Fitted Policy Iteration for a POMDP Peg-In-Hole search task.

Guillaume de Chambrier[a,1,*], Aude Billard[a]

[a]*Learning Algorithms and Systems Laboratory (LASA), École Polytechnique Fédérale de Lausanne (EPFL), Switzerland*

Abstract

Acting optimally given state uncertainty is necessary for robotic systems to achieve autonomy. If uncertainty is not considered appropriately by a policy or planner it can lead to either sub-optimal task execution or failure. We consider a Peg-in-Hole (PiH) search task in which both a human teacher and robot apprentice must locate an electric socket and connect it without using any vision, making the state space partially observable, whilst relying on haptic and proprioceptive information. A search policy can be obtained by applying dynamic programming to the Partially Observable Markov Decision Process (POMDP) formulation of task. This quickly becomes infeasible for continuous state and action spaces when autonomous exploration-exploitation strategies are used since they do not consider informative prior knowledge. We address this problem by demonstrating how human intuition can be leveraged in an Actor-Critic Fitted Policy Iteration (FPI) framework. A belief-space critic is learned offline in a Fitted RL framework from trajectory data demonstrated by a group of blindfolded human teachers. The demonstrations are compresses as a sequence of most likely state and entropy extracted from a Point Mass Filter (PMF). The critic is then used to train the actor policy represented as Gaussian Mixture Model (GMM). Evaluations performed both in simulation and on the KUKA LWR robot showed that the proposed algorithm outperforms both a myopic and a purely data driven policy in terms of distance travelled to localise the socket and perform better than these two alternative policies when the socket has no distinctive features.

Keywords: Fitted Reinforcement Learning, Actor-Critic, POMDP, GMM policy, Programming by Demonstration

1. Introduction

The ability to act optimally given state uncertainty is paramount for all robotic systems acting in environments which are not fully observable. Depending on the task and structure of the state uncertainty if it is not taken into consideration by the control policy can lead to wasteful usage of resources and even failure. Given the potential adverse consequences (disastrous if considering a search and rescue task), it is important to design uncertainty robust policies and planners.

The generic solution to such an optimal control problem is to formulate the task as a Partially Observable Markov Decision Process (POMDP) which is subsequently solved by dynamic programming or reinforcement learning if the transition and observation models are unavailable. However, solving a POMDP directly is infeasible even for the simplest problems [33], which lead to the development of approximate methods.

Advances have been made in applying approximate POMDP algorithms to robotic applications [12]. However the optimisation often requires a discretisation of the action space which is restrictive for tasks which are naturally continuous. In this case a local optimisation with quantifiable actions (macro) [44] or alternatively heuristic approaches [24], based on the most likely state, can be applied. These approaches trade-off global optimality for faster approximate solutions which depending on the problem is often close to optimality.

Solving continuous action POMDPs via reinforcement learning is difficult as the autonomous exploration, traditionally used in RL, becomes quickly inefficient. This leads to the development of planning methods which guarantee local optimality.

In this paper we propose an approximate POMDP method for continuous belief-state and action space. We introduce a Fitted Policy Iteration (FPI) Actor-Critic (AC) Reinforcement Learning (RL) method in which sample episodes (demonstrations) are provided by human teachers in a Programming by Demonstration (PbD) framework. It is assumed that a good mixture of explorative-exploitative behaviour is present in the demonstrated trajectory data set. It has been previously shown [10] that humans exhibit both risk-prone and averse behaviour which constitutes an ideal training set for RL, removing the need for costly autonomous exploration. A belief-space value function is learned from these human demonstrations using a Fitted RL approach and it is used to learn a Gaussian Mixture Model control policy.

We consider a plug power-socket search and connection task, also known as Peg-in-Hole (PiH), in which a robot apprentice must localise a power socket and establish a connection. No vision system is used during the task and only haptic information, provided via a force-torque sensor mounted on the end-effector of the robot. This choice is motivated by two reasons: to validate that humans can be viable expert teachers under these conditions. The second is that PiH is a very important component

*Corresponding author

Email addresses: guillaume.dechambrier@epfl.ch (Guillaume de Chambrier), aude.billard@epfl.ch (Aude Billard)

in manufacturing processes and we seek to demonstrate that this task can be accomplished without the need of a costly vision system.

This paper is organised as follows: Section 2 overviews the Peg-in-hole (PiH) and Actor-critic Fitted Reinforcement Learning literature. Section 3 details the PiH-search task, the formulation of the belief space and the recorded data. Section 4 presents the Fitted Policy Iteration (FPI) algorithm. Section 5 details the control architecture. Section 6 describes the experiments conducted to evaluate the FPI in the PiH-search task. Section 7, provides a discussion and the conclusion.

2. Background

2.1. POMDP

A Partially Observable Markov Decision Process (POMDP) is a generic framework for formulating a temporal decision process given that the state space is not directly observable [39]. A POMDP is defined by the tuple $\{X, U, Y, T, \Omega, R, \gamma\}$, where X, U and Y are the state, action and observation spaces (which can be discrete or continuous); $T := p(x_t|x_{t-1}, u_{t-1})$ is the state transition probability distribution; $\Omega := p(y_t|x_t)$ is the observation model which gives the probability of a measurement $y_t \in Y$ given a state $x_t \in X$; $R(x_t) \in \mathbb{R}$ is the reward function which gives the utility of a state and $\gamma \in (0, 1]$ is the discount factor. As the state space is not observable the agent must consider the entire history $h := \{y_{0:t}, u_{1:t-1}\}$ of measurements and actions when deciding which action $u_t \in U$ to take [36]. Instead of memorising the entire history h_t, it can be iteratively integrated into a belief/information state $b_t := p(x_t|h)$, which is a probability distribution over the state space, without losing any information [20]. This leads to a reformulation of the tuple as a *belief*-MDP $\{\mathcal{B}, U, \tau, R_B, \gamma\}$, where $b_t \in \mathcal{B}$ is the set of all possible beliefs and τ is a belief state transition function $b_t = \tau(b_{t-1}, u_{t-1}, y_t)$ which can be any Bayesian state space filter (eg. an Extended Kalman Filter or a Particle Filter). Both the state T and observation Ω models become part of τ. The belief reward function R_B becomes a function of the state reward function 1.

$$R_B(b) = \sum_{x \in X} b(x)\, R(x) \tag{1}$$

The advantage of the *belief*-MDP formulation is that dynamic programming and reinforcement learning can be applied as $\mathcal{B}$ is observable. The objective is to find a policy Equation 2 which is maximises the infinite-horizon expected reward, Equation 3, where $r_t \in R_B$.

$$\pi_{\boldsymbol{\theta}} : b \mapsto u \tag{2}$$

$$V^{\pi_\theta}(b) = \mathbb{E}_{\pi_\theta}\left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \middle| b_t = b \right\} \tag{3}$$

which can be iteratively evaluated with on-policy value iteration, Equation 4, and an optimal policy can be found by Generalized Policy Iteration [41, Chap. 4.6].

$$V^{\pi_\theta}(b) = R_B(b) + \sum_{u \in U} \pi_{\boldsymbol{\theta}}(b, u)\, \gamma \sum_{y \in Y} p(y|b, u)\, V^{\pi_\theta}(\tau(b, u, y)) \tag{4}$$

Solving discrete state space POMDP problems is a difficult as the number of parameters of the value function grows exponentially with respect to the decision horizon [42, Chap. 15][39], a by-product of preserving the Piece Wise Linear and Convex (PWLC) property of the value function. Recent advances for discrete POMDPs are Point-based Value Iteration (PBVI) methods [33], which focus on pruning, belief selection and exploration strategies, see [43], [14]. However these Value Iteration (VI) approaches do not naturally transfer to a continuous domains [34].

Policy Search (PS) methods [13] directly optimised the policy's parameters without estimating a value function. The advantage is that convergence (local) can be guaranteed and the policy is smooth as taking the derivative of the value function, even when small approximation errors are present, can lead to sub-optimal greedy policies [4]. Policy search methods work well when the number of parameters are relatively low but as the number of parameters increase the learning becomes slow as the variance of the policy's gradient estimate is high [17].

Actor-critic (AC) methods [41, Chap. 6.6],[17] have two separate parameterisations, one for the policy (actor) and one for the value function (critic). The critic is used to update both the policy and value function parameters such to maximise the expected reward. Having a separate policy and value function guarantees a smooth control output, ideal for a robotic controller, whist faster convergence as the critic supplies the actor with low-variance knowledge of the performance.

2.2. Related work

There are two research domains which are closely related this work;

2.2.1. Actor-Critic & Fitted Reinforcement Learning

AC is advantageous in that the policy can be chosen which is computationally efficient in evaluating actions whilst the value function can have a more complex representation.

To guarantee convergence (in model based RL) during temporal difference learning, the value function approximator has to be an averager (tile coding, k-nearest-neighbour, locally weighted averaging) [16]. The extension to a model-free approach with a kernel function approximator (locally weighted averaging, the kernel is a Gaussian function) known as Kernel-Based Approximate Dynamic Programming (KBDP) [30] has proven to be globally optimal in a continuous-space framework. This leads to the wider application of Batch RL methods such as Fitted Value Iteration (FVI) [7] and Fitted Q-Iteration (FQI) [15] (Q-approximator is a random forest ensemble), [28] in RL problems. By remembering all the state transition pairs and by applying multiple synchronous Dynamic Programming (DP) and function approximation updates, the problem of diverging value function approximators is resolved.

By retaining all the data, it is easy in practice to apply function approximators which are not averagers, such as neural networks, to RL problems. A successful example is Neural Fitted Q-Iteration (NFQI) [35] which uses a multi-layer perceptron to represent the Q-function for the cart-pole and mountain car problems and shows rapid convergence to optimal policies. It has since been used in many extensions, [31], [2]. This has lead to the application of more sophisticated regression methods such as Deep Fitted Q-iteration (DFQ) [23] which is used to learn visual control policies and with recent work including learning to play ATRI and ping-pong games [26], [19].

The reader is referred to [8] and [45, Chap 2] for a literature review which includes a taxonomy of Batch RL methods and a concise description of Batch RL from its origins, through how it became popular with Fitted RL approaches and its continuation into Deep Learning.

The Fitted Policy Iteration which we apply to our belief space PiH search task is part of this family of methods. We chose an on-policy approach to avoid the maximisation over the actions, as we are in continuous action space. A Gaussian Mixture Model is used to parameterise the policy and a Locally Weighted Regression (LWR) as the value function approximator.

2.2.2. Peg-in-hole

The Peg-in-Hole (PiH) task is one of the most widespread steps in industrial processes, with examples including the assembly of vehicular transmission components [38] and valves [11]. To be successful, the estimated position of the robot's end-effector and workpiece must be precise, as the clearance between peg and hole is very small.

All approaches use to some extent a vision system [25] to estimate the position of the workpiece. Given the peg's estimated position with respect to the hole an insertion strategy has to be carried out since the hole is often occluded by the robot's manipulator. One approach is to apply blind search patterns, such as circular motions [38], which do not consider the actual state uncertainty. These approaches work well when the peg is within the vicinity of the hole. In this work, with no visual information, high state uncertainty makes the direct application of such blind search methods ill-suited.

Another approach consists of learning task space policies and gradually adapting the parameters based on a reference Force/Torque profile. In [46] the authors learned a time-dependent Dynamic Movement Primitive (DMP) [37] Cartesian end-effector policy for the Cranfield benchmark object from human teleoperated demonstrations. Similarly in [27, 1], a F/T profile is encoded separately by a regressor function along the DMP policy. Successive refinements of the DMP policy are achieved through using force feedback to adapt the parameters of an admittance controller such so as to reproduce the same F/T (encoded by a separate regressor).

Reproducing exactly the same force torque profile for the complete trajectory could be unnecessary as the force torque profile is used predominantly during the final stage of the PiH task, to avoid jamming during insertion [22, Chap. 5].

Reinforcement learning has also been applied to PiH. In [21] a DMP policy is initialised with kinesthetic demonstrations of picking up a pen. The recorded Cartesian trajectories are encoded in a parameterised DMP policy and augmented with a F/T profile. After 110 trials the policy was found to be a 100% successful. In [18] a 18 dimensional input and a 6 dimensional output (linear and angular velocity) neural network is learned and successful after a 100 episodes. This work has a similar approach, however instead of considering autonomous rollouts common in RL, relies solely on the data provided by human teachers.

3. Experiment methods

Figure 1 (*Top-right*), illustrates the PiH-search task. The orange area represents the teacher's starting area and is assumed prior knowledge. The sockets are always positioned at the center of a fake wall (wooden plank) which is clamped to a table. 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 1 (*Top-left*) for an illustration.

The human teacher holds the plug which is attached to a cylindrical handle with an ATI 6 axis force torque sensor (Nano25 [2]) to provide **raw** wrench $\phi \in \mathbb{R}^6$ measurements. We define the **actual** measurement to be a function of the raw wrench, $\tilde{y}_t = h(\phi_t)$, which is a binary feature vector. The feature vector encodes whether a contact is present and the direction in which it occurs, which is discretized to the four cardinalities.

On top of the cylinder there is a set of markers used by a motion capture system OptiTrack[3] (which has millimeter tracking accuracy) to measure both linear, $\dot{x} \in \mathbb{R}^3$, and angular velocity, $\omega \in \mathbb{R}^3$, at each time step which is recorded at a rate of 100 Hz along with the F/T information.

The human's location belief is represented by a probability density function (pdf) which is assumed to be uniformly distributed in the orange area. All subsequent beliefs can be inferred from the measured velocity and measurements provided by the ATI and OptiTrack sensors.

3.1. Belief state

The belief probability density function, $p(x_t|y_{0:t}, \dot{x}_{1:t})$, is a Point Mass Filter (PMF) [5, p.87], which is a non-parametric Bayesian filter. Figure 2 (*Left*) illustrates different time segments of the location belief recording during a demonstration. Figure 2 (*Right*) illustrates the likelihood when an edge is sensed. A PMF is chosen to represent the believed location of the plug as the sensing likelihoods are non-gaussian and lead to multi-modal distributions.

As pdf is high dimensional it is impractical to directly learn a statistical policy $\pi_\theta : p(x_t|y_{0:t}, \dot{x}_{1:t}) \rightarrow \dot{x}$ without some form of

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

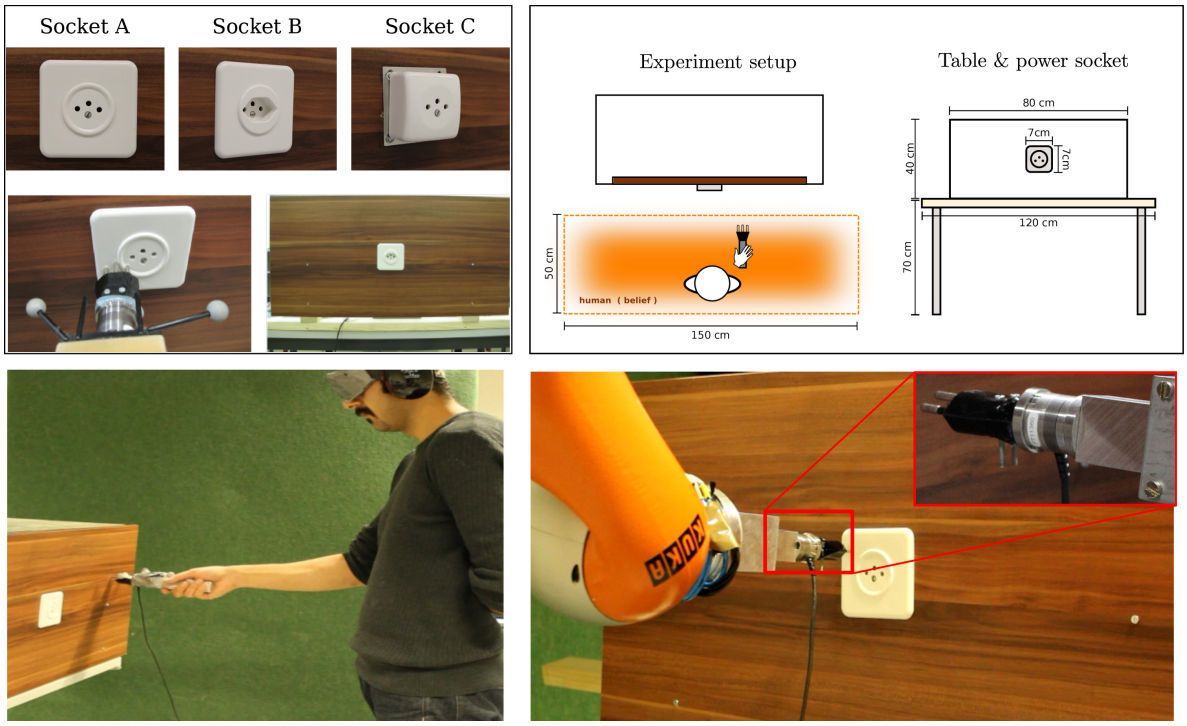


Figure 1: **Peg-in-Hole search task setup**. *Top-left*: Three different sockets are used, socket A will be only used to gather training data whilst socket B and C will be used for evaluation purposes. *Top-right:* Dimensions of the the wall and socket, the orange area illustrates the possible locations in which the human teacher will start the search. *Bottom-left:* A participant (human teacher) is blindfolded and placed within the orange rectangular area always facing the wall. He is holding a cylinder equipped with a peg and an ATI force torque sensor and OptiTrack markers. See Video 1 for an illustrate of a human subject performing the search task. *Bottom-right:* The KUKA LWR robot is equipped with a peg holder mounted with an ATI force torque sensor, it is reproducing a search and connection policy learned from the human demonstrations. See Video 2 for an illustration of the KUKA searching for the socket and then establishing a connection.

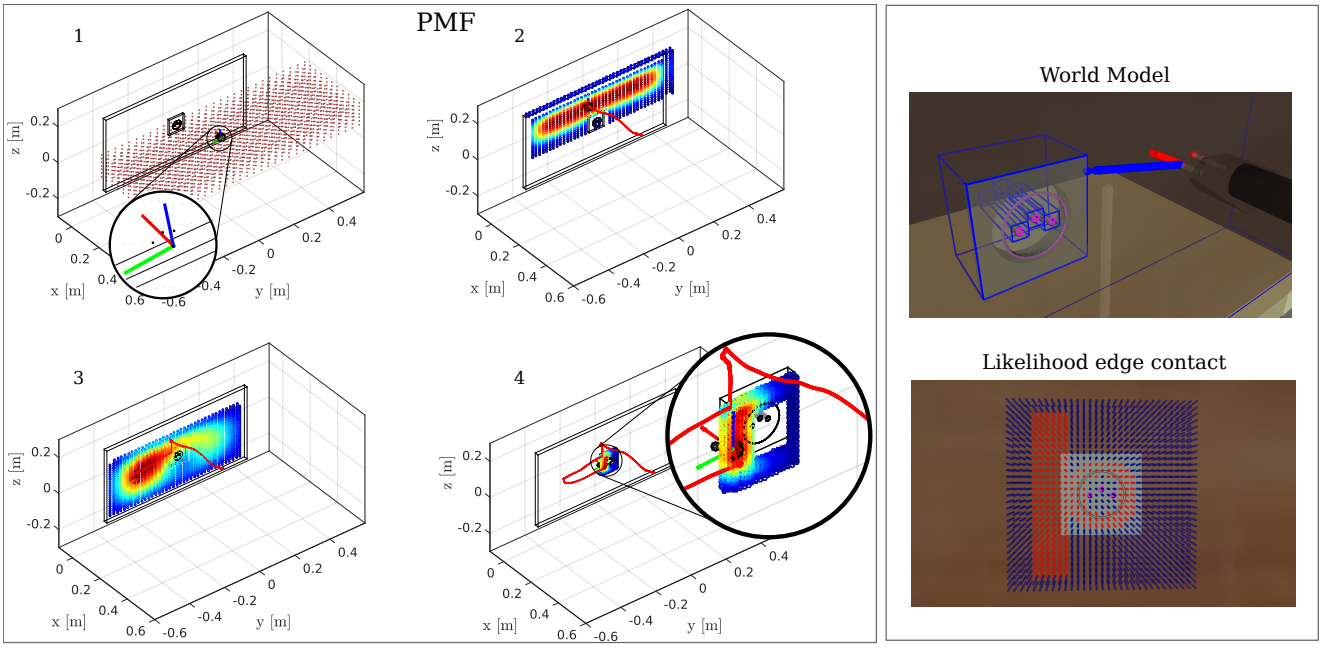


Figure 2: *Left:* Point Mass Filter (PMF) update of a particular human demonstration. (1) Initial uniform distribution spread over the starting region. Each grid cell represents a hypothetical position of the plug. The orientation is assumed to be known. (2) First contact, the distribution is spread 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. See Video 3 for an illustration of the PMF for a subject's search. *Right*: **World model**: The plug is modelled 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.

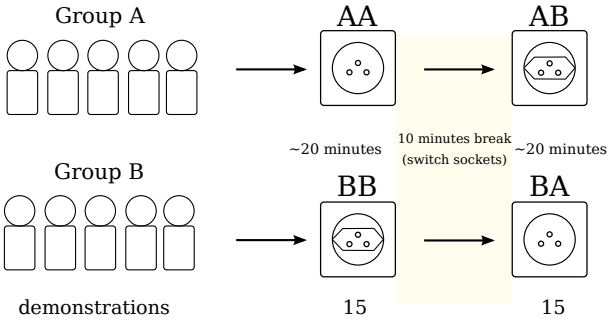


Figure 3: Experiment protocol. The participants are divided in two groups of 5, Group A begins with socket A and after a short break repeats the task with socket B. The same logic holds for Group B. For each socket 15 executions of the task are recorded.

compression. We compress the pdf to a belief space vector $b_t = [\hat{x}_t, U]^{\mathrm{T}}$ composed of the maximum a posteriori, $\hat{x}_t \in \mathbb{R}^3$, and the differentiation entropy, $U = H\{p(x_t|y_{0:t}, \dot{x}_{0:t})\} \in \mathbb{R}$.

Each participant's demonstration results in a dataset $D = \{\dot{x}^{[i]}_{1:T}, \omega^{[i]}_{1:T}, \phi^{[i]}_{1:T}, b^{[i]}_{1:T}\}$, where the upper index $[i]$ references the ith search trajectory (one episode) and subscript $1:T$ denotes the time steps during the trajectory from initialisation $t = 1$ until the end $t = T$.

3.2. Participants and experiment protocol

To perform the PiH search tasks we recruited 10 student volunteers to be teachers (all male Master's and PhD students). The participants were aged between 24 and 30 with an average age of 26 years and a standard deviation of 2.4 years. Each participant carried out 30 demonstrations of the PiH search-task and each session lasted approximately 50 minutes and never exceeded one hour. The 10 participants were divided equally in two groups, A and B. Each member of group A began by performing 15 PiH searches with socket A (AA), followed by a 10 minute break, finishing with an additional 15 searches with socket B (AB). The members of group B performed the same protocol starting with socket B (BB) and ending with socket A (BA). Figure 3 summarises a walk through of the experiment. The only exclusion criteria was the inability of the subject to accomplish the task. All participants gave written consent for taking part in this study. A total of 300 demonstrations were gathered. Both groups A and B took 9 ± 10s to find the socket's edge, regardless of the socket type. This is to be expected since the sockets are at the same location. It took a further 8 ± 7s on average for group B to connect socket B and 12 ± 10s on average for group A to connect socket A. As we can see this is not a straight forward task when considering the sensory deprivation. See Figure 4 (*Bottom*) for the time taken to connect the plug to the socket.

4. Learning Actor and Critic

Two policies are learned: one to map belief space to linear velocity $\pi_{\theta_1} : b_t \mapsto \dot{x}_t$ and other to map sensed wrench to angular velocity, $\pi_{\theta_2} : \phi_t \mapsto \omega_t$. The belief policy π_{θ_1} is learned in a Fitted Actor-Critic framework and the wrench policy π_{θ_2}

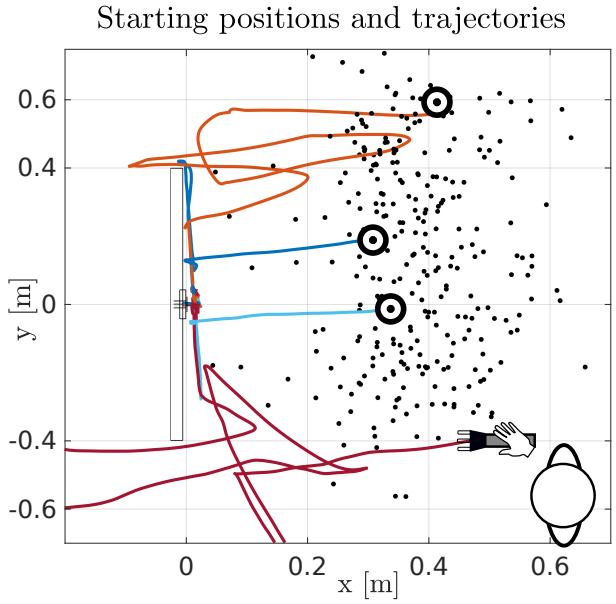


Figure 4: *Top*: Black points represent the starting position of the end-effector for all the demonstrations. Four trajectories are illustrated.

directly from the demonstrated data as was done in [22, Chap. 5]. This proves to be efficient in overcoming jamming during the final insertion step of PiH. We maximise the parameters of the policy, $\pi_{\theta_1} : b \mapsto \dot{x}_t$, with respect to the value function:

$$V^{\pi_{\theta_1}}(b) = \mathbb{E}_{\pi_{\theta_1}}\left\{ \sum_{t=0}^{\infty} \gamma^t r_{t+1} | b_t = b, \pi_{\theta_1} \right\} \tag{5}$$

where $r_t \in \mathbb{R}$ is the reward and $\gamma \in [0, 1)$ the discount factor. It is the expected future reward given the current belief state and policy. For our PiH-search task a reward of $r = 0$ is assigned at each time step until the goal (plug-socket connection) is achieved, where a reward of $r_T = 100$ is obtained. Given the continuous nature and dimensionality of the belief space Locally Weighted Regression [3] (LWR) is used as a function approximator of the value function, $V^{\pi}(b)$.

In an Actor-Critic setting, the temporal difference error, $\delta^{\pi}_t = r_{t+1} + \gamma V^{\pi}(b_{t+1}) - V^{\pi}(b_t)$, of the value function is used as a learning signal to update simultaneously itself and the actor.

4.1. Actor & Critic

Both the linear and angular velocity policies are parameterised by a Gaussian Mixture Model (GMM), Equation 6.

$$\pi_{\theta_1}(\dot{x}, b) = \sum_{k=1}^{K} w^{[k]}\, g(\dot{x}, b; \boldsymbol{\mu}^{[k]}, \boldsymbol{\Sigma}^{[k]}) \tag{6}$$

The parameters $\boldsymbol{\theta}_1 = \{w^{[k]}, \boldsymbol{\mu}^{[k]}, \boldsymbol{\Sigma}^{[k]}\}_{1,\ldots,K}$, are the weights, means and covariances of the individual Gaussian functions, $g(.)$,

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

where $\sum_k w^{[k]} = 1$, $\boldsymbol{\mu}_{\dot{x}}^{[k]} \in \mathbb{R}^3$ and $\boldsymbol{\mu}_b^{[k]} \in \mathbb{R}^4$. In both cases the Bayesian Information Criterion is used to determine the number of Gaussian functions. The next section shows how the parameters of $\pi_{\boldsymbol{\theta}_1}$ can be adapted by the value function.

4.2. Fitted Policy Iteration

Policy evaluation. To learn the value function we take Fitted RL [15] approach is taken. This is an offline method which applies multiple sweeps of the Bellman backup operator over a dataset of tuples $\{(b_t^{[i]}, r_t^{[i]}, b_{t+1}^{[i]})\}_{i=1,\cdots,M}$ until the Bellman residual, $\|\hat{V}_{k+1}^{\pi}(b) - \hat{V}_k^{\pi}(b)\|$, converges, see Algorithm 1.

Algorithm 1: Fitted Policy Evaluation

input : $\epsilon, \{(b_t^{[i]}, r^{[i]}, b_{t+1}^{[i]})\}_{i=1,\cdots,M}$
output: $\hat{V}_k^{\pi}(b_t)$

while $\|\hat{V}_{k+1}^{\pi}(b) - \hat{V}_k^{\pi}(b)\| < \epsilon$ **do**
  $\hat{V}_{k+1}^{\pi}(b_t) = \text{Regress}(b, r_t + \gamma \hat{V}_k^{\pi}(b_{t+1}))$

Most Fitted RL methods have focused on learning the Q-value function directly (Fitted Q-Iteration) [29, 15, 35]. Although this solves the control problem it requires discretisation of the action space or assumes quantifiable actions, so that maximisation in the Q-Bellman backup $\max_{\dot{x}_{t+1}} \hat{Q}(\dot{x}_{t+1}, b_{t+1})$ is easily achievable. Given the dimensionality and continuity of our problem we assume this to be unrealistic and we opt for an on-policy approach mentioned above.

Policy improvement. The Actor is updated given the Critic's value function through a modification of the Maximisation step in Expectation-Maximisation (EM) for Gaussian Mixture Models. This modification is referred to as Q-EM which is strongly related to a Monte-Carlo EM-based policy search approach [13, p.50].

The reward of a demonstrated trajectory (one episode) is given by the discounted return, Equation 7,

$$R(\tau_i) = \sum_{t=0}^{\mathrm{T}^{[i]}} \gamma^t \, r(b_t^{[i]}, \dot{x}_t^{[i]}) \tag{7}$$

where $\tau_i = \{(\dot{x}_0, b_0), \cdots, (\dot{x}_T^{[i]}, b_T^{[i]})\}$ are the state-action samples of the ith episode. All policy gradient approaches seek to find a set of parameters, $\boldsymbol{\theta}$, of the Actor, which will maximise the expected reward, Equation 8,

$$\begin{aligned} J(\boldsymbol{\theta}) &= \mathbb{E}_{p_{\boldsymbol{\theta}}}\{R\} \\ &= \sum_{i=1}^{N} \underbrace{\left(\prod_{t=0}^{T^{[i]}} \pi_{\boldsymbol{\theta}}(\dot{x}_t^{[i]}, b_t^{[i]}) \right)}_{p_{\boldsymbol{\theta}}(\tau_i)} R(\tau_i) \end{aligned} \tag{8}$$

By setting the derivative to zero, the parameters which maximise the cost function $\arg\max_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ can be found. This cannot be done directly, instead the logarithmic lower bound of the cost function is maximised which results in Equation 9. See [13, p.50] for the derivation.

$$\boldsymbol{\mu}_{\mathbf{new}}^{[k]} = \frac{\sum_{j=1}^{M} \gamma_k(\mathbf{x}^{[j]}) \, Q^{\pi_{\theta'}}(\mathbf{x}^{[j]}) \, \mathbf{x}^{[j]}}{\sum_{j=1}^{M} \gamma_k(\mathbf{x}^{[j]}) \, Q^{\pi_{\theta'}}(\mathbf{x}^{[j]})}$$

$$\boldsymbol{\Sigma}_{\mathbf{new}}^{[k]} = \frac{\sum_{j=1}^{M} \gamma_k(\mathbf{x}^{[j]}) \, Q^{\pi_{\theta'}}(\mathbf{x}^{[j]}) (\mathbf{x}^{[j]} - \boldsymbol{\mu}^{[k]})(\mathbf{x}^{[j]} - \boldsymbol{\mu}^{[k]})^{\mathrm{T}}}{\sum_{j=1}^{M} \gamma_k(\mathbf{x}^{[j]}) \, Q^{\pi_{\theta'}}(\mathbf{x}^{[j]})}$$

$$w_{\mathbf{new}}^{[k]} = \frac{\sum_{j=1}^{M} Q^{\pi_{\theta'}}(\mathbf{x}^{[j]}) \, \gamma_k(\mathbf{x}^{[j]})}{\sum_{j=1}^{M} Q^{\pi_{\theta'}}(\mathbf{x}^{[j]})}$$

Figure 5: Q-EM Maximisation of the GMM parameters. We used the same notation and derivation as in [6, Chap. 9.2.2], where $\gamma_k(\mathbf{x}^{[j]})$ is the responsibility factor, denoting the probability that data point $\mathbf{x}^{[j]} = [\dot{x}^{[j]}, b^{[j]}]^{\mathrm{T}}$ belongs to Gaussian function k.

$$\nabla_{\boldsymbol{\theta}} Q(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sum_{i=1}^{N} \sum_{t=0}^{T^{[i]}} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\dot{x}_t^{[i]}, b_t^{[i]}) \, Q^{\pi_{\theta'}}(\dot{x}_t^{[i]}, b_t^{[i]}) \tag{9}$$

In the above equation $\boldsymbol{\theta}'$ are the parameters used to generate the trajectories during the E-step. In our case these are the initial demonstrations provided by the teachers. In most policy search approaches the policy is conditioned on the state space, $\pi_{\boldsymbol{\theta}}(\dot{x}_t|b_t)$. This would lead to a complex expression in the maximisation of Equation 9 and is restrictive in the case of the GMM as it fixes the state space parameters $\boldsymbol{\mu}_b^{[k]}$, $\boldsymbol{\Sigma}_{bb}^{[k]}$ (and partially $\boldsymbol{\Sigma}_{\dot{x}b}^{[k]}$) and thus greatly constrains the solution. Instead Equation 9 is optimised with respect to the joint distribution $\pi_{\boldsymbol{\theta}}(\dot{x}_t, b_t)$ and not the conditional $\pi_{\boldsymbol{\theta}}(\dot{x}_t|b_t)$. This has two benefits. Firstly, the input dimensions (the state space) are no longer fixed allowing the GMM basis functions to move and secondly the optimisation of GMM parameters is very similar to that of the traditional EM. Setting the derivative of Equation 9 to zero and solving for the parameters $\boldsymbol{\theta} = \{w, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$ a new weighted (by $Q^{\pi_{\theta'}}$) Maximisation EM step, see Figure 2, is obtained.

As for the learned value function in the policy evaluation step, the advantage function

$$A^{\pi_\theta}(\dot{x}_t, b_t) = Q^{\pi_\theta}(\dot{x}_t, b_t) - V^{\pi_\theta}(b_t) = \delta_t^{\pi_\theta} \tag{10}$$

is used as a substitute for Q^{π} which is derived from the TD error. Assuming that the estimated value function, $\hat{V}^{\pi}$, is close to the true value function V^{π}, the TD error δ^{π} is an unbiased estimate of the advantage function. Using the advantage function as means of policy search is popular with methods such as Natural Actor Critic (NAC) [32].

Each state-action sample j has an associated weight, $\delta_j \in \mathbb{R}$, where $\delta_j > 0$ means that the jth state action-pair lead to an increase in the value function and $\delta_j < 0$ lead to a decrease in the value function. The data log-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, the TD error is rescaled to be between 0 and 1.

2D example fitted policy iteration. To illustrate the mechanism of fitted policy iteration, we give a 2D example of its application, see Figure 6. The *Top-left* subfigure depicts 10 trajectories demonstrated by two teachers going from start (white circle) to goal (orange star) state. The optimal path is a straight line passing in between two obstacles. Neither teacher demonstrated the optimal straight path.

In the *Bottom-left*, a GMM is fitted $\pi_{\boldsymbol{\theta}}(\dot{x}, x)$ to the teachers' data, using the standard EM-algorithm. Taking the policy to be the output of Gaussian Mixture Regression (GMR) $\mathbb{E}\{\pi_{\boldsymbol{\theta}}(\dot{x}|x)\}$ different behaviours are obtained than those demonstrated by the human teachers. The GMR averages the different modes encoded by the Gaussian functions which results in a mixing of the original demonstrated behaviours. No trajectories of the GMR policy truly replicate the demonstrated behaviour.

In the *Top-right* subfigure, we apply fitted policy evaluation to the original demonstrated data (discount factor $\gamma = 0.99$ and reward $r_T = 1$ when the goal is reached and zero otherwise) and compute the value function.

The *Bottom-right* subfigure illustrates the GMM policy learned with the Q-EM algorithm. As the advantage function $A^{\pi}(x, \dot{x})$ is highest along the start-goal axis, data points following this gradient will have a higher weight. This results in a policy with better rollouts (closer to the optimal path) than the trajectories generated by the policy learned via standard EM.

Belief state fitted policy evaluation. FPI is applied to the data from demonstrations done on socket A. Figure 7 (*Left*) illustrates the value function of the most likely state. As expected, the value function is high closest to the socket and around the axis $z = 0$ and $y = 0$. When Q-EM is applied the Gaussian functions of the GMM will favour these locations.

Figure 7 (*Middle*-right) illustrates the best and worst trajectories in terms of the accumulated value function. It can be seen that the best trajectories (red) tend to be aligned with the socket (star position in front of socket), whilst the worst trajectories are towards the edges of the wall and tend to follow spiralling motions.

In conclusion we learned two policies, one solely from the original human demonstrations which we call GMM and the second which is the result of **one iteration** of fitted policy iteration which we call Q-EM. This compares the effect of one policy evaluation and improvement sequence have without doing any additional rollouts. This ensures that both policies are given the same amount of information.

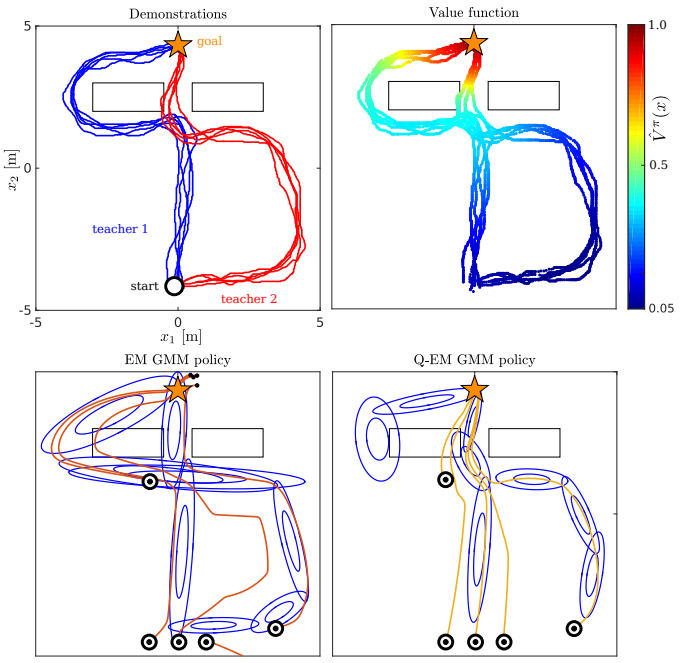


Figure 6: Fitted policy evaluation & improvement example. *Top-left:* The goal of the task is to reach the goal state. The first teacher (blue) demonstrates five trajectories which contour the obstacle in front of the goal. The second teacher (red) demonstrates 5 trajectories which initially deviate from the goal before passing between the two obstacles. *Bottom-left:* The EM algorithm is used to fit a GMM to the teachers' original data. The marginal $\pi_{\boldsymbol{\theta}}(x)$ is plotted in blue and trajectories generated by the policy $\mathbb{E}\{\pi_{\boldsymbol{\theta}}(\dot{x}|x)\}$ in orange. *Top-right Policy Evaluation:*. Value function after fitted policy evaluation terminated, the reward function is binary, $r_T = 1$ at the goal and zero otherwise, and a discount factor $\gamma = 0.99$ is used. *Bottom-right Policy Improvement:* the GMM is learned with the Q-EM algorithm in which each data point's weight is proportional to the advantage function.

5. Control architecture

The direction to search is given by the conditional:

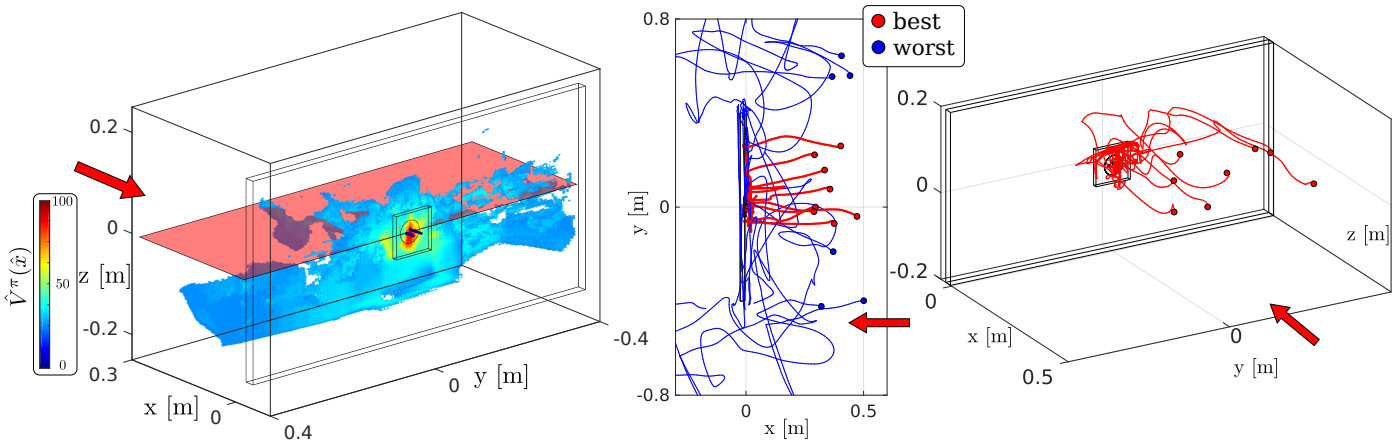


Figure 7: *Left*: LWR value function approximate $\hat{V}^{\pi}(\hat{x})$ for the most likely state $\hat{x}$. The red plane is to help visualise where the value function is above and below the axis $z = 0$. Only states with values above 0.25 are plotted. The red arrow indicates the heading of the human teacher when performing the search task. The discount factor was $\gamma = 0.99$ and the variance of the kernel variance of 1 [cm], which was set experimentally. *Middle-right*: Best and worst trajectories. The red demonstrated trajectories are the best in terms of the amount of value function gain whilst the blue are the worst. The red arrow indicates the teacher's heading. The blue trajectories tend towards the sides of the wall as the initial starting position is on the border of the wall. The red trajectories are centred along the y-axis of socket and tend to move in a straight line towards the wall whilst aligning themselves with the axis $z = 0$.

$$\pi_{\boldsymbol{\theta}}(\dot{x}|b) = \sum_{k=1}^{K} w_{\dot{x}|b}^{[k]}\, g(\dot{x}; \boldsymbol{\mu}_{\dot{x}|b}^{[k]}, \boldsymbol{\Sigma}_{\dot{x}|b}^{[k]}) \tag{11}$$

which is a distribution over the possible normalised velocities. The function $g(\cdot)$ is a multivariate Gaussian function parameterised by mean $\boldsymbol{\mu}_{\dot{x}|b}^{[k]} \in \mathbb{R}^{(3\times1)}$ and Covariance $\boldsymbol{\Sigma}_{\dot{x}|b}^{[k]} \in \mathbb{R}^{(3\times3)}$. The subscript $\dot{x}|b$ indicates that the parameters are the result of the conditional. The reader is referred to [9], [40] for a detailed derivation of the conditional of a GMM. The learned model is multi-modal, as different search velocities are possible in the same belief state. Figure 8 illustrates the multi-modal vector fields of the conditional, Equation 11. In autonomous dynamical systems control, the velocity is obtained from the expectation of the conditional, Equation 11. However, the expectation which is a weighted linear combination of the modes, could result in unobserved behaviour or no movement if the velocities cancel out. As a result we use a modified version of the expectation operator which favours the current direction, Equation 12 - 13.

$$\alpha(\dot{x}) = w_{\dot{x}|b}^{[k]} \cdot \exp(-\cos^{-1}(<\dot{x}, \boldsymbol{\mu}_{\dot{x}|b}^{[k]}>)) \tag{12}$$

$$\dot{x} = \mathbb{E}_{\alpha}\{\pi_{\boldsymbol{\theta}}(\dot{x}|b)\} = \sum_{k=1}^{K} \alpha_k(\dot{x}) \cdot \boldsymbol{\mu}_{\dot{x}|b}^{[k]} \tag{13}$$

When the applied velocity mode is no longer present another direction is sampled. For example, when the robot enters in contact with a feature, greatly reducing the uncertainty, the current mode changes and a new search direction is computed. Figure 8 illustrates the policy vector field for GMM and Q-EM, both learned from teachers demonstrations.

5.1. Robot Implementation

The GMM policy $\underline{\dot{x}} = \mathbb{E}_{\alpha}\{\pi_{\boldsymbol{\theta}}(\dot{x}|b)\}$ outputs a linear velocity which is normalised, $\underline{\dot{x}} \in \mathbb{R}^{(3\times1)}$. This search task is haptic and

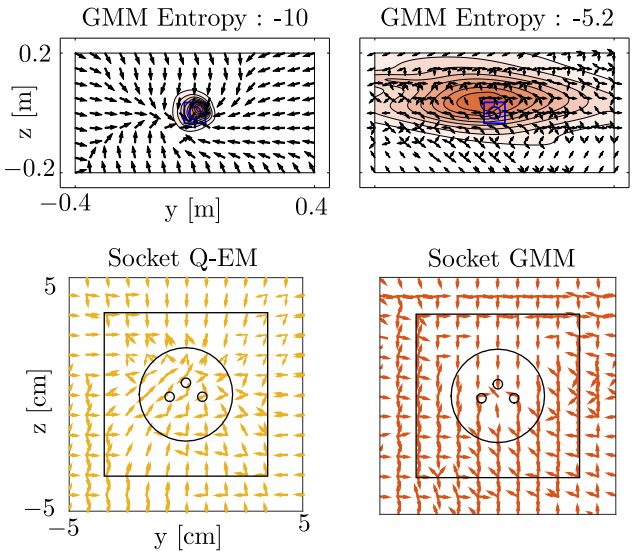


Figure 8: Q-EM and GMM policy vector fields. *Top*: The GMM policy is conditioned on an entropy of −10 and −5.2. For the lowest entropy level, most of the probability mass is close to the socket area since this level corresponds to very little uncertainty; we are already localised. We can see that the 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.

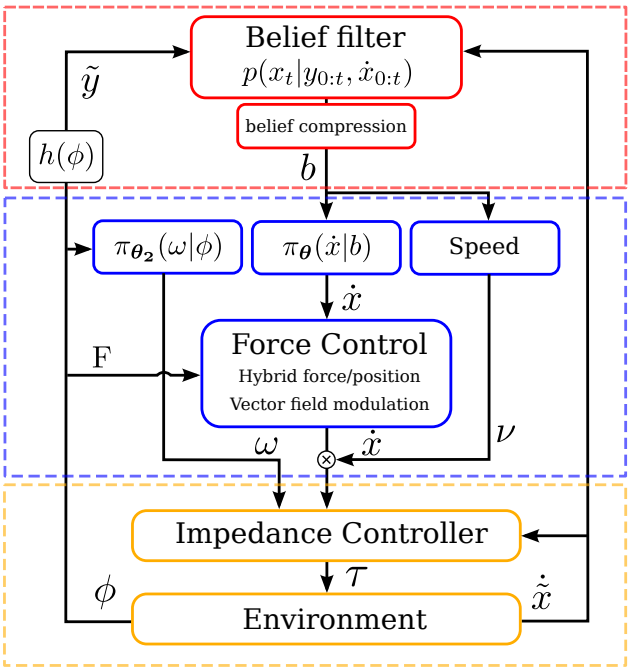


Figure 9: Control architecture. The PMF (belief) receives a measured velocity, $\dot{\tilde{x}}$, and a sensor measurement $\tilde{y}$ and is updated via Bayes rule. The belief is compressed and used by both the GMM policy and the proportional speed controller.

the end-effector of the robot is always 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. The hybrid controller targets a sensed force F_x, in the x-axis, of 3N. The y and z velocity components of the direction vector are given by Equation 13. This is insufficient for the robot to reliably surmount the edges of the socket, hence the vector field of the GMM is modulated in y and z-axis, Equation 14.

$$\dot{\underline{x}} = R_y(c(F_z) \cdot \pi/2) \cdot R_z(c(F_y) \cdot \pi/2) \cdot \dot{\underline{x}} \tag{14}$$

where R_y and R_z are (3×3) rotation matrices around the y and z-axis, and $c(F) \in [-1, 1]$ 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. Figure 9 illustrates the complete control flow.

6. Results

We evaluate the following three aspects:

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 gives an appreciation of the problem's difficulty.
2. **Importance of data** provided by human teachers. We evaluate whether it is possible to learn an improved GMM policy from Greedy demonstrations. This policy which we call Q-Greedy is used to test whether indeed human demonstrations are necessary. We evaluate whether it is possible to obtain a good policy from the two worst teachers' demonstrations as not all teachers are necessarily proficient at the task in question.
3. **Generalisation**. We learn a policy to insert a plug into socket A which is located at the center of a wooden wall. We test the generalisation of the policy in finding a new socket location and whether the policy can generalise to sockets B and C, which were not used during the training phase.

We evaluate aspects 1) and 2) purely in simulation as finding the socket requires much less precision than establishing a connection and the physics of the interaction is simple. Aspect 3), the generalisation, is evaluated both in simulation, up to the point of localising the socket's edge, and on the KUKA LWR robotic platform for the connection phase of the task. The main reason for employing the robot is that the connection phase dynamics is complex and a simulation would be unrealistic. For the robot evaluation we consider the search starting already within the vicinity of the socket.

6.1. Distance taken to reach the socket's edge

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

Experiment 1, three starting locations are chosen: *Center*, *Left* and *Right*. See Figure 10 *(Experiment 1 Top-left)*, 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. The trajectory results show a clear difference between the trajectories generated by the GMM and Q-EM policies (*Experiment 1 Bottom-left*). 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. *Experiment 1 Bottom-right*, illustrates the distribution of the first contact with the wall for the *Center* initial condition. 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.

Experiment 1 Top-right, illustrates the quantitative results of the distance taken to reach the socket for all three experiments. 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.

Experiment 2, Figure 10 (*Experiment 2*), the initial true starting positions of the end-effector are taken from a regular grid, within the red cube (see *Experiment 1*), 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' demonstrations. The Human and GMM show similar distributions of searched locations. 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.

For the Q-EM policy, the distribution of the searched locations is centred around the origin of the z-axis. 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 when it is 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.

Experiment 2 Right, it is clear that all three search policies travel less to find the socket's edge compared with the teachers' demonstrations. All search policies are better than the human teachers with the exception of group BA, 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 with the human teachers and Greedy policy. A critical assumption was made however in order to be able to use this statistical policy 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 by the GMM, the learned policy will perform poorly. Next we evaluate the validity of this assumption and the importance of the training data provided by the human teachers.

6.2. Importance of data

Two tests were performed to evaluate the importance of the teachers training data, referred to as **Experiment 3**. The worst two teachers in terms of distance taken to find the socket's edge are used to learn a GMM and Q-EM policy separately, to evaluate whether it is possible to learn a successful policy given a few bad demonstrations (15 training trajectories for each policy). In the second test a noisy explorative Greedy policy was used as a teacher to gather demonstrations which in turn were then used to learn a new policy, which we call Q-Greedy.

Figure 11 (*Top-left*) illustrates 6 trajectories of teacher # 5. Once localised, the teacher would reposition himself in front of the socket and to try to achieve an insertion. This behaviour was not expected since by losing contact with the wall, the human teacher no longer has the sensory feedback necessary to maintain an accurate position estimate.

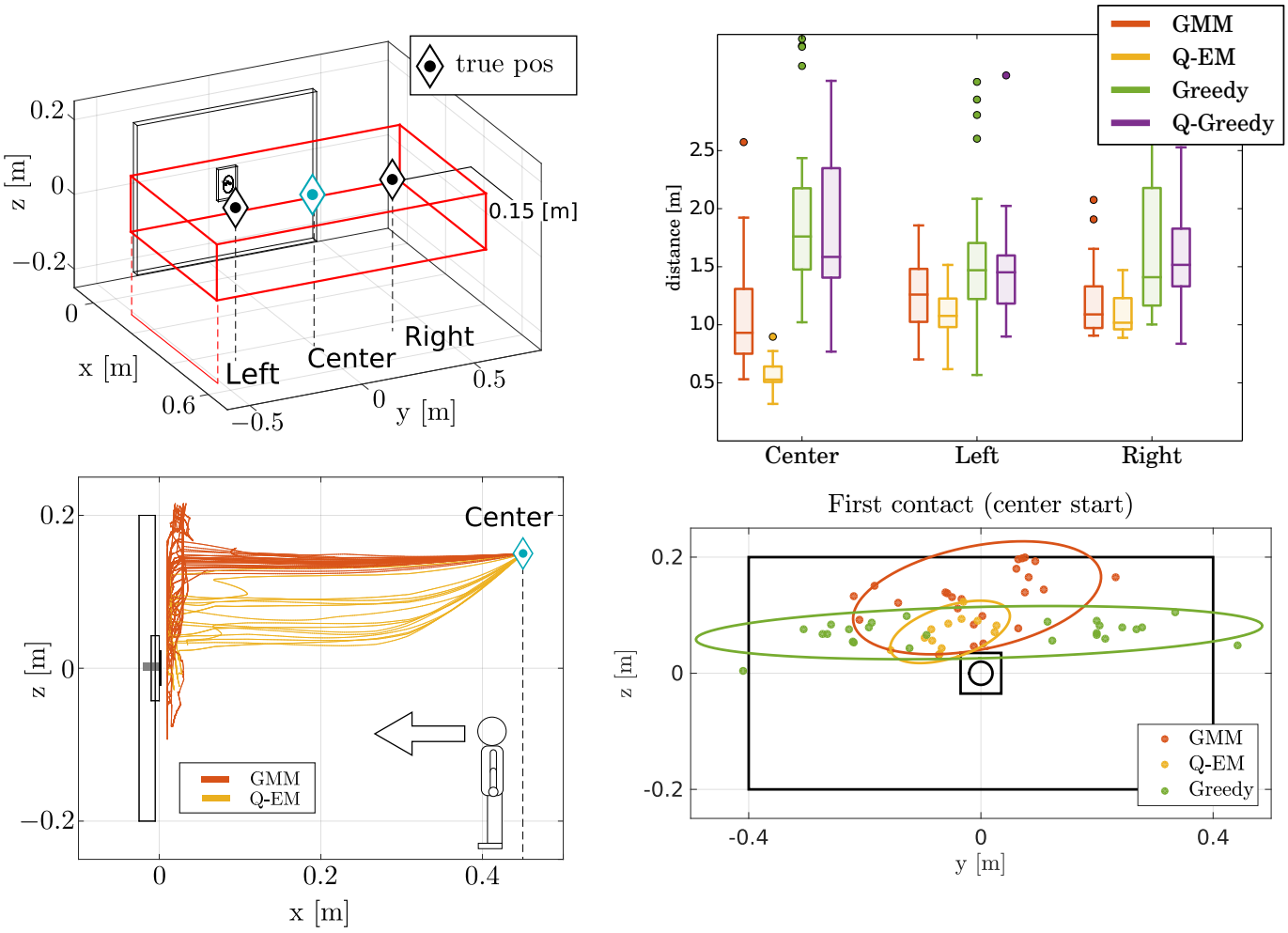
Figure 11 (*Bottom-left*) illustrates the value function of the belief state learned from the data of teacher # 5. The states with the highest value seem to create a path going from the socket towards the right edge of the wall. As before, to learn a GMM policy is learned from the raw data to produce a Q-EM policy in which the data points are weighted by the gradient of the value function. *Experiment 3 Middle-column* illustrates the resulting Marginalised Gaussian Mixture parameters for both the GMM and Q-EM policies where 25 rollouts are plotted of each policy starting at the *Center* initial condition also used in Experiment 1. It can be seen that the trajectories of the GMM policy have much greater variance in contrast to the Q-EM policy, resulting from an excess of variance in the 15 original demonstrations given by the teacher. Too much variance is not necessarily good, a random (uniform) policy in terms of generated trajectories will have the most variance and is as expected extremely inefficient in achieving a goal. 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 12, 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. The Q-Greedy policy was used in combination with random perturbations applied to the velocity to act as a simple exploration technique. A maximum of 150 searches were performed, which terminated once the socket was found. These demonstrations were used to learn a value function and GMM policy which we refer to as Q-Greedy. Figure 10 *Experiment 1-2 (bar plot)*, illustrates the statistical results of the Q-Greedy policy for Experiment 1 and 2 (purple bar chart), showing that there is no difference between the two policies. This exploration method is probably too simplistic to discover meaningful search patterns and we could probably devise better search strategies which would result in a better policy. However this study has shown that human behaviour already does have a usable trade-off between exploration and exploitation which can be used to learn a new policy through our Fitted Policy Iteration framework.

6.3. Generalisation

So far we have trained and evaluated our policies within the same environment. To test whether the GMM and Q-EM poli-

Experiment 1



Experiment 2

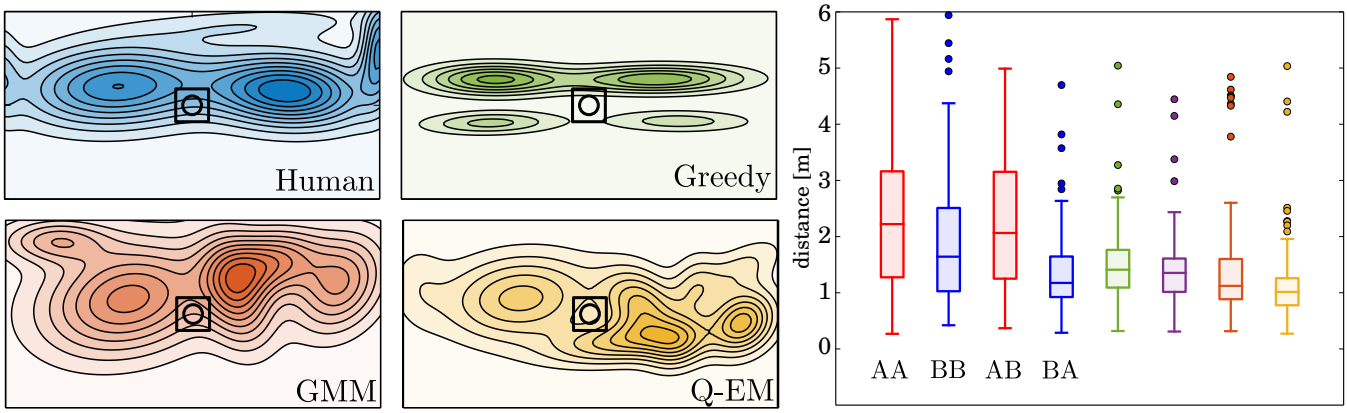


Figure 10: Two simulated search experiments. **Experiment 1:** *Top-left*: Three start positions are considered: *Left*, *Center* and *Right* in which the triangles depict true position of the end-effector. The red cube illustrates the extent of the uncertainty. *Bottom-left*: Trajectories of both the GMM (orange) and Q-EM (yellow) policies. For each start condition a total of 25 searches were performed for each search policy. *Bottom-right:* Distribution of first contact point giving the center initial starting condition. *Top-right*: Distribution of distance travelled until the socket's edge was localised. The Q-EM policy is always the best. **Experiment 2**: *Left*: Distribution of the visited regions during the search for the socket's edge. The Q-EM policy's distribution is more centred along the axis $z = 0$. *Right:* Time taken to find the socket, the search algorithms are better than the humans with the exception of group BA.

Experiment 3

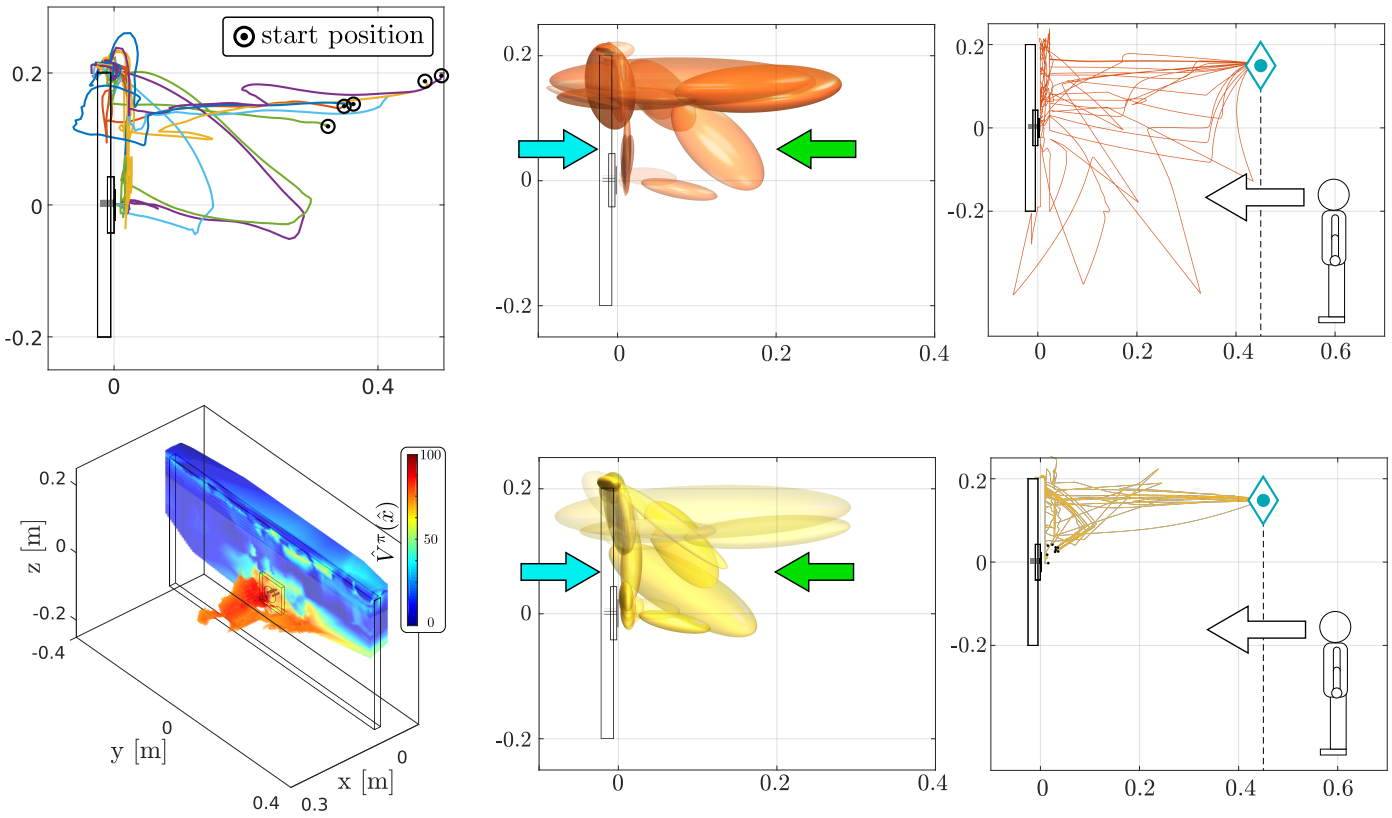


Figure 11: **Experiment 3** *Top-left*: Demonstrations of teacher # 5. *Bottom-left* Value function learned from the 15 demonstrations of teacher #5. The value of the most likely state is plotted. *Middle-column*: Most likely state parameters of the GMM and Q-EM learned from the demonstrations of teacher #5. *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 pattern, whilst the Q-EM policy gives many more consistent trajectories replicating to some extent the pattern of making a jump (no contact with the wall) from the top right corner to the socket's edge.

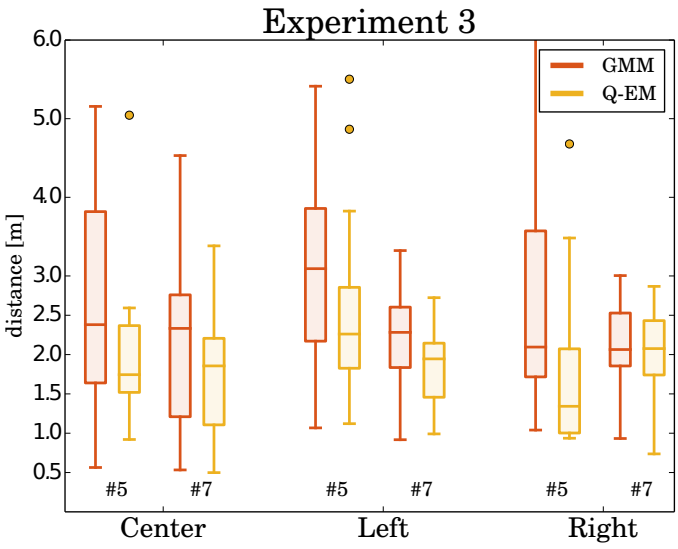


Figure 12: Distance taken to reach the goal for the GMM and Q-EM policies when trained with the worst two teachers. The initial starting conditions are as in Experiment 1. The Q-EM policy nearly always does much better than the GMM policy for both when trained with data from subject #5 or #7.

cies can generalise to a new setting the socket was moved to the upper right corner of the wall. The GMMs were trained in the frame of reference of the socket and when we translated the socket's location it also translated the policy.

The same initial conditions of Experiment 1 were used with an additional new configuration named *Fixed*, in which both the true and believed location are fixed, blue triangle and circle. Figure 13 *(Left)* 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 with both the GMM and Q-EM policies. From the statistical results shown in Figure 13 *(Right)* 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 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 higher is the likelihood that the search quality degrades. 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. We note that the policy will always be able to find the socket once it has localised itself. This can be seen from the vector field of the GMM policy when the uncertainty is low, see Figure 8 on page 8. In this case the policy is a sink function with a single point attractor.

6.4. Distance taken to connect the plug to the socket

This section evaluates the distance taken for the policies and humans to establish a connection, after the socket has been found. The distance is measured 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 human teachers' demonstrations. 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 14 for an illustration of the initial starting condition. This initial configuration was used to evaluate the search policies for the three different sockets, see Figure 1 on page 4 for an illustration of the sockets. The same initial configuration for the evaluation of the three sockets was kept in order to observe the generalisation properties of the policies. Note that only the training data from demonstrations acquired during the search with socket A were used. Socket B has a funnel which should make it easier to connect whilst socket C should be more difficult as it has no informative features on its surface.

For each of the sockets 25 searches were performed starting from the same initial condition. In Figure 14 *(Left)* 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 to directly establish a connection. In Figure 14 *(Right)* illustrates 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 vertically up and down. Only when the uncertainty has been reduced does the GMM policy 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 is not 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.

Figure 15 (a) illustrates the statistics of the distance taken to establish a connection for all three sockets. 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. All three search methods are vastly superior, when compared to the human's performance see Figure 15 (b-c).

The interesting point is that both the GMM and Q-EM algorithms perform better than the Greedy approach for socket

Experiment 4

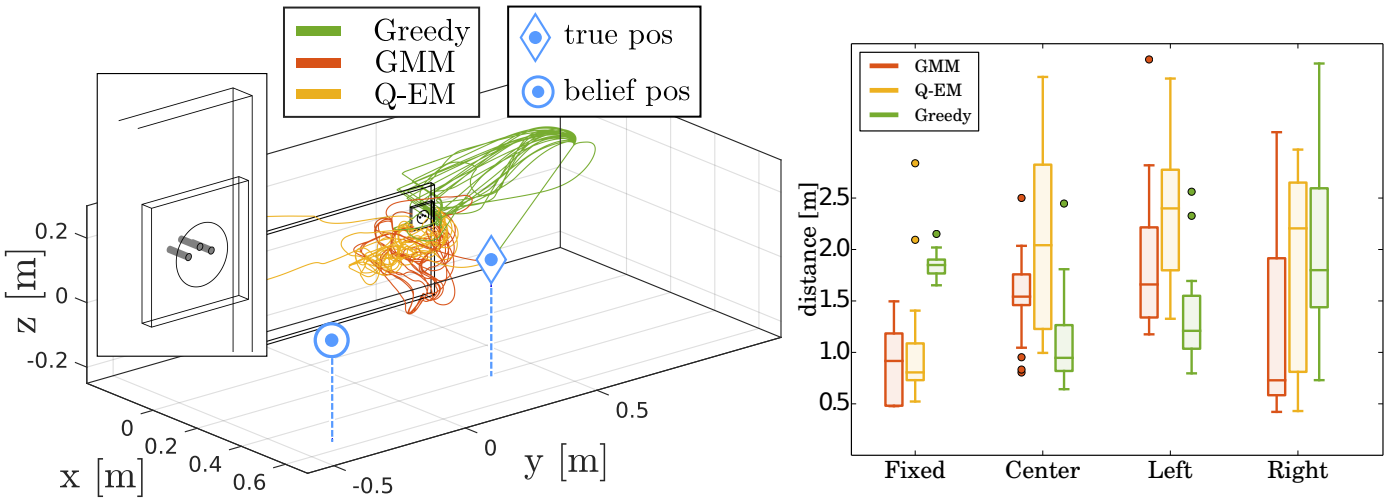


Figure 13: **Experiment 4** Evaluation of generalisation. The socket is located at the top right corner of the wall. We consider a *Fixed* starting location for both the true and believed locations (most likely state $\hat{x}_t$) of the end-effector. The red square depicted in Figure 10 is the extent of the initial uniform uncertainty. *Right:* Distance taken to reach the socket's edge for four initial starting conditions, left, centre and right of Experiment 1 and the fourth is the fixed condition as previously described. For the Fixed setup both the Q-EM and GMM significantly outperform the Greedy.

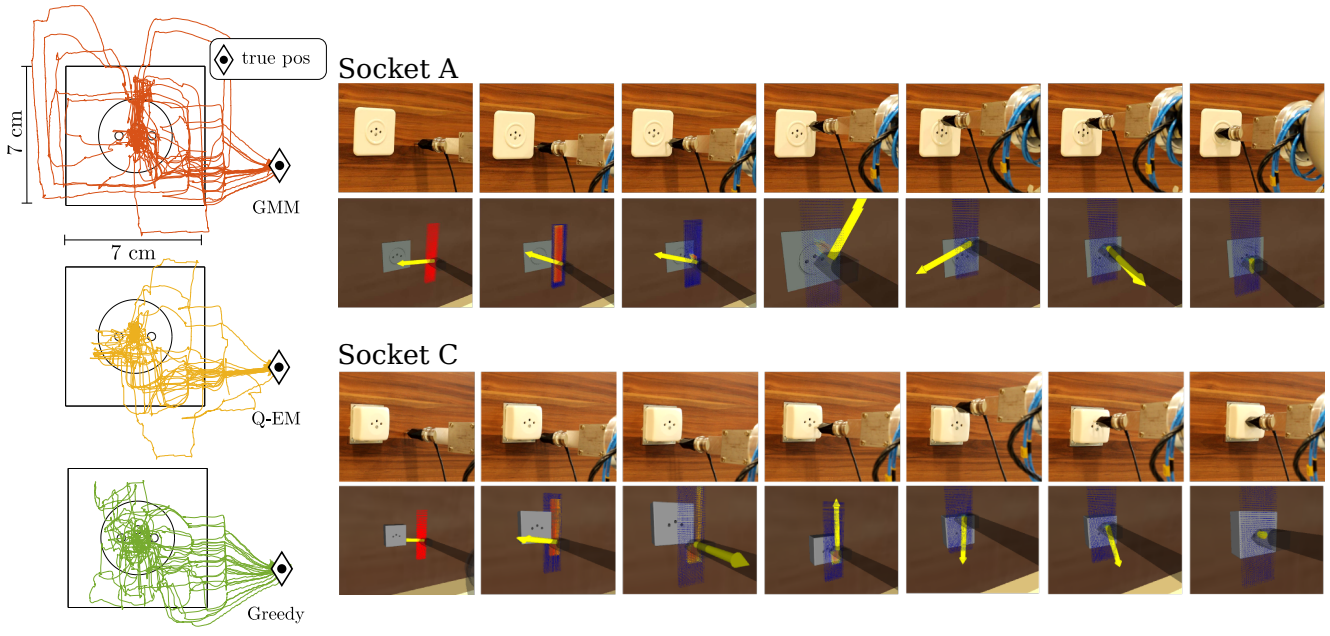


Figure 14: *Left*: 25 search trajectories for each of the three search policies for socket A. *Right* KUKA LWR4 equipped with a holder mounted with a ATI 6-axis force-torque sensor. *Socket A*: The robot's end-effector starts to the right of the socket. The second row shows screen captures taken of ROS 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. *Socket C*: Same initial condition as for socket A. 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 making a connection. See Video 4 and Video 5 for an illustration of the KUKA establishing a connection to socket B and C.

C. Socket C has no informative features on its surface and as a result myopic policies such as the Greedy policy 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. We found that the insertion policy $\pi_{\theta_2}(\omega|\phi)$ only helped with the insertion when the wall was artificially rotated such that a direct insertion was not possible. Otherwise the insertion was always successful 100% of the time. In other more industrial applications the insertion step is more problematic than when considering an electric socket, see [22, Chap. 5] for further details on the jamming dilemma.

The discrepancy between the humans performance and the 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' position belief. Also, 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, humans are far less precise than the KUKA which has sub-millimetre accuracy.

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 were learned from the data recorded during the human teachers' demonstrations. 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 weight training datapoints in the M-step update of Expectation-Maximisation (EM). The second policy, called GMM, was learned using the standard EM algorithm, and considered all training data points equally, following in the footsteps of our initial approach [10]. Both the Q-EM and GMM policies were trained with data solely from the human demonstrations of the search with socket A.

Four different aspects of the learned policies have been evaluated. Firstly, which of three policies, Q-EM, GMM and a Greedy policy, took the least distance to find the socket. Across three different experiments it was shown that the Q-EM algorithm always performs 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, the importance of the data provided by the human teachers was tested. The data from the two worst teachers was used to train an individual GMM and Q-EM policy for each of them. It was 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. A Q-EM policy was also learned from the data provided by a Greedy policy with explorative noise and no improvement was found. From these results we conclude that the exploration and exploitation aspects of the trajectories provided by the human teachers is necessary.

Thirdly, the two policies (GMM and Q-EM) were tested to see whether they were able to generalise to a different socket location. Under a specific condition, called *Fixed*, both policies were significantly better than the Greedy policy. However for the *Center* and *Left* initial conditions the Greedy policy performed better. When 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, all three policies were evaluted on the KUKA LWR robot and all performed better than the human teachers. For socket A 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 performed better than the Greedy, as socket C has no features on its surface, this being a disadvantage for a myopic policy.

We conclude that by simply adding a binary reward function in combination with data provided by human demonstrations, using fitted reinforcement learning, better policy can be learned without the need to perform 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.

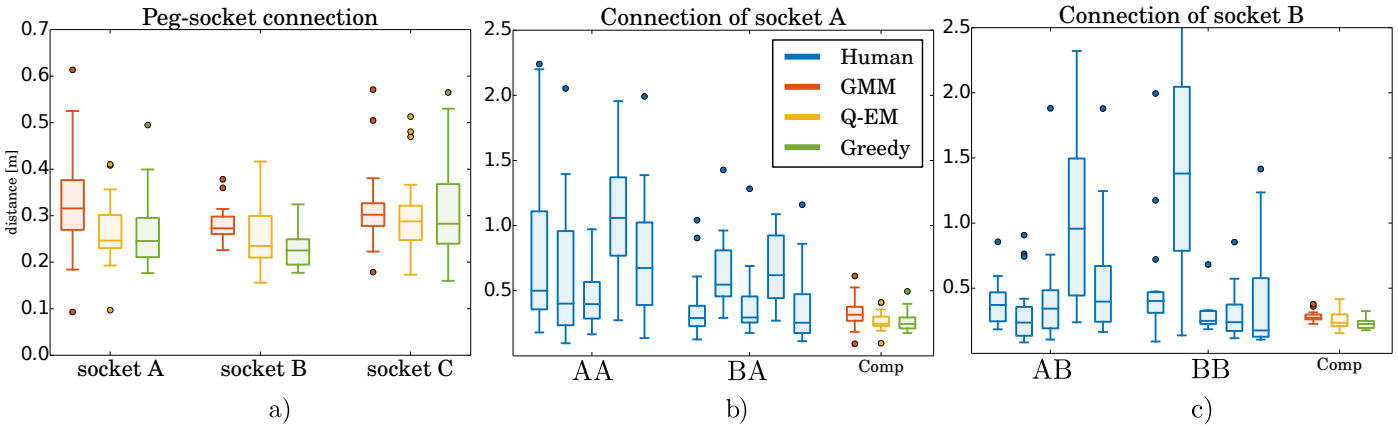


Figure 15: Distance taken to connect the plug to the socket (a) The Q-EM algorithm is the best for both socket A and C. For socket C, the Greedy algorithm does worse than the other two. This is because socket C has no informative features. (b) Group AA are the set of teachers who first started with socket A. They had no previous training on another socket beforehand. Group BA first gave demonstrations on Socket B before giving demonstrations on Socket A. Group BA is better than Group AA 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. (c) Both Groups AB and BB are similar in terms of the distance they took to insert the plug into the socket, the search policies on the other hand travel less to accomplish the task.

[1] Abu-Dakka, F., Nemec, B., Kramberger, A., Buch, A. G., Krüger, N., Ude, A., 2014. Solving peg-in-hole tasks by human demonstration and exception strategies. Industrial Robot 41 (6), 575–584.
URL http://dx.doi.org/10.1108/IR-07-2014-0363

[2] Agostini, A., Celaya, E., July 2010. Reinforcement learning with a gaussian mixture model. In: International Joint Conference on Neural Networks (IJCNN). pp. 1–8.
URL http://dx.doi.org/10.1109/IJCNN.2010.5596306

[3] Atkeson, C., Moore, A., Schaal, S., 1997. Locally weighted learning. Artificial Inteligence, 11–73.
URL http://dx.doi.org/10.1023/A:1006559212014

[4] Baxter, J., Bartlett, P., 2000. Reinforcement learning in pomdp's via direct gradient ascent. In: International Conference on Machine Learning. Vol. 17. Morgan Kaufmann, pp. 41–48.

[5] Bergman, N., Bergman, N., 1999. Recursive bayesian estimation: Navigation and tracking applications. thesis no 579. Tech. rep., Linköping University, Linköping Studies in Science and Technology. Doctoral dissertation.

[6] Bishop, C. M., 2006. Pattern Recognition and Machine Learning. Springer.

[7] Bou-Ammar, H., Voos, H., Ertel, W., Sept 2010. Controller design for quadrotor uavs using reinforcement learning. In: International Conference on Control Applications. pp. 2130–2135.
URL http://dx.doi.org/10.1109/CCA.2010.5611206

[8] Busoniu, L., Ernst, D., de Schutter, B., Babuska, R., April 2011. Approximate reinforcement learning: An overview. In: Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL). pp. 1–8.
URL http://dx.doi.org/10.1109/ADPRL.2011.5967353

[9] Calinon, S., D'halluin, F., Sauser, E., Caldwell, D., Billard, A., jun 2010. Learning and reproduction of gestures by imitation. IEEE Robotics Automation Magazine 17 (2), 44–54.
URL http://dx.doi.org/10.1109/MRA.2010.936947

[10] Chambrier, G. d., Billard, A., 2014. Learning search policies from humans in a partially observable context. Journal of Robotics and Biomimetics 1 (1), 1–16.

[11] Cheng, H., Chen, H., may 2014. Online parameter optimization in robotic force controlled assembly processes. In: International Conference on Robotics and Automation (ICRA). pp. 3465–3470.
URL http://dx.doi.org/10.1109/ICRA.2014.6907358

[12] Cheng, H., Chen, H., Hao, L., Li, W., May 2014. Robot learning based on partial observable markov decision process in unstructured environment. In: International Conference on Robotics and Automation (ICRA). pp. 4399–4404.
URL http://dx.doi.org/10.1109/ICRA.2014.6907500

[13] Deisenroth, M. P., Neumann, G., Peters, J., 2013. A survey on policy search for robotics. Foundations and Trends in Robotics 2 (1-2), 1–142.
URL http://dx.doi.org/10.1561/2300000021

[14] Du, Y., Hsu, D., Kurniawati, H., Lee, W., Ong, S., Png, S., 2010. A pomdp approach to robot motion planning under uncertainty. In: International Conference on Automated Planning and Scheduling, Workshop on Solving Real-World POMDP Problems.

[15] Ernst, D., Geurts, P., Wehenkel, L., April 2005. Tree-based batch mode reinforcement learning. Journal of Machine Learning Research 6, 503–556.

[16] Gordon, G., 1995. Stable function approximation in dynamic programming. In: International Conference on Machine Learning (ICML). Carnegie Mellon University.
URL "http://www.cs.cmu.edu/~ggordon/ml95-stable-dp.ps.gz"

[17] Grondman, I., Busoniu, L., Lopes, G., Babuska, R., Nov 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 42 (6), 1291–1307.
URL http://dx.doi.org/10.1109/TSMCC.2012.2218595

[18] Gullapalli, V., Barto, A., Grupen, R., may 1994. Learning admittance mappings for force-guided assembly. In: International Conference on Robotics and Automation (ICRA). pp. 2633–2638.
URL http://dx.doi.org/10.1109/ROBOT.1994.351117

[19] Hausknecht, M., Stone, P., 2015. Deep recurrent q-learning for partially observable mdps. Association for the Advancement of Artificial Intelligence (AAAI).
URL https://www.aaai.org/ocs/index.php/FSS/FSS15/paper/view/11673

[20] Hauskrecht, M., Aug 2000. Value-function approximations for partially observable markov decision processes. Journal of Artificial Intelligence Research 13 (1), 33–94.
URL http://dl.acm.org/citation.cfm?id=1622262.1622264

[21] Kalakrishnan, M., Righetti, L., Pastor, P., Schaal, S., Sept 2011. Learning force control policies for compliant manipulation. In: International Conference on Intelligent Robots and Systems (ICRA). pp. 4639–4644.
URL http://dx.doi.org/10.1109/IROS.2011.6095096

[22] Kronander, K., 2015. Control and learning of compliant manipulation skills.

[23] Lange, S., Riedmiller, M., July 2010. Deep auto-encoder neural networks in reinforcement learning. In: International Joint Conference on Neural Networks (IJCNN). pp. 1–8.

[24] Lauri, M., Ritala, R., 2016. Planning for robotic exploration based on forward simulation. Robotics and Autonomous Systems.

[25] Meeussen, W., Wise, M., Glaser, S., Chitta, S., et. al, May 2010. Autonomous door opening and plugging in with a personal robot. In: Inter-

national Conference on Robotics and Automation (ICRA). pp. 729–736.
URL http://dx.doi.org/10.1109/ROBOT.2010.5509556

[26] Mnih, V., 02 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.
URL http://dx.doi.org/10.1038/nature14236

[27] Nemec, B., Abu-Dakka, F., Ridge, B., et. al, Nov 2013. Transfer of assembly operations to new workpiece poses by adaptation to the desired force profile. In: International Conference on Advanced Robotics (ICAR). pp. 1–7.
URL http://dx.doi.org/10.1109/ICAR.2013.6766568

[28] Neumann, G., Peters, J., Jun 2009. Fitted q-iteration by advantage weighted regression. In: Advances in Neural Information Processing Systems (NIPS). Vol. 21. pp. 1177–1184.

[29] Neumann, G., Peters, J. R., 2009. Fitted q-iteration by advantage weighted regression. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (Eds.), Advances in Neural Information Processing Systems (NIPS). Vol. 21. Curran Associates, Inc., pp. 1177–1184.

[30] Ormoneit, D., Glynn, P., Oct 2002. Kernel-based reinforcement learning in average-cost problems. Transactions on Automatic Control 47 (10), 1624–1636.
URL http://dx.doi.org/10.1109/TAC.2002.803530

[31] Peters, J., Schaal, S., 2008. Natural actor-critic. European Symposium on Artificial Neural Networks 71 (7-9), 1180–1190.
URL http://dx.doi.org/10.1016/j.neucom.2007.11.026

[32] Peters, J., Schaal, S., mar 2008. Natural actor-critic. Neurocomputing 71 (7-9), 1180–1190.
URL http://dx.doi.org/10.1016/j.neucom.2007.11.026

[33] Pineau, J., Gordon, G., Thrun, S., Aug 2003. Point-based value iteration: an anytime algorithm for pomdps. In: International Joint Conference on Artificial Intelligence (IJCAI). pp. 1025–1030.
URL http://dl.acm.org/citation.cfm?id=1630659.1630806

[34] Porta, J., Vlassis, N., Spaan, M., Poupart, P., Dec 2006. Point-based value iteration for continuous pomdps. J. Mach. Learn. Res. 7, 2329–2367.
URL http://dl.acm.org/citation.cfm?id=1248547.1248630

[35] Riedmiller, M., 2005. Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method. Vol. 16. Springer Berlin Heidelberg, pp. 317–328.
URL http://dx.doi.org/10.1007/11564096_32

[36] Ross, S., Pineau, J., Paquet, S., Chaib-draa, B., Jul 2008. Online planning algorithms for pomdps. Journal Artifcial Intelligence Research 32 (1), 663–704.
URL http://dl.acm.org/citation.cfm?id=1622673.1622690

[37] Schaal, S., Peters, J., Nakanishi, J., Ijspeert, A., 2004. Learning movement primitives. In: 11th International Symposium on Robotics Research (ISRR).
URL http://dx.doi.org/10.1007/11008941_60

[38] Siddharth, C., Branicky, M., 2001. Search strategies for peg-in-hole assemblies with position uncertainty. In: International Conference on Intelligent Robots and Systems (ICRA). Vol. 3. pp. 1465–1470.
URL http://dx.doi.org/10.1109/IROS.2001.977187

[39] Smallwood, R., Sondik, E., Oct 1973. The optimal control of partially observable markov processes over a finite horizon. Journal of Operational Research 21 (5), 1071–1088.
URL http://dx.doi.org/10.1287/opre.21.5.1071

[40] Sung, H. G., 2004. Gaussian mixture regression and classification. Ph.D. thesis, Rice University.

[41] Sutton, R. S., Barto, A., 1998. Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA.

[42] Thrun, S., Burgard, W., Fox, D., 2005. Probabilistic Robotics. The MIT Press.

[43] Veiga, T., Spaan, M., Lima, P., 2014. Point-based POMDP solving with factored value function approximation. In: Conference on Artificial Intelligence (AAAI). Vol. 28. pp. 2512–2518.

[44] Vien, N., Toussaint, M., Sept 2015. Pomdp manipulation via trajectory optimization. In: International Conference on Intelligent Robots and Systems (IROS). pp. 242–249.
URL http://dx.doi.org/10.1109/IROS.2015.7353381

[45] Wiering, M., van Otterio, M., 2012. Reinforcement Learning State-of-the-Art. Springer-Verlag Berlin Heidelberg.

[46] Yang, Y., Lin, L., Song, Y., Nemec, B., Ude, A., Buch, A., Kruger, N., Savarimuthu, T., 2014. Fast programming of peg-in-hole actions by human demonstration. pp. 990–995.
URL http://dx.doi.org/10.1109/ICMC.2014.7231702