$\dot{x} = \mathbb{E}_{\alpha}\{\pi_{\boldsymbol{\theta}}(\dot{x}|b)\} = \sum_{k=1}^{K} \alpha_k(\dot{x}) \cdot \boldsymbol{\mu}_{\dot{x}|b}^{[k]} \tag{4.11}$$

When a velocity mode being applied is no longer present (because we have moved into a region of belief space where the current applied velocity has not been seen) another direction mode is sampled. As an example, when the robot suddenly enters in contact with a feature, which greatly reduces the uncertainty, the current modes will dramatically change and cause a new search direction to be computed.

The above steps can control the general behaviour of the search but are insufficient for a successful implementation on a robotic system, such as the KUKA LWR. This search task is haptic and as a result the end-effector of the robot will always be in contact with the environment. To make the robot compliant with the environment we use an impedance controller in combination with a hybrid position-force controller. Our hybrid controller targets a sensed force F_x, in the x-axis, of 3N. The other two velocity components of the direction vector are given by Equation 4.11.This force by itself in insufficient to reliably surmount the edges of the socket and the robot will become stuck at the edges, unable able to surmount the friction as these right angle contacts. To overcome the edges we locally modulated the vector field of the GMM in y and z-axis, Equation 4.12.

$$\dot{x} = R_y(c(F_z) \cdot \pi/2) \cdot R_z(c(F_y) \cdot \pi/2) \cdot \dot{x} \tag{4.12}$$

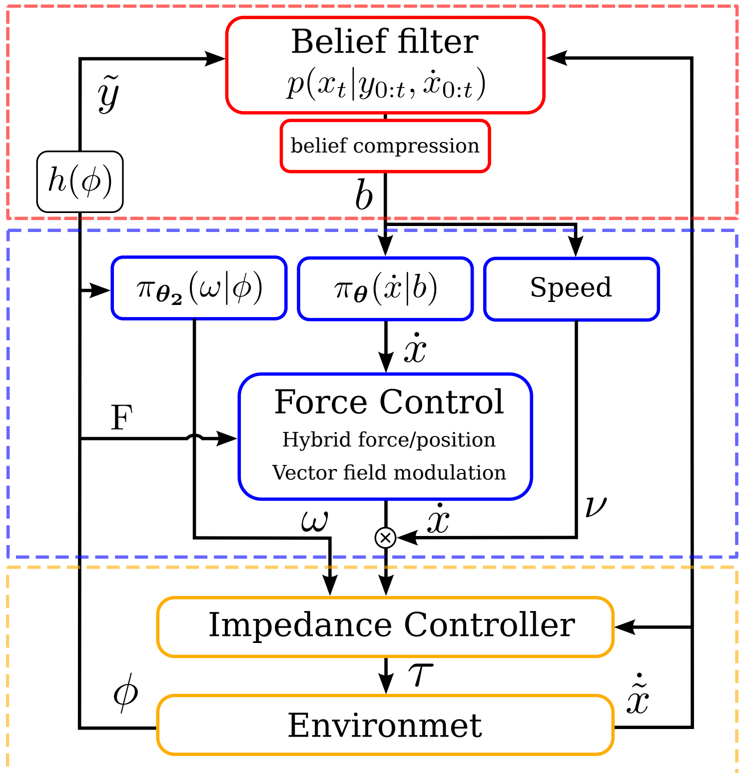


Figure 4.5: Control architecture. The PMF (belief) received a measured velocity, $\dot{\tilde{x}}$, and sensor feature $\tilde{y}$ and gets updated via Bayes rule. The belief is compressed and used by both the GMM policy and the proportional speed controller, Equation 4.13.

R_y and R_z are rotation matrix around the y and z-axis, $c(F) \in [-1, 1]$ is a truncated scaling function of the sensed force. When a force F_z of 5N is sensed, a rotation of $R_y(\pi/2)$ is applied to the original direction resulting in the robot getting over the edge. The direction velocity is always normalised up to this point. The amplitude of the velocity is a proportional controller based on the believed distance to the goal,

$$\nu = \max(\min(\beta_1, K_p(x_g - \hat{x}), \beta_2) \tag{4.13}$$

where the β's are lower and upper amplitude limits, x_g is the position of the goal, and K_p the proportional gain which was tuned through trials. In Figure 4.5 we illustrate the complete control flow.

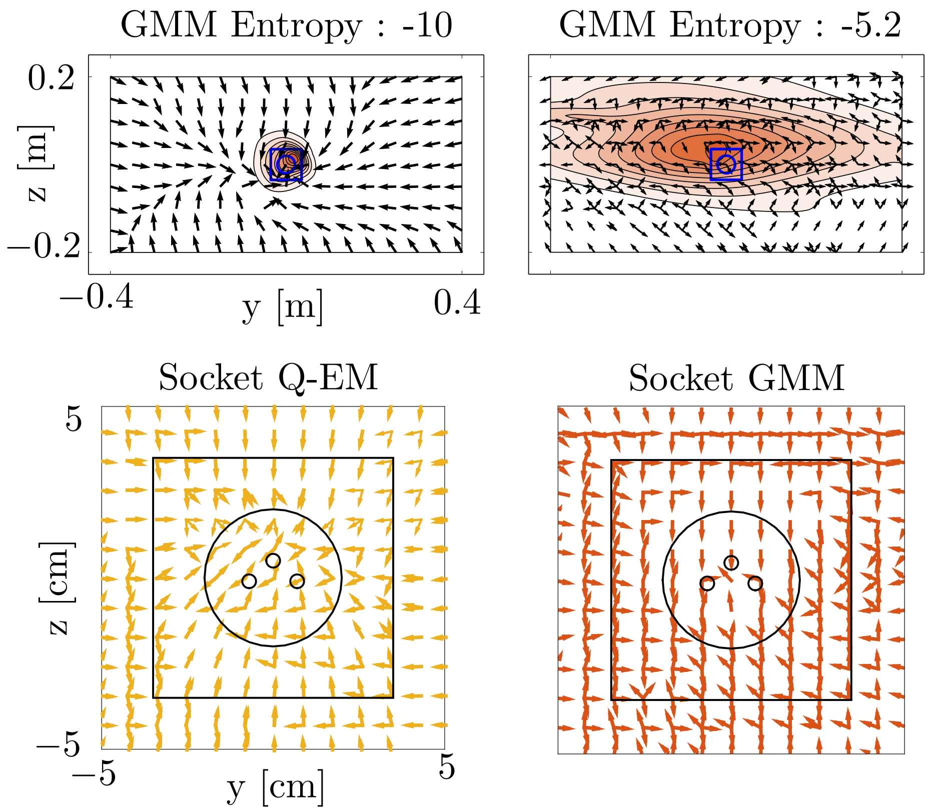


Figure 4.6: Q-EM and GMM policy vector fields. *Top*: The GMM policy is conditioned on an entropy of -10 and -5.2. For the lowest entropy level, most of the probability mass is close to the socket area since this level corresponds to very little uncertainty; we are already localised. We can see that the policy converges to the socket area regardless of the location of the believed state. For an entropy of -5.2 we can see that the likelihood of the policy is present across wall. The vector field directs the end-effector to go towards the left or right edge of the wall. *Bottom*: The entropy is marginalised out, the yellow vector field is of the Q-EM and orange of the GMM. The Q-EM vector field tends to be closer to a sink and there is less variation.

4.6 Results

We evaluate the following three properties of the policy learned in our Actor-Critic framework:

1. **Distance taken to accomplish the goal** (connect plug to socket). We compare the Q-EM policy with a GMM policy learned through standard EM and a myopic Greedy policy. This highlights the difference between complicated and simplistic search algorithms and as a result gives an appreciation of the problem's difficulty.

2. **Importance of data** provided by human teachers. We evaluate whether it is possible to improve the Greedy using it as means of generating demonstrations, which we call Q-Greedy. This is to test whether a human teacher are necessary instead of using heuristics to gather demonstrations. We evaluate whether it is possible to obtain a good policy from the worst two teachers. Not all teachers are necessarily proficient at the task in question and we want to test whether our methodology can be applied in these cases. We evaluate if we are able to obtain an improved policy from the worst two teachers.

3. **Generalisation**. We learn a policy to insert a plug into socket A which was located at the center of the wooden wall. We test the generalisation of the policy in finding a new socket location. We further test whether the policy can generalise to two new sockets which were not used during the training phase.

We evaluate the above properties under two separate conditions. In the **first condition** we consider the period between the start of the search until the socket is localised. In the **second condition** we consider the period from the point the socket is found until a connection has been established. In the first condition the evaluation is done in simulation whilst in the second condition, when the socket is found, we perform the evaluation with a physical robot, the KUKA LWR4.

This choice is motivated by the fact that the two parts of the task require different levels of precision. Finding the socket requires much less precision than establishing a connection. It is thus more informative to consider the performance of these two parts separately. Another aspect is that the search for the socket can be reliably evaluated in simulation since the physics of the interaction is simple. The connection phase is more complicated and a simulation would be unrealistic. For the evaluation of the connection of the plug to the socket we consider the search start point already within the vicinity of the socket.

4.6.1 Distance taken to reach the socket's edge (Qualitative)

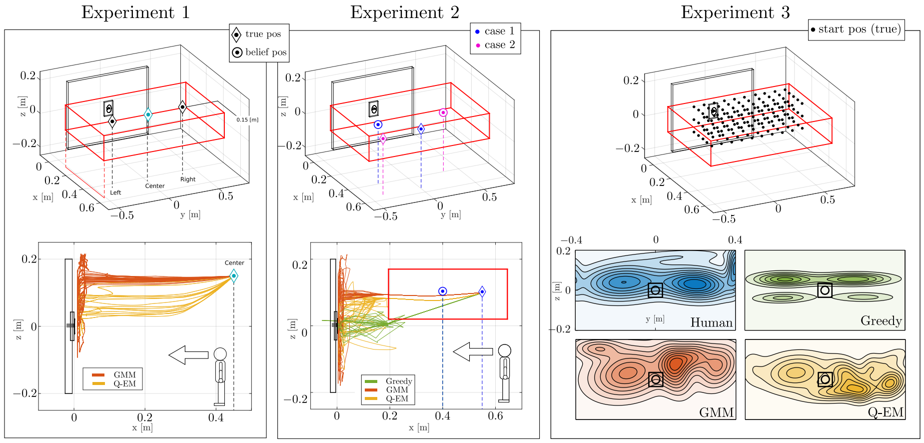


Figure 4.7: Three simulated search experiments. **Experiment 1:** Three start positions are considered: *Left*, *Center* and *Right* in which the triangles depict true position of the end-effector. The red cube illustrates the extent of the uncertainty. In the second row of Experiment 1, we illustrate the trajectories of both the GMM (orange) and Q-EM (yellow) policies. For each start condition a total of 25 searches were performed for each search policy. **Experiment 2:** Two cases are considered: *Case 1* blue, the initial belief state (circle) is fixed facing the left edge of the wall and the true location (diamond) is facing the socket. *Case 2* pink, the initial belief state (circle) is fixed to the right facing the edge of the wall and the true location is the left edge of the wall. In the second row, the trajectories are plotted for *Case 1*. **Experiment 3:** A 150 start locations are deterministically generated from a grid in the start area. In the second row, we plot the distribution of the areas visited by the true position during the search.

We consider three search experiments which we refer to as **Experiment 1, 2** and **3**, in order to evaluate the performance (distance travelled to reach the socket) of three search policies: GMM, Q-EM and Greedy. In these three experiments the task is considered accomplished when a search policy finds the socket's edge.

In **Experiment 1**, three starting locations are chosen: *Center*, *Left* and *Right*, see Figure 4.7, *Experiment 1*, for an illustration of the initial condition. This setup tests the effect of the starting positions. A total of 25 searches are carried out for each of the search policies.

In **Experiment 2**, two *Cases* are chosen in which the believed state (most likely state of the PMF) and the true position of the end-effector are relatively far apart. The location of the beliefs are chosen to be symmetric, see the Figure 4.7, *Experiment 2*. A total of 25 searches are carried for each of the two conditions.

In **Experiment 3**, Figure 4.7, *Experiment 3*, the initial true starting positions of the end-effector are taken from a regular grid covering the whole start region, also used as the initial distribution for the human demonstrations. A total of a 150 searches are carried out for each of the three policies. This experiment compares the search policies with the human teachers.

We evaluate the performance of the three experiments in terms of the trajectories and their distribution in reaching the edge of the socket.

We can see a clear difference between the trajectories generated by the GMM and Q-EM policies in Experiment 1, see Figure 4.7 *Experiment 1, second row.* The orange GMM policy trajectories go straight towards the wall, whilst the yellow Q-EM policy trajectories drop in height making them closer to the socket. The same effect can be seen in Experiment 2 (*second row*). The Q-EM trajectories follow a downward trend towards the location of the socket. The gradient is less due to the initial starting condition being lower than in Experiment 1.

The trajectories of the Greedy policy depend on the chosen believed location (most likely state of the PMF). In the second experiment there is no variance in the Greedy's trajectories until it reaches the edge of the red square, where the branching occurs as the believed location is disqualified. This happens as no sensation is registered when the believed location reaches the wall as the true location is before the believed location, see Figure 4.7, *Experiment 2, second row.*

In Figure 4.7 *Experiment 3, second row*, both Human and GMM distributions of searched locations are similar. They cover the upper region of the wall and top corners, to some extent. These distributions are not identical for two reasons. The first is that the learning of the GMM is a local optimisation which is dependent on initialisation and number of parameters. The second reason is that the synthesis of trajectories from the GMM is a stochastic process.

The distribution of the searched locations of the Q-EM policy is centred around the origin of the z-axis, see Figure 4.7 *Experiment 3, second row.* The uncertainty is predominantly located in the x and y-axis. The Q-EM policy takes this uncertainty into consideration by restraining the search to the y-axis regardless of the starting position. The uncertainty is reduced whilst remaining in the vicinity of the socket. The Greedy's policy search distribution is multi-modal and centred around the z-axis where the modes are above and below the socket. This shows that the Greedy policy acts according to the most likely state which changes from left to right of the socket, because of motion noise, resulting in left-right movements and little displacement. As a result the Greedy policy spends more time at these modes.

In Figure 4.8 (*Top-left*), we illustrate the distribution of the first contact with the wall during Experiment 1 for the *Center* initial conditions. The distribution of the first contact of the Greedy method is uniform across the entire y-axis of the wall. It does not take into account the variance of the uncertainty. In contrast, the GMM policy remains centred with respect to the starting position and the Q-EM is even closer to the socket and there is much less variance in the location of the first contact.

4.6.2 Distance taken to reach the socket's edge (Quantitative)

In Figure 4.9 we illustrate the quantitative results of the distance taken to

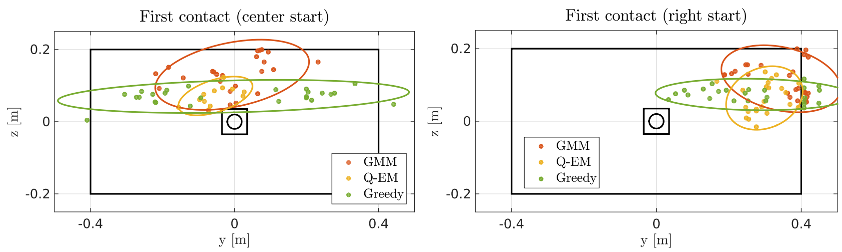


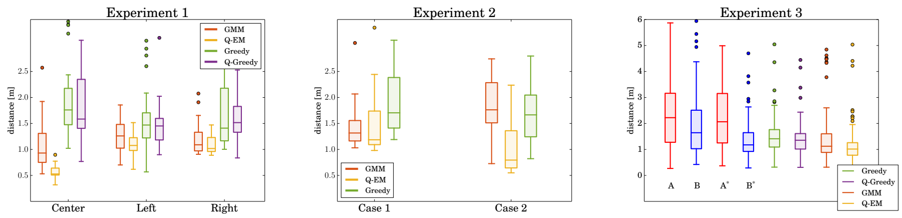
Figure 4.8: First contact with the wall, during experiment 1. (a) Contact distribution for initial condition "Center" . (b) Contact distribution for initial condition was "Right". The ellipses correspond to two standard deviations of a fitted Gaussian function.

reach the socket for all three experiments. In **Experiment 1**, for the *Center* initial condition, the Q-EM policy travels far less than the other search policies. Considering that the initial position of the search is 0.45 [m] away from the wall, the Q-EM policy finds the socket very quickly once contact has been established with the wall. For the *Right* and *Left* starting conditions both the GMM and Q-EM policies travel less distance to reach the socket, with a smaller variance when compared with the Greedy search policy.

In **Experiment 2**, the Q-EM search policy is the most efficient. For *Case 1* of Experiment 2, the initial most likely state is fixed to the left and the true position is facing the socket. As the belief is chosen to be to the left, upon contact with the wall the policy takes a left action since it is more likely to result in a localisation, given that the left edge of the wall is within close proximity. This on average results in an exploration in the upper left area of the wall, which explains why *Case 1* does worse than Experiment 1 for the *Center* initial condition. In *Case 2* however, where the true state is facing the left edge and the believed position is facing the right edge, less distance is taken to find the socket than it does for Case 1, Figure 4.9 (b), as reason for the improvement over Case 1, is that in *Case 2* the true location of the end-effector is close to an edge which is an informative feature and results in a much faster localisation.

From **Experiment 3**, Figure 4.9 (c), it is clear that the three search policies have less variation in the distance travelled to find the socket's edge than the human teachers. All search policies are better than the human teachers with the exception of group B*, which is performing the task with socket A. The Q-EM policy remains the best.

We have shown that under three different experimental settings the Q-EM algorithm is predominantly the best in terms of distance taken to localise the socket. The GMM policy learned solely from the data provided by the human teachers also performs well in comparison to the human teachers and Greedy policy. We made, however a critical assumption in order to be able to use our (RL-)PbD-POMDP approach. This **assumption** is that a human teacher is proficient in accomplishing the task. If a teacher is not able to accomplish the task in a repetitive and consistent way so that a search patter can be encoded



(a) (b) (c)

Figure 4.9: Distance travelled until the socket's edge is reached. (a) Three groups correspond to the initial conditions: Center, Left and Right depicted in Figure 4.7, *top left*. The Q-EM method is always better than the other methods, in terms of distance. (b) Results of the two initial conditions depicted in Figure 4.7, *top middle*, both the true position and most likely state are fixed. The Q-EM method always improves on the GMM. (c) Results corresponding to Experiment 3, Figure 4.7, *top right*. Again the Q-EM method is better, but at a less significant level.

by the GMM, the learned policy will perform poorly. We next evaluate the validity of this assumption and the importance of the training data provided by the human teachers.

4.6.3 Importance of data

We perform two tests to evaluate the importance of the teachers training data for learning a search policy. Firstly we take the worst two teachers in terms of distance taken to find the socket's edge and learn a GMM and Q-EM policy separately from their demonstrations. In this way we can evaluate whether it is possible to learn a successful policy given a few bad demonstrations (15 training trajectories for each policy). Our second evaluation consists of using a noisy explorative Greedy policy as a teacher to gather demonstrations which can then be used to learn a new policy, which we call Q-Greedy.

Figure 4.10 illustrates 6 trajectories of teacher # 5. The human teacher preferred to localise himself at the top of the wall before either proceeding to a corner or going directly towards the socket. Once localised, the teacher would reposition himself in front of the socket and try to achieve an insertion. This behaviour was not expected since by losing contact with the wall, the human teacher no longer has sensory feedback which is necessary to maintain an accurate position estimate.

Figure 4.11 illustrates, the value function of the belief state learned from the data of teacher # 5. The states with the highest values seem to create a path going from the socket towards the right edge of the wall. We proceed as before to learn a GMM policy from the raw data and a Q-EM policy in which the data points are weighted by the gradient of the value function. In Figure 4.12, we illustrate the resulting Marginalised Gaussian Mixture parameters for both the GMM and Q-EM policies and we plot 25 rollouts of each policy starting at the

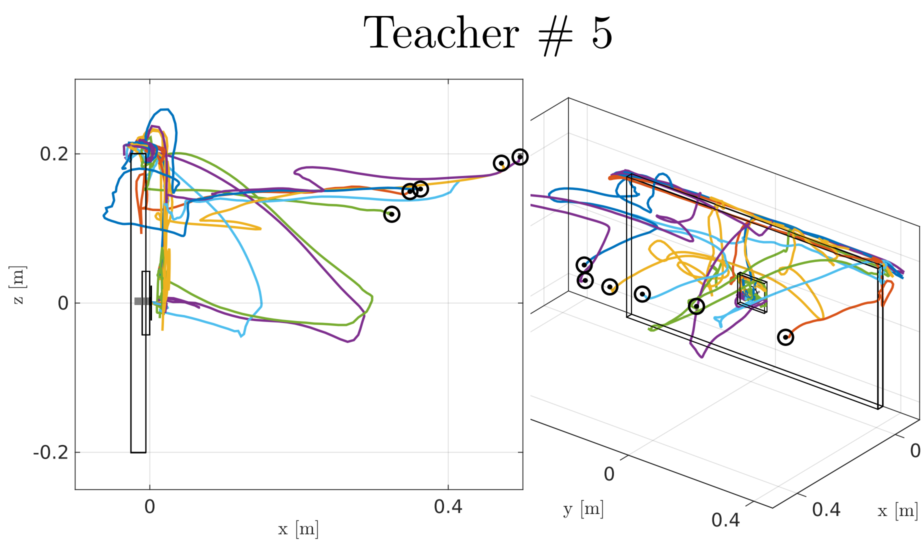


Figure 4.10: Original teacher # 5 demonstrations. The teacher demonstrates a preference to go first to the top of the wall. He then leaves contact with the wall to position himself in front of the socket before trying to find it

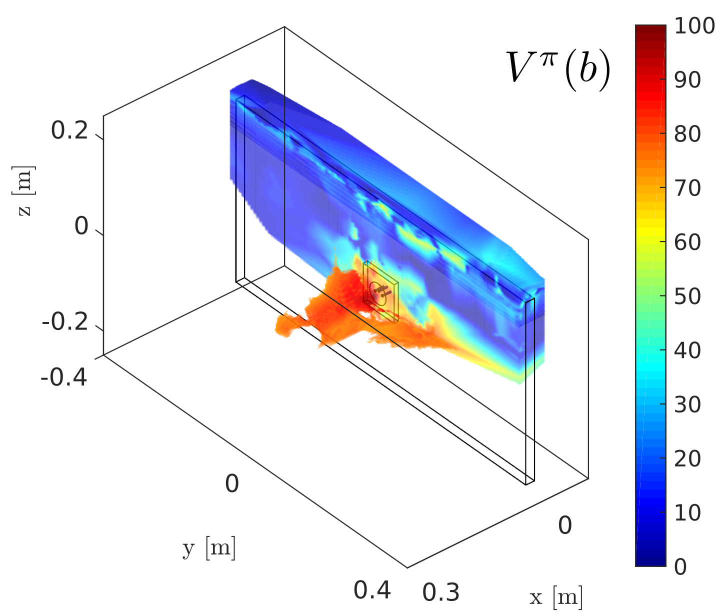


Figure 4.11: Value function learned from the 15 demonstrations of teacher #5.

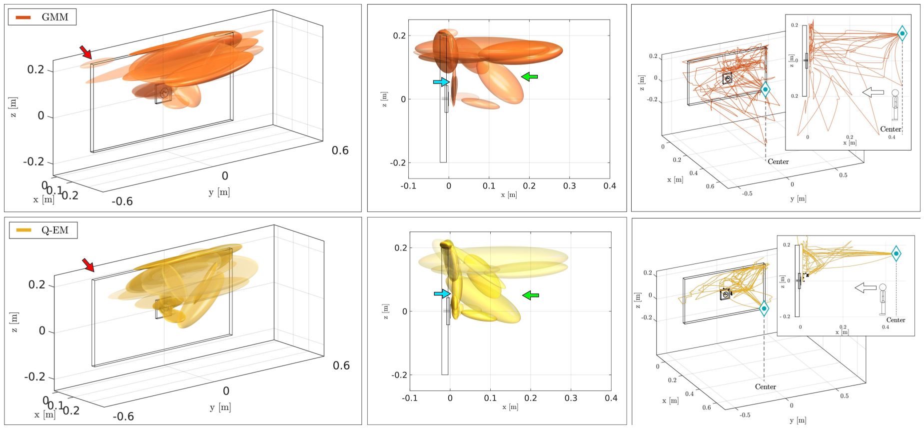


Figure 4.12: Marginalised Gaussian Mixture parameters of the GMM and Q-EM learned from the demonstrations of teacher #5. The illustrated transparency of the Gaussian functions is proportional to their weight. *Left column*: The Gaussian functions of the Q-EM have shifted from the left corner to the right. This is a result of the value function being higher in the top right corner region, see Figure 4.11. *Center column*: The original data of the teacher went quite far back which results in a Gaussian function given a direction which moves away from the wall (green arrow), whilst in the case of the Q-EM parameters this effect is reduced and moved closer towards the wall. We can also see from the two plots of the Q-EM parameters that they then follow the paths encoded by the value function. *Right column*: Rollouts of the policies learned from teacher #5. We can see that trajectories from the GMM policy have not really encoded a specific search patter, whilst the Q-EM policy gives many more consistent trajectories which replicate to some extent the pattern of making a jump (no contact with the wall) from the top right corner to the socket's edge.

Center initial condition used in Experiment 1. We note that the trajectories of the GMM policy seem to have a lot of variance in contrast to the Q-EM policy, resulting from an access of variance amongst the 15 original demonstrations given by the teacher. Furthermore there is insufficient data to encode a pattern for the GMM model. In contrast, the Q-EM finds a pattern by combining multiple parts of the available data and as a result fewer data points are necessary to achieve a good policy. This effect is clear in Figure 4.13, showing the performance of the GMM and Q-EM algorithms under the same initial conditions as in Experiment 1. For all the conditions and for both teachers #5 and #7 the Q-EM policy always does better than the GMM.

We also tested whether we could use the Greedy policy as a means of gathering demonstrations in order to learn a value function and train a Q-Greedy policy. We used the Q-Greedy algorithm in combination with random perturbations applied to the Greedy velocity, to act as a simple exploration technique. We performed a maximum of 150 searches, which terminated once the socket was found and used these demonstrations to learn a value function and GMM policy which we refer to as Q-Greedy. Figure 4.9 illustrates the statistical results of the Q-Greedy policy for Experiment 1 and 3, showing that there is no difference between two policies. Our exploration method is probably too simplistic to discover meaningful search patterns and we could probably devise

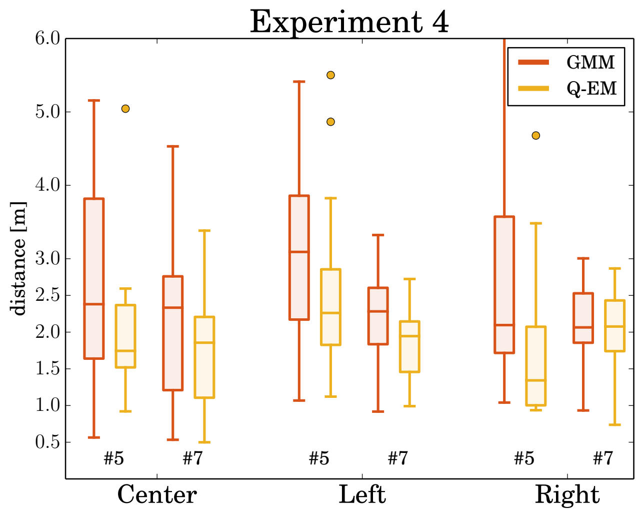


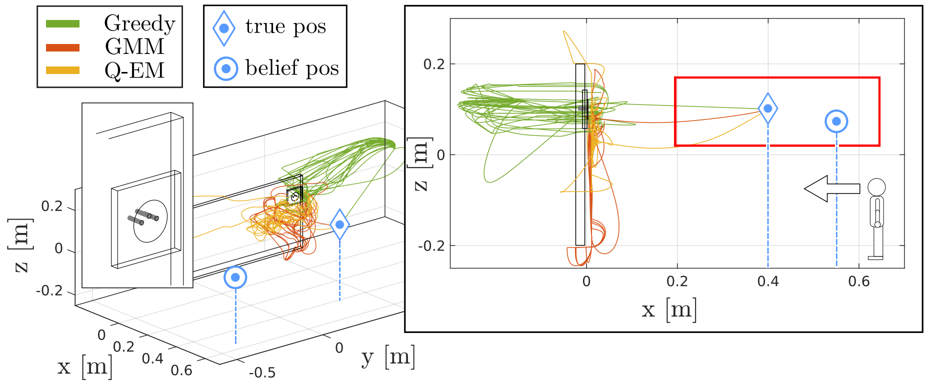
Figure 4.13: Results of a GMM and Q-EM policy under the same test conditions as Experiment 1. The Q-EM policy nearly always does much better than the GMM policy.

better search strategies which would result in a good policy. However we have shown that human behaviour does already have a usable trade-off between exploration and exploitation which can be used to learn a new policy through our RL-PbD-POMDP framework.

4.6.4 Generalisation

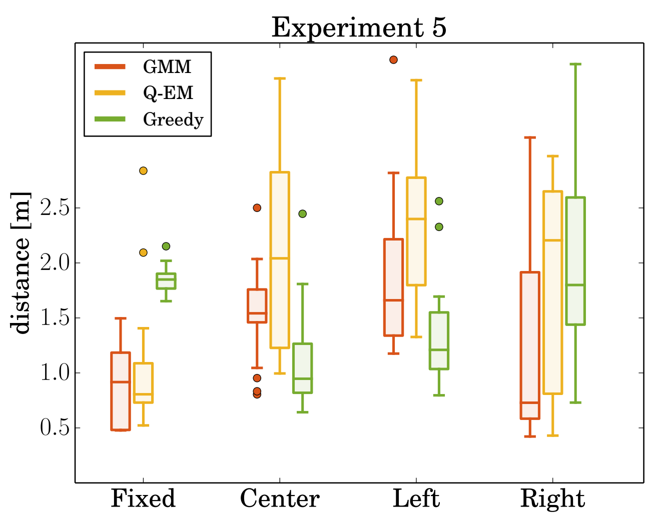
An important aspect of a policy or any machine learning methodology is to be able to generalise. So far we have trained and evaluated our policy within the same environment. To test whether our GMM policies can generalise to a new setting we changed the location of the socket to the upper right corner of the wall. The GMM was trained in the frame of reference of the socket and when we translated the socket's location it also translated the policy.

To evaluate the generalisation of our learned policy we use the same initial conditions of Experiment 1 with an additional new configuration named *Fixed*, in which both the true and believed location are fixed, blue triangle and circle, see Figure 4.14, which illustrates the trajectories of the three search policies for the *Fixed* initial condition. The Greedy policy moves in a straight line towards the top right corner of the table. As the true position is to the right, it takes the Greedy policy longer to find the wall in contrast to both the GMM and Q-EM policies. From the statistical results shown in Figure 4.15 we can see that for the *Fixed* and *Right* initial condition, which are similar, both GMM and Q-EM are better. However, for the *Center* and *Left* initial condition this is no longer the case. The Greedy method is better under this condition since the socket is close to informative features (it is located close to the edges of the wall). Once



(a)

Figure 4.14: Evaluation of generalisation. The socket is located in at the top right corner of the wall. We consider a *Fixed* starting location for both the true and believed location of the end-effector. The red square depicts the extent of the initial uncertainty, which is uniform. (b) Distance taken to reach the socket's edge. For the Fixed setup (see (a) for the initial condition), both the Q-EM and GMM significantly outperform the Greedy. The other three conditions are the same as for Experiment 1.



(a)

Figure 4.15: Distance taken to reach the socket's edge. For the Fixed setup (see Figure 4.14) for the initial condition), both the Q-EM and GMM significantly outperform the Greedy.

the end-effector has entered in contact with the wall the actions of the Greedy policy always result in a decrease of uncertainty, which was not the case when the socket was located in the center of wall. Thus in both the *Fixed* and *Right* initial condition the Greedy method does worse because it takes longer to find the wall.

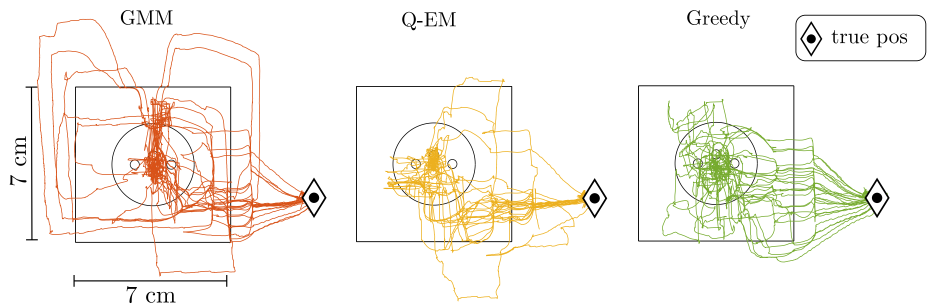
The GMM based policies are still able to generalise under different socket locations. In general, as the socket's location is moved further from the original frame of reference in which it was learned, the more likely will the search quality degrade. We chose the upper right corner since it is the furthest point from the origin and the GMM and Q-EM policies were still able to find the socket. The policy will always be able to find the socket once it has localised itself. This is can be seen from the vector field of the policy, see Figure 4.6, when the entropy is low. In this case the policy acts like a point attractor.

4.6.5 Distance taken to connect the plug to the socket

In this section we evaluate the distance taken for the policies and humans to establish a connection, after the socket has been found. We start measuring the distance from the point that the plug enters in contact with the socket's edge until the plug is connected to the socket. All the following evaluations are done on a KUKA LWR4 robot. The robot's end-effector is equipped with a plug holder on which is attached a force-torque sensor, the same holders used during the demonstration of the human teachers. In this way both the teacher and robot apprentice share the same sensory interface.

We chose to have the robot's end-effector located to the right of the socket and a belief spread uniformly along the z-axis. See Figure 4.17 for an illustration of the initial starting condition. This initial configuration was used to evaluate the search policies for three different sockets, see Figure 4.16 (a) for an illustration of the sockets. We kept the same initial configuration for the evaluation of the three sockets so that we can observe the generalisation properties of the policies. As a reminder we only used the training data from demonstrations acquired during the search with socket A. Socket B has a funnel which should make it easier to connect whilst socket C should be harder as it has no informative features on its surface.

For each of the sockets we performed 25 searches starting from the same initial condition. In Figure 4.16 we plot the trajectories of each of the search methods for socket A. The GMM reproduces some of the behaviour exhibited by humans, such as first localising itself at the top of the socket before trying to attempt to make a connection. The Q-EM algorithm exhibits less variation than the GMM and tends to pass via the bottom of the socket to establish a connection. The Greedy method in contrast is much more stochastic since it does not take into consideration the variance of the uncertainty but instead tries



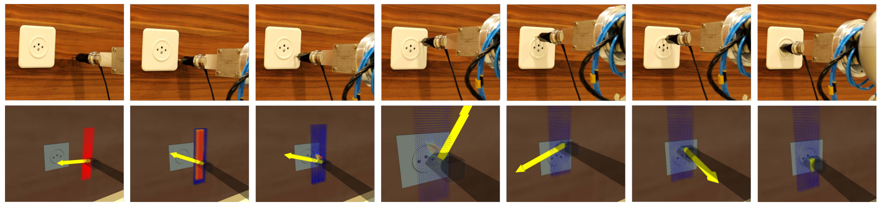
(a)

Figure 4.16: 25 search trajectories for each of the three search policies for socket A.

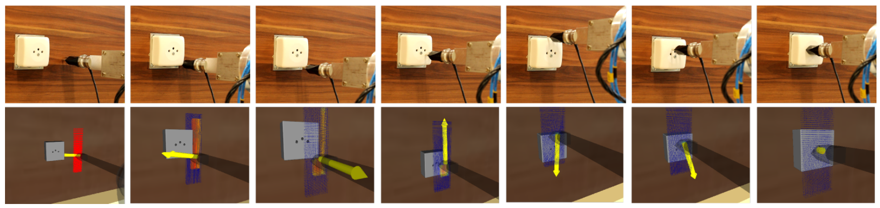
to directly establish a connection. In Figure 4.18 (c) we can see that for socket A both the Greedy and Q-EM are better than the GMM and the Q-EM has less variance in comparison to the Greedy searches. When compared to the human's performance, all three search methods are vastly superior, see Figure 4.18. In Figure 4.17 we illustrate a typical rollout of the GMM search policy for both socket A and C. Once a contact is made with the socket's edge the policy tends to stay close to informative features and tends to wander vertical up and down motions. Only when the uncertainty has been reduced does the GMM policy try to go towards the socket's connector.

The GMM and Q-EM policies are able to generalise to both socket B and C, as the geometric shape and connector interface of the two sockets are similar to socket A. The local force modulation of the policy's vector field, which isn't learned, allows the end-effector to surmount edges and obstacles whilst trying to maintain a constant contact force in the x-axis. This modulation makes it possible for the plug to get on top of socket C. In Figure 4.18 (c) we illustrate the statistics of the distance taken to establish a connection for all three sockets. The point of interest is that both the GMM and Q-EM algorithms do better than the Greedy approach for socket C. Socket C has no informative features on its surface and as a result myopic policies such as in the Greedy case will perform poorly. However for socket A and B, the Greedy policy performs better as both of these sockets have edges around their connector point allowing for easy localisation. It can also be seen that most search methods perform better on socket B than A, since the funnel shape connector helps in maintaining the plug within the vicinity of the socket's holes.

The search discrepancy between the performance of the humans and search policies can be attributed to many causes. One plausible reason is that the PMF probability density representation of the belief is more accurate than the human teachers. The motion noise parameter was fixed to be proportional to the velocity and the robot moves at gentle pace ($\sim$ 1 cm/s) as opposed to some of the human teachers. In actuality, we are far less precise than the KUKA which has sub-millimetre accuracy.

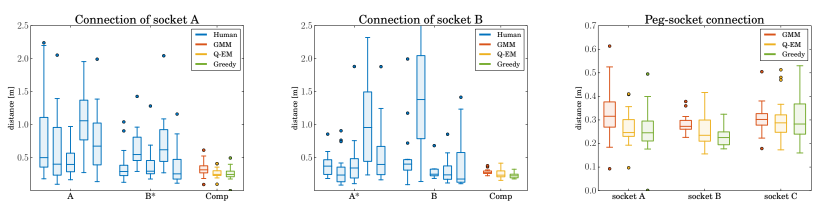


(a)



(b)

Figure 4.17: KUKA LWR4 equipped with a holder mounted with a ATI 6-axis force-torque sensor. (a) The robot's end-effector starts to the right of socket A. The second row are screen captures of the ROS Rviz data visualiser in which we see the Point Mass Filter (red particles) and a yellow arrow indicating the direction given by the policy. In this particular run, the plug remained in contact with the ring of the socket until the top was reached before making a connection. (b) Same initial condition as in (a) but with socket C. The policy leads the plug down to the bottom corner of the socket before going the center of the top edge, localising itself, and then makes a connection.



(a) **(b)** **(c)**

Figure 4.18: Distance taken to connect plug to socket once the socket is localised. (a) **Socket A**. The human Group A are the set of teachers who first started with socket A. They had no previous training on another socket beforehand. Group B* first gave demonstrations on Socket B before giving demonstrations on Socket A. Group B* is better than group A at doing the task. This is most likely a training effect. However all policy search methods are far better at connecting the plug to the socket. (b) **Socket B**. Both groups A* and B are similar in terms of the distance they took to insert the plug into the socket and as was the case for (a), the search policies travel less to accomplish the task. (c) Distance taken (measured from point of contact of plug with socket edge) to connect the plug to the socket.

4.7 Discussion & Conclusion

In this work we learned search policies from demonstrations provided by human teachers for a task which consisted of first localising a power socket (either socket A, B or C) and then connecting it with a plug. Only haptic information was available as the teachers were blindfolded. We made the assumption that the position belief of the human teachers was initially uniformly distributed in a fixed rectangular region of which they were informed and is considered prior knowledge. All subsequent beliefs were then updated in a Bayesian recursion using the measured velocity obtained from a vision tracking system, and wrench acquired from a force torque sensor attached to the plug. The filtered probability density function, represented by a Point Mass Filter, was then compressed to the most likely state and entropy.

Two Gaussian Mixture Model policies where learned from the data recordedj during the teaching by the human teachers . The first policy, called Q-EM, was learned in an Actor-Critic RL framework in which a value function was learned over the belief space. This was then used to weigh training datapoints in the M-step update of Expectation-Maximisation (EM). The second policy, called GMM, was learned using the standard EM algorithm, considering all training data points equally, following in the footsteps of our initial approach Chambrier and Billard (2014). Both the Q-EM and GMM policies were trained with data solely from the demonstrations of the search with socket A.

We evaluated 4 different aspects of the learned policies. Firstly, we evaluated which of three policies, Q-EM, GMM and a Greedy policy, took the least distance to find the socket. We concluded that across three different Experiments the Q-EM algorithm was always the best. It was clear that the Q-EM policy was less random and more consistent than the GMM policy as it tried to enter in contact with the wall at the same height as the socket thus increasing the chances of finding the socket.

Secondly, we tested the importance of the data provided by the human teachers. We took the worst two teachers and trained an individual GMM and Q-EM policy for each of them. We found that the performance of the Q-EM was better than the GMM in terms of distance travelled to find the socket. When qualitatively evaluating the trajectories of the GMM with respect to the Q-EM for the worst teacher, it is clear that the Q-EM policy managed to extract a search pattern, which was not the case for the GMM policy. We also tried to learn a Q-EM policy from the data provided by a Greedy policy with explorative noise and we found no improvement. From these results we conclude that the exploration and exploitation aspects of the trajectories provided by the human teachers is necessary.

Thirdly we tested whether the two policies were able to generalise to a different socket location. Under a specific condition, which we called *Fixed*, both policies were significantly better than the Greedy policy. However for the *Center* and *Left* initial conditions the Greedy policy was better. For the initial conditions in which the Greedy policy enters in contact with the wall at an early stage, it performs better than the GMM and Q-EM. The reason for this is that the actions taken by the Greedy policy in this setting will always result in a decrease of entropy when the location of the socket is close to a corner, as opposed to being in the center of the wall.

Fourthly we evaluated the three policies on the KUKA LWR4 robot. First all the policies did better than the human teachers. For socket A, on which both the GMM and Q-EM policies were trained, there is no clear distinction between the Q-EM and Greedy policy. On socket B, which was novel, the Greedy policy performed better than the statistical controllers, which we hypothesize was a result of a funnel which would make it easier for a myopic policy. For socket C, both the GMM and Q-EM policies do better than the Greedy, as socket C has no features on its surface, this being a disadvantage for a myopic policy.

We concluded by making the observation that by simply adding a binary reward function in combination with data provided by human demonstrations, with Fitted reinforcement learning, we can learn a better policy without the need of doing expensive exploration-exploitation rollouts traditionally associated with reinforcement learning and designing complicated reward functions. This is especially advantageous when only a few demonstrations are available.

4.8 Appendix

4.8.1 EM policy search

Steps taken to make a policy $pi_{\boldsymbol{\theta}}(\dot{x}, b)$ maximise the objective function, $J(\boldsymbol{\theta})$. The policy will be maximised with respect to the lower bound of the cost function $J(\boldsymbol{\theta})$:

$$
\begin{aligned}
J(\boldsymbol{\theta}') &= \sum_{i \in \mathbb{T}} \pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})\, R(\dot{x}^{[i]}, b^{[i]}) \\
&= \sum_{i \in \mathbb{T}} \frac{\pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})}{\pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]})}\, \pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]})\, R(\dot{x}^{[i]}, b^{[i]}) \qquad (4.14)
\end{aligned}
$$

where $\mathbb{T}$ is the set of all rollouts. Next we take the logarithm and make use of Jensen's inequality and move the logarithm into the summation.

$$
\begin{aligned}
\log(J(\boldsymbol{\theta}')) &= \log \sum_{i \in \mathbb{T}} \frac{\pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})}{\pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]})} \, \pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]}) \, R(\dot{x}^{[i]}, b^{[i]}) \\
&\geq \sum_{i \in \mathbb{T}} \log \left(\frac{\pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})}{\pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]})} \right) \pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]}) \, R(\dot{x}^{[i]}, b^{[i]}) \qquad (4.15)
\end{aligned}
$$

We take the derivative of the lower bound of $\log(J(\boldsymbol{\theta}'))$, Equation 4.15, with respect to $\boldsymbol{\theta}'$ and set it to zero so as to maximise the cost function.

$$
\begin{aligned}
&\nabla_{\boldsymbol{\theta}'} \log(J(\boldsymbol{\theta}')) = \\
&\sum_{i \in \mathbb{T}} \nabla_{\boldsymbol{\theta}'} \log \left(\pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})\right) \pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]}) \, R(\dot{x}^{[i]}, b^{[i]}) \\
&- \underbrace{\nabla_{\boldsymbol{\theta}'} \log \left(\pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]})\right) \pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]}) \, R(\dot{x}^{[i]}, b^{[i]})}_{=0} \\
&= \sum_{i \in \mathbb{T}} \nabla_{\boldsymbol{\theta}'} \log \left(\pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})\right) \pi_{\boldsymbol{\theta}}(\dot{x}^{[i]}, b^{[i]}) \, R(\dot{x}^{[i]}, b^{[i]}) \\
&= \mathbb{E}_{\pi_{\boldsymbol{\theta}}(\dot{x}, b)} \Big\{ \nabla_{\boldsymbol{\theta}'} \log \left(\pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]})\right) R(\dot{x}^{[i]}, b^{[i]}) \Big\} \qquad (4.16)
\end{aligned}
$$

$$
\begin{aligned}
\nabla_{\boldsymbol{\theta}'} \log(J(\boldsymbol{\theta}')) &= \mathbb{E}_{\pi_{\boldsymbol{\theta}}(\dot{x}, b)} \Big\{ R(b^{[i]}, \dot{x}^{[i]}) \sum_{t=0}^{T} \nabla_{\boldsymbol{\theta}'} \log \pi_{\boldsymbol{\theta}'}(\dot{x}^{[i]}, b^{[i]}) \Big\} \qquad (4.17) \\
&= \sum_{i=1}^{N} \sum_{t=0}^{T^{[i]}} \nabla_{\boldsymbol{\theta}'} \log \pi_{\boldsymbol{\theta}'}(\dot{x}_t^{[i]}, b_t^{[i]}) \, Q^{\pi}(\dot{x}_t^{[i]}, b_t^{[i]}) \qquad (4.18)
\end{aligned}
$$

The reader is referred to Deisenroth et al. (2013a) for more details regarding Expectation-Maximisation and policy search in reinforcement learning.

4.8.2 Q-EM for GMM derivation

Making the substitution $x = (\dot{x}, b)^{\mathrm{T}}$ (small abuse of the notation) and insuring a positive Q-function, $Q^{\pi}(x^{[m]}) \geq 0$ and by setting the derivative of Equation 4.18 to zero and solving for the parameters $\boldsymbol{\theta} = \{w, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ we get a new weighted Maximisation update step in EM:

$$
\begin{aligned}
\nabla_{\boldsymbol{\mu}^{[k]}} \log J(\boldsymbol{\theta}) &= \sum_{m=1}^{M} \alpha(z_{mk})\, Q(x^{[m]})\, {\boldsymbol{\Sigma}^{[k]}}^{-1} (x^{[m]} - \boldsymbol{\mu}^{[k]}) = 0 \\
\boldsymbol{\mu}_{\mathbf{new}}^{[k]} &= \frac{\sum_{m=1}^{M} \alpha(z_{mk})\, Q(x^{[m]})\, x^{[m]}}{\sum_{j=1}^{M} \alpha(z_{jk})\, Q(x^{[j]})}
\end{aligned} \tag{4.19}
$$

where $\alpha(z_{mk})$ is the responsibility factor, denoting the probability that data point m is a member of the Gaussian function k.

$$
\alpha(z_{mk}) = \frac{w^{[k]} \cdot g(x^{[m]}; \boldsymbol{\mu}^{[k]}, \boldsymbol{\Sigma}^{[k]})}{\sum_{j=1}^{K} w^{[j]} \cdot g(x^{[m]}; \boldsymbol{\mu}^{[j]}, \boldsymbol{\Sigma}^{[j]})} \tag{4.20}
$$

$$
\boldsymbol{\Sigma}_{\mathbf{new}}^{[k]} = \frac{\sum_{m=1}^{M} Q(x^{[m]}) \alpha(z_{mk}) (x^{[m]} - \boldsymbol{\mu}^{[k]})(x^{[m]} - \boldsymbol{\mu}^{[k]})^{\mathrm{T}}}{\sum_{j=1}^{M} Q(x^{[j]})\, \alpha(z_{jk})} \tag{4.21}
$$

$$
w_{\mathbf{new}}^{[k]} = \frac{\sum_{m=1}^{M} Q(x^{[m]})\, \alpha(mk)}{\sum_{j=1}^{M} Q(x^{[j]})} \tag{4.22}
$$

4.8.3 Unbiased estimator

The temporal difference error is an unbiased estimate of the advantage function:

$$
\begin{aligned}
\mathop{\mathbb{E}}_{\pi_{\boldsymbol{\theta}}}\{\delta_t^{\pi} | b_t, u_t\} &= \mathop{\mathbb{E}}_{\pi_{\boldsymbol{\theta}}}\{r_{t+1} + \gamma V^{\pi}(b_{t+1}) | b_t, u_t\} - V^{\pi}(b_t) \\
&= Q^{\pi}(b_t, u_t) - V^{\pi}(b_t) \\
&= A^{\pi}(b_t, u_t)
\end{aligned} \tag{4.23}
$$

References

A. a. Agha-mohammadi, S. Chakravorty, and N. M. Amato. Firm: Feedback controller-based information-state roadmap - a framework for motion planning under uncertainty. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4284–4291, Sept 2011. doi: 10.1109/IROS.2011.6095010. 2.3.3

A. a. Agha-mohammadi, S. Agarwal, A. Mahadevan, S. Chakravorty, D. Tomkins, J. Denny, and N. M. Amato. Robust online belief space planning in changing environments: Application to physical mobile robots. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 149–156, May 2014. doi: 10.1109/ICRA.2014.6906602. 2.3.3

Douglas Aberdeen and Jonathan Baxter. Scaling internal-state policy-gradient methods for pomdps. In Claude Sammut and Achim Hoffman, editors, *Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002)*, pages 3–10, San Francisco, CA, USA, 2002. Morgan Kauffman. ISBN 1-55860-873-7. URL `http://users.rsise.anu.edu.au/~daa/files/papers/gradIstate-icml.pdf`. 2.3.2

Fares Abu-Dakka, Bojan Nemec, Aljaz Kramberger, Anders Glent Buch, Norbert Krüger, and Ales Ude. Solving peg-in-hole tasks by human demonstration and exception strategies. *Industrial Robot*, 41(6):575–584, 2014. ISSN 0143-991X. doi: http://dx.doi.org/10.1108/IR-07-2014-0363. 4.1.1

M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, February 2002. 3.4

Christopher G. Atkeson, Andrew W. Moore, and Stefan Schaal. Locally weighted learning. *ARTIFICIAL INTELLIGENCE REVIEW*, pages 11–73, 1997. 4.4.2

Chris Bake, Joshua. Tenenbaum, and Rebecca Saxe. Bayesian theory of mind: Modeling joint belief-desire attribution. *Journal of Cognitive Science*, 2011. 1.2.1, 3.2.2, 3.4, 3.7

Chris L. Baker, Joshua B. Tenenbaum, and Rebecca R. Saxe. Bayesian models of human action understanding. In *Advances in Neural Information Processing Systems 18*, pages 99–106, 2006. 3.2.2

Simon Baron-Cohen. *Mindblindness*. MIT Press, 1995. 3.2.2

Jonathan Baxter and Peter L. Bartlett. Reinforcement learning in pomdp's via direct gradient ascent. In *In Proc. 17th International Conf. on Machine Learning*, pages 41–48. Morgan Kaufmann, 2000. 2.3.2, 2.3.2

Niclas Bergman and C Niclas Bergman. Recursive bayesian estimation: Navigation and tracking applications. thesis no 579. Technical report, Linköping University, Linköping Studies in Science and Technology. Doctoral dissertation, 1999. 4.3.1

D. Bernoulli. Exposition of a New Theory on the Measurement of Risk (1748). *Econometrica*, 22(1):23–36, 1954. 2.1.1

A. Billard and D. Grollman. Robot learning by demonstration. 8(12):3824, 2013. 2.4

A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Robot programming by demonstration. In B. Siciliano and O. Khatib, editors, *Handbook of Robotics*, pages 1371–1394. Springer, Secaucus, NJ, USA, 2008a. 1.1, 2.4

Aude Billard, Sylvain Calinon, Rüdiger Dillmann, and Stefan Schaal. Robot programming by demonstration. In Oussama Khatib Prof. Bruno Siciliano Prof., editor, *Springer Handbook of Robotics*, pages 1371–1394. Springer, Springer, 2008b. 3.5.1

Sebastian Brechtel, Tobias Gindele, and Rüdiger Dillmann. Solving continuous pomdps: Value iteration with incremental learning of an efficient space representation. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 370–378. JMLR Workshop and Conference Proceedings, May 2013. URL http://jmlr.org/proceedings/papers/v28/brechtel13.pdf. 2.3.1

Alex Brooks and Stefan Williams. A monte carlo update for parametric pomdps. In Makoto Kaneko and Yoshihiko Nakamura, editors, *Robotics Research*, volume 66 of *Springer Tracts in Advanced Robotics*, pages 213–223. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-14742-5. doi: 10.1007/978-3-642-14743-2_19. URL http://dx.doi.org/10.1007/978-3-642-14743-2_19. 2.3.1

Neil Burgess. Spatial memory: how egocentric and allocentric combine. *Trends in Cognitive Sciences*, 10(12):551 – 557, 2006. ISSN 1364-6613. doi: http://dx.doi.org/10.1016/j.tics.2006.10.005. URL http://www.sciencedirect.com/science/article/pii/S1364661306002713. 3.2.1

J. Butterfield, O. C. Jenkins, D. Sobel, and J. Schwertfeger. Modeling aspects of Theory of Mind with Markov Random Fields. *International Journal of Social Robotics*, 1(1):41–51, January 2009. 3.2.2

S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. G. Billard. Learning and reproduction of gestures by imitation. *IEEE Robotics Automation Magazine*, 17(2):44–54, June 2010. ISSN 1070-9932. 4.5

A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien. Acting under uncertainty: discrete bayesian models for mobile-robot navigation. In *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, volume 2, pages 963–972 vol.2, Nov 1996a. 1.1

A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien. Acting under uncertainty: discrete bayesian models for mobile-robot navigation. In *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, volume 2, pages 963–972 vol.2, Nov 1996b. 2.3.4

Guillaume de Chambrier and Aude Billard. Learning search policies from humans in a partially observable context. *Robotics and Biomimetics*, 1 (1):1–16, 2014. ISSN 2197-3768. doi: 10.1186/s40638-014-0008-1. URL http://dx.doi.org/10.1186/s40638-014-0008-1. 4.7

D. Chen and G. von Wichert. An uncertainty-aware precision grasping process for objects with unknown dimensions. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 4312–4317, May 2015. doi: 10.1109/ICRA.2015.7139794. 2.3.4

H. Cheng and H. Chen. Online parameter optimization in robotic force controlled assembly processes. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3465–3470, May 2014. doi: 10.1109/ICRA.2014.6907358. 4.1.1

S. R. Chhatpar and M. S. Branicky. Search strategies for peg-in-hole assemblies with position uncertainty. In *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, volume 3, pages 1465–1470 vol.3, 2001. doi: 10.1109/IROS.2001.977187. 4.1.1

Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2011. ISSN 1935-8253. doi: 10.1561/2300000021. URL http://dx.doi.org/10.1561/2300000021. 2.3.2

Marc Peter Deisenroth, Gerhard Neumann, and Jan Peters. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013a. ISSN 1935-8253. doi: 10.1561/2300000021. URL http://dx.doi.org/10.1561/2300000021. 4.8.1

MP. Deisenroth, G. Neumann, and J. Peters. A survey on policy search for robotics, foundations and trends in robotics. *Foundations and Trends in Robotics*, 2(1-2):1–142, 2013b. 4.4.3

Sandra Devin and Rachid Alami. An implemented theory of mind to improve human-robot shared plans execution. In *The Eleventh ACM/IEEE International Conference on Human Robot Interation, HRI 2016, Christchurch, New Zealand, March 7-10, 2016*, pages 319–326, 2016. doi: 10.1109/HRI.2016.7451768. URL http://dx.doi.org/10.1109/HRI.2016.7451768. 3.2.2

Y.Z. Du, D. Hsu, H. Kurniawati, W.S. Lee, S.C.W. Ong, and S.W. Png. A pomdp approach to robot motion planning under uncertainty. In *Int. Conf. on Automated Planning and Scheduling, Workshop on Solving Real-World POMDP Problems*, 2010. URL papers/icaps10_pomdpApsInRobotics.pdf. 2.2.1, 2.3.1

Tom Erez and William D. Smart. A scalable method for solving high-dimensional continuous pomdps using local approximation. In *Conf. on Uncertainty in Artificial Intelligence*, 2010. 2.3.3

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, April 2005a. 4.4.2, 4.4.2

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *J. Mach. Learn. Res.*, 6:503–556, December 2005b. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1046920.1088690. 2.3.1

William D. Fisher and M. Shahid Mujtaba. Hybrid position/force control: A correct formulation. *The International Journal of Robotics Research*, 11(4):299–311, 1992. doi: 10.1177/027836499201100403. URL http://ijr.sagepub.com/content/11/4/299.abstract. 4.1.1

I. Grondman, L. Busoniu, G. A. D. Lopes, and R. Babuska. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307, Nov 2012. ISSN 1094-6977. doi: 10.1109/TSMCC.2012.2218595. 2.3.2

Vijaykumar Gullapalli, Andrew G. Barto, and Roderic A. Grupen. Learning admittance mappings for force-guided assembly. In *Proceedings of the 1994 International Conference on Robotics and Automation, San Diego, CA, USA, May 1994*, pages 2633–2638, 1994. doi: 10.1109/ROBOT.1994.351117. URL http://dx.doi.org/10.1109/ROBOT.1994.351117. 4.1.1

Bradley Hamner, Sanjiv Singh, and Sebastian Scherer. Learning obstacle avoidance parameters from operator behavior. *Field Robotics*, 23(11/12):1037–1058, December 2006. 3.2.3

Kris Hauser. *Algorithmic Foundations of Robotics IX: Selected Contributions of the Ninth International Workshop on the Algorithmic Foundations of Robotics*, chapter Randomized Belief-Space Replanning in Partially-Observable Continuous Spaces, pages 193–209. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-17452-0. doi: 10.1007/978-3-642-17452-0_12. URL http://dx.doi.org/10.1007/978-3-642-17452-0_12. 2.3.4

Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. 2015. URL https://www.aaai.org/ocs/index.php/FSS/FSS15/paper/view/11673. 2.3.1

Ruijie He, S. Prentice, and N. Roy. Planning in information space for a quadrotor helicopter in a gps-denied environment. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1814–1820, May 2008. doi: 10.1109/ROBOT.2008.4543471. 2.3.3

P. Hebert, T. Howard, N. Hudson, J. Ma, and J.W. Burdick. The next best touch for model-based localization. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 99–106, May 2013. doi: 10.1109/ICRA.2013.6630562. 2.3.4

G. A. Hollinger, B. Englot, F. Hover, U. Mitra, and G. S. Sukhatme. Uncertainty-driven view planning for underwater inspection. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4884–4891, May 2012. doi: 10.1109/ICRA.2012.6224726. 2.3.4

K. Hsiao, L. Kaelbling, and T. Lozano-Perez. Task-driven tactile exploration. In *Proceedings of Robotics: Science and Systems*, Zaragoza, Spain, June 2010. doi: 10.15607/RSS.2010.VI.029. 2.3.4

M.F. Huber, Tim Bailey, H. Durrant-Whyte, and U.D. Hanebeck. On entropy approximation for gaussian mixture random vectors. In *Multisensor Fusion and Integration*, pages 181–188, 2008. 3.4

Tina Iachini, Gennaro Ruggiero, and Francesco Ruotolo. Does blindness affect egocentric and allocentric frames of reference in small and large scale spaces? *Behavioural Brain Research*, 273(0):73 – 81, 2014. ISSN 0166-4328. doi: http://dx.doi.org/10.1016/j.bbr.2014.07.032. URL http://www.sciencedirect.com/science/article/pii/S0166432814004811. 3.2.1

Shervin Javdani, Matthew Klingensmith, Drew Bagnell, Nancy S. Pollard, and Siddhartha S. Srinivasa. Efficient touch based localization through submodularity. *CoRR*, abs/1208.6067, 2012. URL http://arxiv.org/abs/1208.6067. 2.3.4

Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Tingfan Wu, Daniel Duran, Marshall Floyd, Peter Abeles, Douglas Stephen, Nathan Mertins, Alex Lesman, John Carff, William Rifenburgh, Pushyami Kaveti, Wessel Straatman, Jesper Smith, Maarten Griffioen, Brooke Layton, Tomas de Boer, Twan Koolen, Peter Neuhaus, and Jerry Pratt. Team ihmc's lessons learned from the darpa robotics challenge trials. *Journal of Field Robotics*, 32(2): 192–208, 2015. ISSN 1556-4967. doi: 10.1002/rob.21571. URL http://dx.doi.org/10.1002/rob.21571. 1.1

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artif. Intell.*, 101(1-2):99–134, May 1998. ISSN 0004-3702. doi: 10.1016/S0004-3702(98)00023-X. URL http://dx.doi.org/10.1016/S0004-3702(98)00023-X. 2.2.1

M. Kalakrishnan, L. Righetti, P. Pastor, and S. Schaal. Learning force control policies for compliant manipulation. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4639–4644, Sept 2011. doi: 10.1109/IROS.2011.6095096. 4.1.1

Michael Kasper, Gernot Fricke, Katja Steuernagel, and Ewald von Puttkamer. A behavior-based mobile robot architecture for learning from demonstration. *Robotics and Autonomous Systems*, 34(2):153–164, February 2001. 3.2.3

H. J. Kim, Michael I. Jordan, Shankar Sastry, and Andrew Y. Ng. Autonomous helicopter flight via reinforcement learning. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 799–806. MIT Press, 2004. URL http://papers.nips.cc/paper/2455-autonomous-helicopter-flight-via-reinforcement-learning.pdf. 2.3.2

J. Kober and J. Peters. Learning motor primitives for robotics. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 2112–2118, May 2009. doi: 10.1109/ROBOT.2009.5152577. 2.3.2

J. Kober, J. Andrew (Drew) Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, July 2013. 2.3.2

Seung kook Yun. Compliant manipulation for peg-in-hole: Is passive compliance a key to learn contact motion? In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1647–1652, May 2008. doi: 10.1109/ROBOT.2008.4543437. 4.1.1

P. Kormushev, S. Calinon, and D. G. Caldwell. Robot motor skill coordination with EM-based reinforcement learning. In *Proc. IEEE/RSJ Intl Conf. on*

Intelligent Robots and Systems (IROS), pages 3232–3237, Taipei, Taiwan, October 2010a. 2.3.2

P. Kormushev, S. Calinon, R. Saegusa, and G. Metta. Learning the skill of archery by a humanoid robot icub. In *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, pages 417–423, Dec 2010b. doi: 10.1109/ICHR.2010.5686841. 2.3.2

Hanna Kurniawati, David Hsu, and Wee Sun Lee. Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces. In *In Proc. Robotics: Science and Systems*, 2008. 2.3.1

Pamela Banta Lavenexa, Valérie Boujonb, Angélique Ndarugendamwob, and Pierre Lavenexa. Human short-term spatial memory: Precision predicts capacity. 3.2.1

Alan M. Leslie. *ToMM,ToBY, and Agency: Core architecture and domain specificity*. Cambridge University Press, 1994. 3.2.2

Miao Li, Kaiyu Hang, Danica Kragic, and Aude Billard. Dexterous grasping under shape uncertainty. *Robotics and Autonomous Systems*, 75, Part B:352 – 364, 2016. ISSN 0921-8890. doi: http://dx.doi.org/10.1016/j.robot.2015.09.008. URL http://www.sciencedirect.com/science/article/pii/S0921889015001967. 2.3.4

Xin Li, William K. Cheung, and Jiming Liu. Improving POMDP Tractability via Belief Compression and Clustering. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(1):125–136, February 2010. ISSN 1083-4419. doi: 10.1109/tsmcb.2009.2021573. URL http://dx.doi.org/10.1109/tsmcb.2009.2021573. 2.3.1

Georgios Lidoris. *State Estimation, Planning, and Behavior Selection Under Uncertainty for Autonomous Robotic Exploration in Dynamic Environments*. kassel university press GmbH, Kassel University, 2011. 3.2.3

Yong Lin, Xingjia Lu, and Fillia Makedon. Approximate planning in pomdps via MDP heuristic. In *2014 International Joint Conference on Neural Networks, IJCNN 2014, Beijing, China, July 6-11, 2014*, pages 1304–1309. IEEE, 2014. doi: 10.1109/IJCNN.2014.6889576. URL http://dx.doi.org/10.1109/IJCNN.2014.6889576. 2.3.4

Michael L. Littman, Anthony R. Cassandra, and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *International Conference on Machine Learning (ICML)*. Morgan Kaufmann, 1995. URL http://people.csail.mit.edu/lpk/papers/ml95.ps. 2.3.4

M. Jokesch J. Suchy M. Bdiwi, A. Winkler. Improved peg-in-hole (5-pin plug) task: Intended for charging electric vehicles by robot system automatically. In *Proc. of 12th IEEE International Multi-Conference on Systems, Signals and Devices*, 2015. 4.1.1

W. Meeussen, M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, V. Eruhimov, T. Foote, J. Hsu, R. B. Rusu, B. Marthi, G. Bradski, K. Konolige, B. Gerkey, and E. Berger. Autonomous door opening and plugging in with a personal robot. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 729–736, May 2010. doi: 10.1109/ROBOT.2010.5509556. 4.1.1

George Miller. The magical number seven, plus or minus two: Some limits on our capacity for processing information, 1956. URL http://cogprints.org/730/. One of the 100 most influential papers in cognitive science: http://cogsci.umn.edu/millennium/final.html. 3.2.1

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 02 2015. URL http://dx.doi.org/10.1038/nature14236. 2.3.1

B. Nemec, F. J. Abu-Dakka, B. Ridge, A. Ude, J. A. Jørgensen, T. R. Savarimuthu, J. Jouffroy, H. G. Petersen, and N. Krüger. Transfer of assembly operations to new workpiece poses by adaptation to the desired force profile. In *Advanced Robotics (ICAR), 2013 16th International Conference on*, pages 1–7, Nov 2013. doi: 10.1109/ICAR.2013.6766568. 4.1.1

Gerhard Neumann and Jan R. Peters. Fitted q-iteration by advantage weighted regression. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 1177–1184. Curran Associates, Inc., 2009. 4.4.2

Andrew Y. Ng and Michael Jordan. Pegasus: A policy search method for large mdps and pomdps. In *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*, UAI'00, pages 406–415, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc. ISBN 1-55860-709-9. URL http://dl.acm.org/citation.cfm?id=2073946.2073994. 2.3.2

Monica N. Nicolescu and Maja J. Mataric. Learning and interacting in human-robot domains. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 31(5):419–430, September 2001. 3.2.3

A. Nowé, P. Vrancx, and Y-M. De Hauwere. *Reinforcement Learning: State-of-the-Art*, chapter Game Theory and Multi-agent Reinforcement Learning, pages 441–470. Springer, 2012. URL http://www.springer.com/engineering/computational+intelligence+and+complexity/book/978-3-642-27644-6. 2.3.4

J. Nunez-Varela, B. Ravindran, and J. L. Wyatt. Where do i look now? gaze allocation during visually guided manipulation. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4444–4449, May 2012. doi: 10.1109/ICRA.2012.6225226. 2.3.4

Hyeonjun Park, Ji-Hun Bae, Jae-Han Park, Moon-Hong Baeg, and Jaeheung Park. Intuitive peg-in-hole assembly strategy with a compliant manipulator. In *Robotics (ISR), 2013 44th International Symposium on*, pages 1–5, Oct 2013. doi: 10.1109/ISR.2013.6695699. 4.1.1

Achille Pasqualotto, Mary Jane Spiller, Ashok S. Jansari, and Michael J. Proulx. Visual experience facilitates allocentric spatial representation. *Behavioural Brain Research*, 236(0):175 – 179, 2013. ISSN 0166-4328. doi: http://dx.doi.org/10.1016/j.bbr.2012.08.042. URL http://www.sciencedirect.com/science/article/pii/S0166432812005682. 3.2.1

J. Peters and S. Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008a. 4.4.3

Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7–9): 1180 – 1190, 2008b. ISSN 0925-2312. doi: http://dx.doi.org/10.1016/j.neucom.2007.11.026. URL http://www.sciencedirect.com/science/article/pii/S0925231208000532. Progress in Modeling, Theory, and Application of Computational Intelligenc15th European Symposium on Artificial Neural Networks 200715th European Symposium on Artificial Neural Networks 2007. 2.3.2

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025 – 1032, August 2003. 2.3.1

R. Platt, L. Kaelbling, T. Lozano-Perez, and R. Tedrake. Non-gaussian belief space planning: Correctness and complexity. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 4711–4717, May 2012. doi: 10.1109/ICRA.2012.6225223. 2.3.3

Robert Platt, Russell Tedrake, Leslie Kaelbling, and Tomás Lozano-Pérez. Belief space planning assuming maximum likelihood observations. In *Robotics Science and Systems Conference (RSS)*, 2010. URL http://groups.csail.mit.edu/robotics-center/public_papers/Platt10.pdf. 2.3.3

Josep M. Porta, Nikos Vlassis, Matthijs T. J. Spaan, and Pascal Poupart. Point-based value iteration for continuous pomdps. *JOURNAL OF MACHINE LEARNING RESEARCH*, 7:2329–2367, 2006. 2.3.1

S. Prentice and N. Roy. The belief roadmap: Efficient planning in belief space by factoring the covariance. *International Journal of Robotics Research*, 8 (11-12):1448–1465, December 2009. 2.3.3

Kerstin Preuschoff, Peter NC Mohr, and Ming Hsu. Decision making under uncertainty. *Frontiers in Neuroscience*, 7(218), 2013. ISSN 1662-453X. doi: 10.3389/fnins.2013.00218. URL http://www.frontiersin.org/decision_neuroscience/10.3389/fnins.2013.00218/full. 1.1

Akshara Rai, Guillaume De Chambrier, and Aude Billard. Learning from failed demonstrations in unreliable systems. In *Humanoid Robots (Humanoids), 2013 13th IEEE-RAS International Conference on*, pages 410–416. IEEE, 2013. 1.2.2

Edward J. Sondik Richard D. Smallwood. The optimal control of partially observable markov processes over a finite horizon. *Oper. Res.*, 21(5):1071–1088, October 1973. ISSN 0030-364X. doi: 10.1287/opre.21.5.1071. URL http://dx.doi.org/10.1287/opre.21.5.1071. 2.2.1

Hilary Richardson, Chris Bake, Joshua. Tenenbaum, and Rebecca Saxe. The development of joint belief-desire inferences. In *Proceedings of the 34th Annual Meeting of the Cognitive Science Society (COGSCI)*, August 2012. 3.2.2

Martin Riedmiller. Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method. In *In 16th European Conference on Machine Learning*, pages 317–328. Springer, 2005a. 4.4.2

Martin Riedmiller. Neural fitted q iteration – first experiences with a data efficient neural reinforcement learning method. In *In 16th European Conference on Machine Learning*, pages 317–328. Springer, 2005b. 2.3.1

Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-draa. Online planning algorithms for pomdps. *J. Artif. Int. Res.*, 32(1):663–704, July 2008. ISSN 1076-9757. URL http://dl.acm.org/citation.cfm?id=1622673.1622690. 2.3.1

N. Roy, W. Burgard, D. Fox, and S. Thrun. Coastal navigation-mobile robot navigation with uncertainty in dynamic environments. In *IEEE International Conference on Robotics and Automation*, pages 35–40, 1999. 2.3.4, 3.5.2

Nicholas Roy. Finding Approximate POMDP solutions Through Belief Compression. *Journal of Artificial Intelligence Research*, 23, 2005. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.35.6180. 2.3.1

Nicholas Roy and Geoffrey J Gordon. Exponential family pca for belief compression in pomdps. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 1667–1674. MIT Press, 2003. URL http://papers.nips.cc/paper/2319-exponential-family-pca-for-belief-compression-in-pomdps.pdf. 2.3.1

Nicholas Roy and Sebastian Thrun. Coastal navigation with mobile robots. In *In Advances in Neural Processing Systems 12*, pages 1043–1049, 1999. 2.3.1

Brian Scassellati. Theory of mind for a humanoid robot. *Auton. Robots*, 12(1):13–24, January 2002. ISSN 0929-5593. doi: 10.1023/A:1013298507114. URL http://dx.doi.org/10.1023/A:1013298507114. 3.2.2

Stefan Schaal, Jan Peters, Jun Nakanishi, and Auke Ijspeert. Learning movement primitives. In *International Symposium on Robotics Research (ISRR2003*. Springer, 2004. 4.1.1

David Silver, J. Andrew Bagnell, and Anthony Stentz. Learning from demonstration for autonomous navigation in complex unstructured terrain. *IJRR*, 29(12):1565–1592, October 2010. 3.2.3

Trey Smith and Reid Simmons. Heuristic search value iteration for pomdps. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, UAI '04, pages 520–527, Arlington, Virginia, United States, 2004. AUAI Press. 2.3.1

Trey Smith and Reid G. Simmons. Point-based POMDP algorithms: Improved analysis and implementation. *CoRR*, abs/1207.1412, 2012. URL http://arxiv.org/abs/1207.1412. 2.3.1

Beate Sodian and Susanne Kristen. Theory of mind. In Britt Glatzeder, Vinod Goel, and Albrecht Müller, editors, *Towards a Theory of Thinking*, On Thinking, pages 189–201. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-03128-1. doi: 10.1007/978-3-642-03129-8_13. URL http://dx.doi.org/10.1007/978-3-642-03129-8_13. 3.2.2

Matthijs T. J. Spaan and Nikos Vlassis. Planning with continuous actions in partially observable environments. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3469–3474, Barcelona, Spain, 2005. 2.3.1

C. Stachniss, G. Grisetti, and W. Burgard. Information gain-based exploration using rao-blackwellized particle filters. In *Proc. of Robotics: Science and Systems (RSS)*, Cambridge, MA, USA, 2005. 2.3.4

B.J. Stankiewicz, G.E. Legge, J.S. Mansfield, and E.J. Schlicht. Lost in virtual space: Studies in human and ideal spatial navigation. *Journal of Experimental Psychology: Human Perception and Performance.(under review)*, 32(3):688–704, 2006. 1.1, 3.2.1, 3.2.1

F. Stulp, E. Theodorou, M. Kalakrishnan, P. Pastor, L. Righetti, and S. Schaal. Learning motion primitive goals for robust manipulation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 325–331, Sept 2011. doi: 10.1109/IROS.2011.6094877. 2.3.2

F. Stulp, E. A. Theodorou, and S. Schaal. Reinforcement learning with sequences of motion primitives for robust manipulation. *IEEE Transactions on Robotics*, 28(6):1360–1370, Dec 2012. ISSN 1552-3098. doi: 10.1109/TRO.2012.2210294. 2.3.2

Wen Sun and R. Alterovitz. Motion planning under uncertainty for medical needle steering using optimization in belief space. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 1775–1781, Sept 2014. doi: 10.1109/IROS.2014.6942795. 2.3.3

H.G Sung. *Gaussian Mixture Regression and Classification*. PhD thesis, Rice University, 2004. 4.5

R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*, volume 116. Cambridge Univ Press, 1998. 4.4.3

Sebastian Thrun. Monte carlo POMDPs. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller, editors, *Advances in Neural Information Processing Systems (NIPS 1999)*, pages 1064–1070. MIT Press, 2000. ISBN 0-262-19450-3. URL `http://robots.stanford.edu/papers/thrun.mcpomdp.pdf`. 2.3.1

Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005. ISBN 0262201623. 2.2.1, 2.3.1, 2.3.4

Joan Vallve and Juan Andrade-Cetto. Dense entropy decrease estimation for mobile robot exploration. In *2014 IEEE International Conference on Robotics and Automation, ICRA 2014, Hong Kong, China, May 31 - June 7, 2014*, pages 6083–6089, 2014. doi: 10.1109/ICRA.2014.6907755. 2.3.4

Jur Van Den Berg, Pieter Abbeel, and Ken Goldberg. Lqg-mp: Optimized path planning for robots with motion uncertainty and imperfect state information. *Int. J. Rob. Res.*, 30(7):895–913, June 2011. ISSN 0278-3649. doi: 10.1177/0278364911406562. URL `http://dx.doi.org/10.1177/0278364911406562`. 2.3.3

Jur van den Berg, Sachin Patil, and Ron Alterovitz. Motion planning under uncertainty using iterative local optimization in belief space. *The International Journal of Robotics Research*, 31(11):1263–1278, 2012. doi: 10.1177/0278364912456319. URL `http://ijr.sagepub.com/content/31/11/1263.abstract`. 2.3.3

Manuela M. Veloso, editor. *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, 2007. 2.3.1

N. A. Vien and M. Toussaint. Pomdp manipulation via trajectory optimization. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 242–249, Sept 2015. 2.3.4

Sethu Vijayakumar, Tomohiro Shibata, and Stefan Schaal. Reinforcement learning for humanoid robotics. In *Autonomous Robot*, page 2002, 2003. 2.3.2

John Von Neumann and O. Morgenstern. *The theory of games and economic behavior*. Princeton, 3 edition, 1990. 2.1.1

Jiexin Wang, Eiji Uchibe, and Kenji Doya. Em-based policy hyper parameter exploration: application to standing and balancing of a two-wheeled smartphone robot. *Artificial Life and Robotics*, 21(1):125–131, 2016. doi: 10.1007/s10015-015-0260-7. URL http://dx.doi.org/10.1007/s10015-015-0260-7. 2.3.2

R. Frances Wang. Spatial updating. *Scholarpedia*, 2(10):3839, 2007. 3.2.1

Ranxiao Frances Wang and Elizabeth Spelke. Updating egocentric representations in human navigation. *Cognition*, 77(12):215 – 250, 2000. URL http://www.wjh.harvard.edu/~lds/pdfs/wang2000.pdf. 3.2.1

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, 1992. doi: 10.1007/BF00992696. URL http://dx.doi.org/10.1007/BF00992696. 2.3.2

David A. Winter. *Biomechanics and motor control of human movement*. 2009. 2.4

Thomas Wolbers and Mary Hegarty. What determines our navigational abilities? *Trends in Cognitive Sciences*, 14(3):138 – 146, 2010. ISSN 1364-6613. doi: http://dx.doi.org/10.1016/j.tics.2010.01.001. URL http://www.sciencedirect.com/science/article/pii/S1364661310000021. 3.2.1

Thomas Wolbers, Mary Hegarty, Christian Buchel, and Jack M Loomis. Spatial updating: how the brain keeps track of changing object locations during observer motion. *Nature Neuroscience*, 1124(1), 2008. ISSN 1097-6256. doi: http://dx.doi.org/10.1038/nn.2189. URL http://www.nature.com/neuro/journal/v11/n10/suppinfo/nn.2189_S1.html. 3.2.1

Yang Yang, Linglong Lin, Y. T. Song, Bojan Nemec, Ales Ude, Anders Glent Buch, Norbert Krüger, and Thiusius Rajeeth Savarimuthu. *Fast programming of peg-in-hole actions by human demonstration*, pages 990–995. IEEE, 2014. ISBN 978-1-4799-2537-7. doi: 10.1109/ICMC.2014.7231702. 4.1.1

C. Zito, M. S. Kopicki, R. Stolkin, C. Borst, F. Schmidt, M. A. Roa, and J. L. Wyatt. Sequential trajectory re-planning with tactile information gain for dexterous grasping under object-pose uncertainty. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4013–4020, Nov 2013. doi: 10.1109/IROS.2013.6696930. 2.3.3