Efficient Robot Task Learning and Transfer via Informed Search in Movement Parameter Space

Nemanja Rakicevic and Petar Kormushev
Robot Intelligence Lab
Imperial College London
{n.rakicevic, p.kormushev}@imperial.ac.uk

Abstract

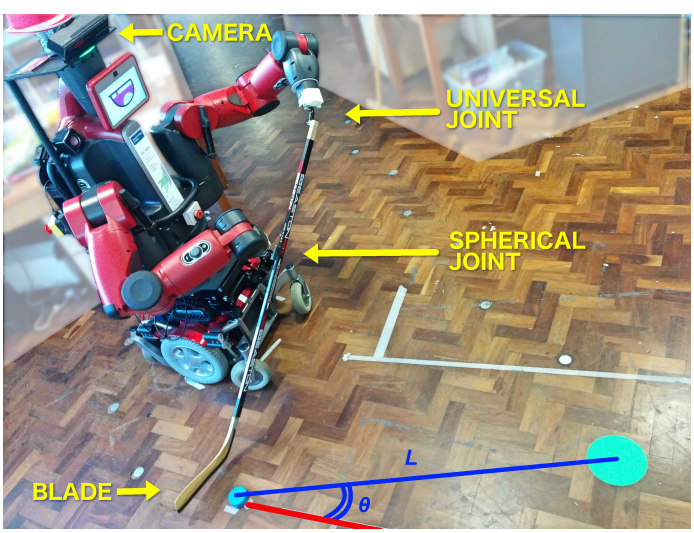
Learning complex physical tasks via trial-and-error is still challenging for high-degree-of-freedom robots. The greatest challenge is optimally selecting the next trials to evaluate while improving sample efficiency. We propose a novel learning framework consisting of the *task modelling* and *exploration* components. The task model maps the parameter space defining a trial, to the task outcome space. The exploration model enables efficient search in the trial parameter space to generate the subsequent most informative trials, by simultaneously exploiting all the information gained from previous trials and reducing the task model's overall uncertainty. We validate our framework on a challenging bimanual-robot puck-passing task. Results show that the robot successfully acquires the necessary skills after only 100 trials without any prior information about the task or target positions. Our approach also enables efficient skill transfer to new environments which was validated experimentally.

1 Introduction

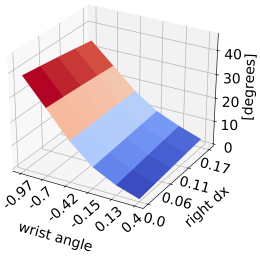
The motivation for this work comes from the approach humans take when learning complex tasks such as acquiring new skills, using new tools or learning sports. Most of their learning process is centred around trials and errors [19]. Even though there may be demonstrations present (explanatory videos, user manuals or a human instructor) several trials are necessary to acquire adequate motoric capabilities, e.g. feel the weight of the utensil, practice hand-eye coordination, etc. For robot learning, each trial can be uniquely defined by a set of movement parameters [13] which means that performing a trial is equivalent to selecting a point in the movement parameter space and evaluating it.

In this paper, we introduce a novel online active learning approach consisting of the informed search in the movement parameter space for generating trials during the training phase. Our aim is to develop a method which is sample efficient and does not make random or exhaustive exploration, intended for systems where trial execution is expensive. Moreover, during the training phase, the agent does not have any prior information about the task or the environment, in order to minimise domain knowledge. To this end, we propose a learning framework consisting of a *task modelling* and an *exploration* component. The task model is implemented as a Gaussian Process (GP) Regression (GPR) [22] function that maps the movement parameters as inputs to the task outcomes as the desired outputs. The exploration model enables efficient search in the movement parameter space to generate a parameter vector that encodes a subsequent trial which is most informative for the task model. This is done by exploiting the scarce trial data via probabilistic modeling and by taking into account the uncertainty inherent to the task model. The exploration model is implemented as a *selection function* defining a probability distribution over the movement parameter space, from which parameter vectors are sampled. The proposed approach performs coupled learning of the task and exploration models

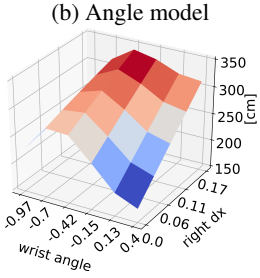
Workshop on Acting and Interacting in the Real World: Challenges in Robot Learning, 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.



(a) Robot setup



(b) Angle model



(c) Distance model

Figure 1: (a) Experimental setup of the ice hockey puck passing task: θ and L are the polar coordinates estimated using the head-mounted Kinect camera. The image coordinates are transferred to the floor coordinates using perspective transformation. The red line in the bottom is parallel to the heading direction and is used as the zero-angle reference. The obtained target coordinates are used by the framework during testing to produce the appropriate swing action needed to pass the puck to the target. On the right are the GPR task models learned from the successful trials; (b) Learned model for the angle (c) Distance model. To visualise the 6-dimensional parameter space we fix the remaining parameters and show functions w.r.t. the wrist angle and right hand x-axis displacement.

by trial and error. During the training phase, the algorithm iteratively finds points in the parameter space to produce trials which are the most informative about the task model and which lower the task model's prediction uncertainty. Since the exploration model maintains information about the useful trials, they can be directly reproduced (transferred) in different environmental settings. As a consequence, new task models can be learned from scratch with significantly less trial evaluations.

The framework is executed directly on a physical robot. For evaluation we have selected the ice bimanual hockey puck-passing task, as shown in Fig. 1a. During the testing phase, the robot is presented with a target position and expected to automatically generate an appropriate swing movement action, based on the learned task model, to hit the puck so that it lands on the target. We selected this particular task as it is interesting for its complexity: (i) it requires dual-arm coordination, (ii) there is a non-trivial extension of the robot's kinematic model via the ice hockey stick and (iii) the surface friction and stick-surface contact models are quite difficult to model. The proposed approach requires very little prior knowledge about the system: no previous task knowledge (strategies, desired movements, etc.), prior kinematic (stick and joint constraints) nor environment (surface friction, contact forces, etc.) models are provided. No demonstrations or expert human supervision are necessary. Only the number of input parameters is given (which can be anything from basic low-level control inputs, to more abstract notions such as motion primitives) and their ranges, without any contextual information regarding their influence or importance.

To summarise, the main contributions of this paper are:

- The probabilistic framework for informed and sample-efficient search of feasible movement parameter vectors defining the robot motion for the next trial evaluation. The trials are generated in such a way that their outcomes provide diverse and informative data points in order to construct accurate, invertible and generalisable task models.
- As a consequence of the proposed framework setup, efficient task transfer to new environments is possible, as shown experimentally. The robot successfully learns the task models for the new environments in significantly less trials.

2 Related work

The concept of efficient exploration strategies is crucial in Reinforcement Learning (RL), but also in supervised learning where it can improve sample selection and lead to more sample-efficient models. Several authors argue the importance of exploration and benefits of moving it directly to the parameter space, as opposed to e.g. action space in RL [20, 24], as it can reduce the variance caused by noisy trajectories, and generally avoids converging prematurely to suboptimal solutions. Evolutionary Strategy-based methods [11, 12, 25, 31] introduce noise in the parameter space to guide explorations, but they act as a black-box optimiser and lead to poor sample-efficiency. Probabilistic approaches such as Bayesian Optimisation [5, 7, 18, 30] use the prediction entropy to guide the exploration in a more efficient way. Such methods usually rely on the GP framework due to its generalisability and intrinsic uncertainty measure derived for each prediction, which can further contribute to the optimisation process, as utilised in PILCO [8] as well.

The methods above use entropy to guide the exploration and aid the optimisation process, which by definition requires a cost function. In our problem setting we exclude such a performance measurement function as it is unfeasible to evaluate the model at each fitting step. Methods relying on techniques like Motor Babbling [9, 16] can learn the robot's forward model by iteratively performing random motor commands and recording their outcomes. However, these methods are usually data-inefficient due to random exploration. Therefore, we concentrate on the Active Learning (AL) [4, 28] paradigm as it has proved useful in robotic applications [1, 10, 29] where it helps relieve the sample complexity — one of the main limitations imposed by hardware for robotic experiments. An interesting active learning exploration approach applied in robotics is the Estimation-Exploration Algorithm (EEA) [2]. EEA performs system identification, by selecting trials whose outcome causes most disagreement among the candidate system models. The probabilistic modelling approaches, particularly GPs, are exploited within AL [14, 17, 23, 27] to provide uncertainty-based exploration. However, most of the AL sample query strategies which rely on prediction uncertainty, do not take into account the historical evolution of the models and the data samples' relations explicitly, rather implicitly through the model and its prediction estimates. This is especially important in robotics where physical constraints play a crucial role [26].

An example of learning robotic tasks with complex kinematics, through exploration in the low-level control space is presented in [16]. Elements which are not part of the original kinematic chain are incorporated by adding links and leverage points to skew the mapping of motor torques to the end-effector pose, and the robot successfully adjusts to these modifications. No explicit model of the robot's kinematics is provided, but such approach would have difficulties when scaled. Relevant example a of single-arm robot learning to play hockey using RL is presented in [6] where the robot learns to send the puck into desired reward zones and gets feedback after each trial. However, kinaesthetic teaching is required to extract the shape of the movement. Also in [15] learning from demonstrations is used to teach the robot to play minigolf. A recent approach [3] combines model-free and model-based RL updates to learn the optimal policy that shoots the puck to one of the three possible goals. The tracked puck-to-goal distance is used within the cost function, so this heuristic provides some form of a reward shaping. Our approach differs from the above two, because during the training phase no information about the goal nor the environment is provided.

3 Problem formulation and movement parameterisation

We consider the problem of autonomously learning the ice-hockey puck-passing task with a bimanual Baxter robot with two 7-DOF-arms, as shown in Fig. 1a. The puck-passing motion that the robot performs consists of a swing motion, making the contact with the puck and transferring the necessary impulse to move the puck to a certain location. The robot learns this through trial and error without any target positions provided during the training phase, just by exploring different swing movements in an informed way and recording their outcomes. Each trial consists of a potential swing movement which is encoded using a set of movement parameters. We propose a set of 6 movement parameters which are empirically chosen and sufficient to describe a swing. These parameters represent the amount of displacement relative to the initial arm configurations. The displacements considered are along the x and y axes of the robot coordinate frame (task space) for the left (l_x, l_y) and right (r_x, r_y) hands, the joint rotation angle of the left wrist (w), and the overall speed coefficient (s) which defines how fast the entire swing movement is executed. In this way the swing movement is parameterised

and can be executed as one action. In the proposed setup, the parameters take discrete values from a predefined fixed set, equally spaced within the robot's workspace limits. Although the inverse kinematics model is used via the position controller, our framework does not exploit this information. The generated swing movement can either be feasible or not for the robot to execute. If feasible, the generated swing movement can potentially hit the puck. The hit slides the puck from its fixed starting position to some final position, which is encoded via polar coordinates θ and L as shown in Fig. 1a. Such a trial is considered successful and contributes to the task models. In the testing phase, the robot is presented with target positions that the puck needs to achieve. The target is visually perceived as a green circle which is placed on the floor by the user (Fig. 1a). Having received the target coordinates (θ_d and L_d), the robot applies a proper swing action that passes the puck to the target. Therefore, a method that constructs a reliable, generic and invertible task model is needed.

4 Proposed approach

The proposed approach consists of two coupled components, both of which are updated based on the previous experience, i.e. previous trials, the task model and exploration model components. The pseudocode of the proposed algorithm is presented in the Appendix.

4.1 Task model component

The task model component uses the information from scarce successful trials: the movement parameter values, as inputs, and the puck's final position, as outputs, and creates a mapping between them. This component creates two independent task models for each of the puck's polar coordinates, angle and distance, as depicted in Figures 1b and 1c, respectively. To this end, we use GPR as it generalises well with limited function evaluations, which in our case are the successful trials executed on the robot. Let us define a point in the movement parameter space $\boldsymbol{x} \in \mathbb{R}^D$, with D being the parameter vector dimensionality. The main assumption is that for any finite set of N points $\boldsymbol{X} = \{\boldsymbol{x_i}\}_{i=1}^N$, the corresponding function evaluations at these points can be considered as another set of random variables $\mathcal{F} = \{f_{\boldsymbol{x_i}}\}_{i=1}^N$, whose joint distribution is a multivariate Gaussian:

$$f\boldsymbol{x_i} \sim \mathcal{N}(\mu(\boldsymbol{x_i}), K(\boldsymbol{x_i}, \boldsymbol{x_i'}))$$

where $\mu(\boldsymbol{x_i})$ is the prior mean function and $K(\boldsymbol{x_i}, \boldsymbol{x_i'})$ is the kernel function for some pair of parameter vectors $\boldsymbol{x_i}, \boldsymbol{x_i'}$. When applied to all the pairs from $\boldsymbol{X}$ the kernel produces the matrix of covariances $\boldsymbol{K}$. Having a joint probability of the function variables, it is possible to get the conditional probability of some parameter vector's evaluation $f_{\boldsymbol{x_i^*}}$ given the others, and this is how we derive the posterior based on observations from the trials. In our case, $\boldsymbol{X}$ is the set of movement parameter vectors which led to successful trials during the training phase. Set $\boldsymbol{X^*}$ contains all the possible parameter combinations, since we need to perform inference to obtain the task models. We define the extended joint probability as below, and use matrix algebra to deduce the posterior:

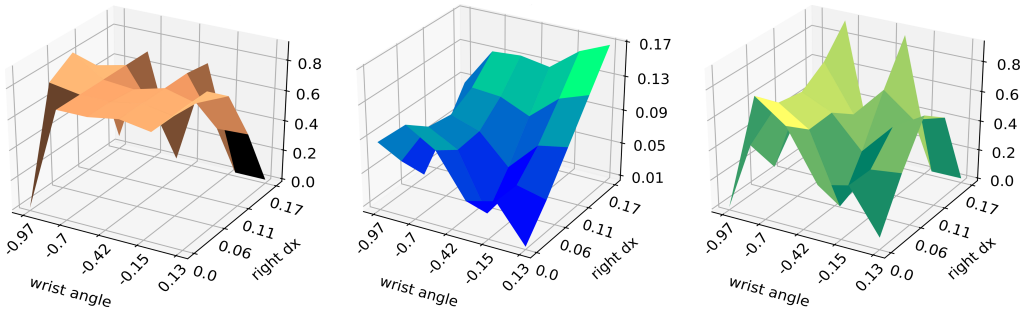
$$\begin{bmatrix} f_{\boldsymbol{X}} \\ f_{\boldsymbol{X^*}} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \boldsymbol{K} & \boldsymbol{K_*^T} \\ \boldsymbol{K_*} & \boldsymbol{K_{**}} \end{bmatrix}\right)$$

$$p(f_{\boldsymbol{X^*}} | f_{\boldsymbol{X}}, \boldsymbol{X^*}, \boldsymbol{X}) \sim \mathcal{N}\left(\boldsymbol{K_* K^{-1}} f_{\boldsymbol{x}}, \boldsymbol{K_{**}} - \boldsymbol{K_* K^{-1} K_*^T}\right)$$

We assume a mean of 0 for our prior. Similarly to $\boldsymbol{K}$, $\boldsymbol{K_{**}}$ is the matrix of covariances for all the pairs from the set $\boldsymbol{X^*}$, and $\boldsymbol{K_*}$ gives us the similarity of the observed parameter vectors $\boldsymbol{X}$ to each point in the parameter space $\boldsymbol{X^*}$. Within the kernel definition we also consider zero mean Gaussian noise, $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, to account for both modelling and measurement inaccuracies. The kernel function used is the rational quadratic (1), as it showed the best performance.

$$\boldsymbol{K}_{RQ}(\boldsymbol{x}, \boldsymbol{x'}) = \sigma_f^2 \left(1 + \frac{\boldsymbol{d}^2}{2\alpha\sigma_l^2}\right)^{-\alpha} \tag{1}$$

The distance measure $\boldsymbol{d}$ is defined as the Euclidean distance between the points in the parameter space $\boldsymbol{d}(\boldsymbol{x}, \boldsymbol{x'}) = \|\boldsymbol{x} - \boldsymbol{x'}\| = \sqrt{\sum_{j=1}^{D}(x_j - x_j')^2}$. From the similarity measure given by the kernel we get that the points, which are far away from each other, will have a higher variance associated with their prediction. The coefficients σ_f^2 and σ_l^2 are the variance and the lengthscale of the kernel, respectively, with values set as $\sigma_f^2 = 1$, $\sigma_l^2 = 1$ and $\alpha = D/2$. One of the advantages of GPR is that



(a) Penalisation function (b) Model uncertainty (c) Selection function

Figure 2: Components used to generate movement parameter vectors that define a trial: (a) penalisation function which inhibits failed trials (b) task model uncertainty (c) selection function which combines the previous two to get the distribution from which the next parameter vector is sampled.

for every point for which we estimate the posterior distribution, we know its mean and variance. The means are interpreted as the current models' predictions, and the variance as the model's confidence about these predictions. Therefore, regions of the parameter space which are farther away from the training points, have a higher variance and thus the uncertainty about their predictions is higher. After each new successful trial, we re-estimate the posteriors over the whole movement parameter space, in order to update both the angle and distance task models as well as the uncertainty. This exact inference is memory demanding but executes in seconds on a workstation with a GTX 1070 GPU.

4.2 Exploration model component

The Exploration Model component exploits all the past (both successful and failed) trial information, in order to obtain the *selection function* that guides the movement parameter search and selection process. The elements contributing to the selection function are the information about the unsuccessful trials, expressed through a penalisation function, and the GPR model uncertainty. These functions are represented as improper density functions (IDF) since their values for each point in the parameter space are in the range $[0, 1]$ but they do not sum to 1. An example of these IDFs is visualised in Fig. 2.

Penalisation IDF (PIDF) probabilistically penalises regions around the points in the movement parameter space that have led to failed trials. This inhibits repetition and reduces the probability of selecting parameter vectors which lead to failed trials. A trial is classified as failed in four cases:

(1) The inverse kinematic solution cannot be found for the generated displacements.
(2) The displacements would produce physical damage to the robot (self collision, ignoring the stick constraint or hitting itself with the stick).
(3) The mechanical fuse breaks off due to an excessive force applied.
(4) The swing movement does not make contact with the puck (no puck movement).

The PIDF is implemented as a mixture of inverted D-dimensional Gaussians (2), which are added to a uniform prior distribution $\mathbf{\Pi_0}$. The uniform prior ensures that initially all the movement actions have an equal probability of being selected. Each of the K failed trials is represented by a Gaussian with a mean $\boldsymbol{\mu_k^P}$ coinciding with the parameter vector associated with this trial. The covariance matrix $\mathbf{\Sigma_k^P}$ is a diagonal matrix, calculated based on how often does each of the D parameters take repeated values, considering all the previously failed trials. This is implemented by using a counter for each parameter. In this way, the Gaussians have a smaller variance along the dimensions corresponding to the parameters that have frequently repeated values, thus applying more intense penalisation and forcing them to change from these 'stuck' values. For the parameters with evenly distributed values, the Gaussian will be more spread out. This procedure inhibits the selection of parameter values which are likely to contribute to failed trials, and stimulates selecting others. Conversely, the parameter vector leading to a successful trial is stimulated with a low positive value Gaussian with a high variance which allows promotion of the nearby region of the space.

$$p_p(\boldsymbol{x^*}) = \mathbf{\Pi_0} + \sum_{k=1}^{K} \phi_k \mathcal{N}(\boldsymbol{\mu_k}, \mathbf{\Sigma_k}), \begin{cases} \phi_k = -1, & \text{failed trial} \\ \phi_k = +1, & \text{successful trial} \end{cases} \tag{2}$$

Model uncertainty IDF (UIDF) is intrinsic to the GPR model (3) and it is used to encourage the exploration of the parameter space regions expected to yield most information about the underlying task models. The UIDF is updated for both failed and successful trials as the exploration does not depend on the actual model values.

$$p_u(\boldsymbol{x}^*) = \boldsymbol{K}_{**} - \boldsymbol{K}_* \boldsymbol{K}^{-1} \boldsymbol{K}_*^T \tag{3}$$

Selection IDF (SIDF) combines the information provided by the PIDF, which can be interpreted as the prior over possible movements, and the UIDF as the likelihood of improving the task models. Through the product of PIDF and UIDF, we derive SIDF (4), our query function, as the posterior IDF from which the optimal parameter vector for the next trial is sampled.

$$p_{sel}(\boldsymbol{x}^*) = p_p(\boldsymbol{x}^*) p_u(\boldsymbol{x}^*) \tag{4}$$

4.3 Learned task model evaluation

In order to evaluate the performance during the testing phase, it is necessary for the angle $\theta(\boldsymbol{x})$ and distance $L(\boldsymbol{x})$ models to be invertable. Given the target coordinates, a single appropriate movement parameter vector defining the swing action that passes the puck to the target needs to be generated. It is difficult to generate exactly one unique parameter vector $\hat{\boldsymbol{x}}$ which precisely realises both the desired coordinate values θ_d and L_d, so the one which approximates them both best and is feasible for robot execution is selected. This is achieved by taking the parameter vector which minimises the pairwise squared distance of the coordinate pair within the combined model parameter space, as in (5). Additional constraint on the parameter vector is that its corresponding PIDF value has to be below a certain threshold ϵ to avoid penalised movements. In our numerical approach, this is done iteratively over the whole parameter space, and takes a couple of milliseconds to run. Alternatively, this could be achieved using any standard optimisation algorithm. More importance can be given to a specific coordinate by adjusting the weighing factor δ.

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \left(\sqrt{(1-\delta)(\theta(\boldsymbol{x}) - \theta_d)^2 + \delta(L(\boldsymbol{x}) - L_d)^2} + (p_p(\boldsymbol{x}) - \epsilon) \right) \tag{5}$$

4.4 Task transfer

After the task model is learned, if we were to repeat the approach (with the fixed seed) in a different environment, we would still get the same trials. However, only the successful trials contribute to forming the task models, as they actually move the puck, while the failed trials are inherent to the robot's physical structure. Therefore, we can decouple the failed and successful trials, and execute only the latter in the new environment in order to retrain the task models. This significantly reduces the amount of trials needed to learn the skill in a new environment, because usually while exploring the parameter space the majority of the trials executed are failed.

5 Experimental setup and training

The proposed approach is evaluated with a Baxter robot using a standard right-handed ice hockey stick and puck [1]. To enable the robot to use this stick, we have equipped its end-effectors with custom passive joints for attaching the stick. A universal joint is mounted on the left hand, while the spherical joint is installed on the right (refer to Fig. 1a). This configuration inhibits the undesired idle roll rotation around the longitudinal stick axis, while allowing good blade-orientation control. The connection points on the stick are fixed, restricting the hands from sliding along it. This imposes kinematic constraints on the movement such that the relative displacement of the two hands along either axis cannot be greater than the distance between the fixture points along the stick. Due to the right-handed design of the ice hockey stick, the initial position of the puck is shifted to the right side of the robot and placed approximately $20\,\mathrm{cm}$ in front of the blade. For monitoring the movement effect on the puck, the head-mounted Kinect camera pointing downwards at a 45 degree angle is used. A simple object-tracking algorithm is applied to the rectified RGB camera image in order to extract the position of the puck and the target. For calculating the polar coordinates of the puck, the mapping from pixel coordinates to the floor coordinates w.r.t. the robot is done by applying the perspective

[1] The video of the experiments is available at `https://youtu.be/S9_PHsl0W7g`

transformation obtained via homography. All elements are interconnected using ROS [21]. The initial configuration of the robot arms and the ranges of the movement parameter values are assigned empirically. The approach could be extended to autonomously detect the limits for the parameters, but this could lead to physical damage. This implicitly reduces the number of learning trials, especially the failed ones. Discretisation resolution of the parameter values is due to the numerical approach to obtaining the GPR models, and higher resolution and parameters dimensions would require more memory resources. The parameter value sets assigned are [m]: $l_x = \{-0.3, -0.2, -0.1, 0, 0.1\}$, $l_y = \{-0.1, -0.05, 0, 0.05, 0.1\}$, $r_x = \{0, 0.042, 0.085, 0.128, 0.17\}$, $r_y = \{-0.1, 0.05, 0.2, 0.35, 0.5\}$, $w = \{-0.97, -0.696, -0.422, -0.148, 0.126, 0.4\}$ and $s = \{0.5, 0.625, 0.75, 0.875, 1.0\}$. This produces a parameter space of size $6 \times 5^5 = 18750$. The GPR generalises well despite the crude discretisation. The parameter values are considered normalized as they are in the range $[-1, 1]$.

The training phase consisted of 100 trials of which 24 were successful and contributed to the task models. The rest of the failed trials did not contribute to the task model explicitly, rather implicitly, through the exploration component. The stopping criterion is when the model's average uncertainty drops below 10% and the last 5 updates do not lead to more than 0.5% improvement each. Further uncertainty reduction would not make sense as it depends on the inherent task uncertainty which is hard to quantify. This task uncertainty is affected by the system's hardware repeatability and noise in the motion outcome amongst others. The overall training time including resetting is approximately 45 min. Figures 1b and 1c show the angle and distance models learned based on the 24 successful trials. For visualisation purposes we slice the model and display it along two of the six dimensions. We visualise r_x and w, while the remaining parameters are fixed with values: $l_x = -0.3$, $l_y = 0.1$, $r_y = 0.35$ and $s = 1.0$, which is equivalent to a backward motion of the left hand and a full speed swing. The angle model shows how for this particular swing configuration, the wrist rotation angle greatly affects the final angle of the puck. This is in line with how professional hockey players manipulate the puck. Conversely, the right hand displacement along the robot's x-axis does not contribute as much. The distance model shows more complex dependencies, where the right hand displacement has a high positive correlation with the final puck distance, for positive wrist angles. As the wrist angle value decreases, so does the influence of r_x. The range of motions that the puck can achieve after training are from 0 to 25 degrees for the angle, and the distance from 50 to $350\,\mathrm{cm}$.

6 Results and discussion

The essential interest is to evaluate the main contributions: the informed search approach, and its application to efficient task transfer. The hypothesis is that the proposed approach needs significantly less trials to learn a confident and generalisable task models, because the trials generated in this manner are the most explanatory for the model. To quantitatively asses the performance of our approach, we analyse the test execution accuracy, i.e. the ability to reach previously unseen targets [2]. During testing, the robot is presented with a target position (green circle as in Fig. 1a) and required to generate appropriate movement parameters for a swing action that will pass the puck to the target. We evaluate the accuracy using 28 different target positions, placed in the mechanically feasible range with 4 increments of the angle $\{0, 10, 15, 20\}$, and 7 of the distance $\{100, 120, 150, 175, 200, 250, 300\}$. These coordinates have not necessarily been reached during training. For specific target coordinates, the model is inverted to give an appropriate and unique movement parameter vector, as described in Sec 4.3. The final repeatability is the one achievable by the robot hardware ($\pm 5\,\mathrm{cm}$) and is consistent.

Firstly, we compare the results of our approach to those of a model learned from randomly generated trials. We generated 100 random points in the movement parameter space which were evaluated on the robot and used to create the GPR task models. We produced 5 such random models with different initial seeds, verified their performance on the test target set and averaged the results (see Table 1). As shown, our model is on average twice as accurate and more importantly, almost three times more confident than the models produced by random search, based on the standard deviation. This demonstrates that the informed search selects training points which provide the model with better generalisation capabilities. We did not consider the grid search approach, as it is not feasible to evaluate all 18750 movement parameter combinations. Regarding the performance in the related work, in [6] the puck is sent to a target zone of $40\,\mathrm{cm}$ in width, while in [3] there are only three fixed $25\,\mathrm{cm}$-wide goals, in which the execution is deemed as successful. From the results, our method on average achieves better accuracy over 28 previously unseen target positions.

[2]The code and experiment data will be made available on the website of the lab upon publication.

Table 1: Performance comparison of the achieved accuracy in Euclidean distance between the puck and the target, averaged over 28 test target positions.

Movement Generation Method	Mean [cm]	STD [cm]
"Original" environment (blue puck, hardwood floor)		
Informed Search	29.48	16.33
Random Sampling	64.18	45.72
Inexperienced Volunteers	32.16	27.82
Experienced Volunteers	22.96	18.07
New environments		
"Original" model (blue puck, marble floor)	66.18	50.75
Re-trained model (blue puck, marble floor)	43.73	37.08
"Original" model (red puck, marble floor)	63.4	41.85
Re-trained model (red puck, marble floor)	38.32	31.05

Secondly, we compare our approach to human-level performance. We asked 10 volunteers who had no previous ice hockey experience and 5 members of the college ice hockey club to participate, under the same settings as the robotic counterpart. They had 24 practice shots to get accustomed to the stick, puck and the surface. After, they were presented with the same test target positions, and their averaged results are presented in Table 1. We have to emphasise that such a comparison is not straightforward to analyse: this task is difficult for a human as it requires repeatability in the arm control and hand-eye coordination; although the inexperienced subjects have not practiced hockey previously, through their lifetime they have developed a good general notion of the physical rules and limb control. The inexperienced volunteers achieve slightly worse accuracy, yet the variance among the subjects is high, which could be attributed to their various skillsets that are more or less akin to ice hockey. Expert volunteers performed better than the robot and this can be explained with their domain knowledge. From qualitative observations we note that several inexperienced volunteers who showed good performance, discovered that sliding the puck in the blade on the ground improves the accuracy. This technique was employed by all expert volunteers and was also learned by the robot.

Finally, we demonstrate the task transfer aspect of the proposed approach by re-learning the task models for different environments. The robot can adapt and perform the task in a new environment by executing only a the set of 24 movement parameter vectors that generated successful trials from the "original" training session (standard puck on hardwood floor), not all 100 trials. The successful trials are independent of the environment and provide descriptive samples for the GPR task models. The new environments we consider are the marble floor which has a higher friction coefficient than the hardwood floor, and a wooden puck (red puck) which is lighter than the standard puck. Successful trials are executed by the robot on the new surface, using both pucks. Two new task models are learned, evaluated on the test target set, and the results are shown in Table 1. As a benchmark, we show results of directly transferring the model learned in the "original" environment. The decrease in accuracy is due to the higher friction and thus decreased sensitivity, so that not all test positions could be achieved. However, we see that using the blue puck as in the "original" setup, on the new floor performs worse than the lighter (red) puck, which can be explained by the fact that a lighter puck on a higher-friction (marble) floor acts as an equivalent to a heavier puck on a lower-friction (hardwood) floor. Even though completely new task models are learned after only 24 trials, the average accuracy is still in line with the literature examples and outperforms the random case by more than $20\,\mathrm{cm}$.

7 Conclusions and future work

We have presented a probabilistic framework for learning the robot's task model and exploration components based solely on its sensory data, by means of informed search in the movement parameter space. The presented approach is validated on a physical robot doing bimanual manipulation of an ice hockey stick in order to pass the puck to target positions. The robot learns the task from scratch in approximately $45\,\mathrm{min}$ with an accuracy comparable to human-level performance, and is capable of adapting it in different environments in significantly less time. We further plan to explore the applicability of this approach to sequential tasks as a policy search exploration method.

References

[1] Adrien Baranes and Pierre-Yves Oudeyer. Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 2013.

[2] Josh Bongard and Hod Lipson. Automatic synthesis of multiple internal models through active exploration. In *AAAI Fall Symposium: From Reactive to Anticipatory Cognitive Embodied Systems*, 2005.

[3] Yevgen Chebotar, Karol Hausman, Marvin Zhang, Gaurav Sukhatme, Stefan Schaal, and Sergey Levine. Combining model-based and model-free updates for trajectory-centric reinforcement learning. *arXiv preprint arXiv:1703.03078*, 2017.

[4] David A Cohn, Zoubin Ghahramani, and Michael I Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 1996.

[5] Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 2015.

[6] Christian Daniel, Gerhard Neumann, Oliver Kroemer, and Jan Peters. Learning sequential motor tasks. In *International Conference on Robotics and Automation*, 2013.

[7] Christian Daniel, Malte Viering, Jan Metz, Oliver Kroemer, and Jan Peters. Active reward learning. In *Robotics: Science and Systems*, 2014.

[8] Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning*, 2011.

[9] Yiannis Demiris and Anthony Dearden. From motor babbling to hierarchical learning by imitation: a robot developmental pathway. *Workshop on Epigenetic Robotics*, 2005.

[10] Cristian Dima, Martial Hebert, and Anthony Stentz. Enabling learning from large datasets: Applying active learning to mobile robotics. In *International Conference on Robotics and Automation*, 2004.

[11] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). *Evolutionary computation*, 2003.

[12] Verena Heidrich-Meisner and Christian Igel. Neuroevolution strategies for episodic reinforcement learning. *Journal of Algorithms*, 2009.

[13] Auke J Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. In *Advances in neural information processing systems*, 2003.

[14] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Active learning with gaussian processes for object categorization. In *International Conference on Computer Vision*, 2007.

[15] Seyed Mohammad Khansari-Zadeh, Klas Kronander, and Aude Billard. Learning to play minigolf: A dynamical system-based approach. *Advanced Robotics*, 2012.

[16] Petar Kormushev, Yiannis Demiris, and Darwin G Caldwell. Kinematic-free position control of a 2-dof planar robot arm. In *International Conference on Intelligent Robots and Systems*, 2015.

[17] OB Kroemer, Renaud Detry, Justus Piater, and Jan Peters. Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems*, 2010.

[18] Alonso Marco, Philipp Hennig, Jeannette Bohg, Stefan Schaal, and Sebastian Trimpe. Automatic lqr tuning based on gaussian process global optimization. In *International Conference on Robotics and Automation*, 2016.

[19] KM Newell. Motor skill acquisition. *Annual review of psychology*, 1991.

[20] Matthias Plappert, Rein Houthooft, Prafulla Dhariwal, Szymon Sidor, Richard Y Chen, Xi Chen, Tamim Asfour, Pieter Abbeel, and Marcin Andrychowicz. Parameter space noise for exploration. *arXiv preprint arXiv:1706.01905*, 2017.

[21] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[22] Carl Edward Rasmussen and Christopher KI Williams. Gaussian processes for machine learning. *The MIT Press, Cambridge, MA, USA*, 2006.

[23] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. Gaussian process classification and active learning with multiple annotators. In *International conference on machine learning*, 2014.

[24] Thomas Rückstiess, Frank Sehnke, Tom Schaul, Daan Wierstra, Yi Sun, and Jürgen Schmidhuber. Exploring parameter space in reinforcement learning. *Paladyn*, 2010.

[25] Tim Salimans, Jonathan Ho, Xi Chen, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[26] Jens Schreiter, Duy Nguyen-Tuong, Mona Eberts, Bastian Bischoff, Heiner Markert, and Marc Toussaint. Safe exploration for active learning with gaussian processes. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2015.

[27] Sambu Seo, Marko Wallat, Thore Graepel, and Klaus Obermayer. Gaussian process regression: Active data selection and test point rejection. In *International Joint Conference on Neural Networks*, 2000.

[28] Burr Settles. Active learning literature survey. Computer sciences technical report, University of Wisconsin-Madison, 2009.

[29] Sebastian B Thrun and Knut Möller. Active exploration in dynamic environments. In *Advances in neural information processing systems*, 1992.

[30] Zi Wang and Stefanie Jegelka. Max-value entropy search for efficient bayesian optimization. *arXiv preprint arXiv:1703.01968*, 2017.

[31] Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *Journal of Machine Learning Research*, 2014.

Appendix

Algorithm 1 Task and Exploration model learning

Inputs:
  movement parameters: $(l_x, l_y, r_x, r_y, w, s)$
Initialize:
  PIDF $\leftarrow \mathbf{\Pi_0}$
  UIDF $\leftarrow 1$
repeat
  SIDF $\leftarrow$ PIDF * UIDF
  $\boldsymbol{x}_t \sim$ SIDF ▷ Sample movement parameter vector
  trial_outcome, θ_{puck}, L_{puck} = ROBOT($\boldsymbol{x}_t$) ▷ Trial execution on the robot
  if trial_outcome $=$ failed **then**
    $\boldsymbol{\Sigma_{fail}}[\boldsymbol{x}_t] \mathrel{+}= 1$
    PIDF $\mathrel{-}= \mathcal{N}(\boldsymbol{x}_t, \boldsymbol{\Sigma_{fail}})$
    UIDF $= GPR(\boldsymbol{x}_t)$
  else
    $\boldsymbol{\Sigma_{fail}}[\boldsymbol{x}_t] \mathrel{-}= 1$
    PIDF $\mathrel{+}= \mathcal{N}(\boldsymbol{x}_t, \boldsymbol{\Sigma_{succ}})$
    $\boldsymbol{\theta_x}, \boldsymbol{L_x}$, UIDF $= GPR(\boldsymbol{x}_t)$
until stopping_conditions