



# Active learning via informed search in movement parameter space for efficient robot task learning and transfer

Nemanja Rakicevic<sup>1</sup> · Petar Kormushev<sup>1</sup>

Received: 15 January 2018 / Accepted: 7 February 2019 / Published online: 21 February 2019  
© The Author(s) 2019

## Abstract

Learning complex physical tasks via trial-and-error is still challenging for high-degree-of-freedom robots. Greatest challenges are devising a suitable objective function that defines the task, and the high sample complexity of learning the task. We propose a novel active learning framework, consisting of decoupled *task model* and *exploration* components, which does not require an objective function. The task model is specific to a task and maps the parameter space, defining a trial, to the trial outcome space. The exploration component 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 analyse the performance of our framework in a simulation environment and further validate it 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. Decoupling the framework's components also enables efficient skill transfer to new environments which is validated experimentally.

**Keywords** Active learning · Parameterised movements · Parameter space exploration · Bimanual manipulation

## 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 (Newell 1991). These trials do not necessarily lead directly to accomplishing the task, but eventually a confident task execution is learned (Pugh et al. 2016). For robot learning, each trial can be uniquely defined, i.e. parameterised, by a set of movement parameters (Ijspeert et al. 2013) which means that performing a trial is equivalent to selecting a point in the movement parameter space and evaluating it.

In this paper, we focus on a problem of learning a task through trial and error, where a task can be executed by selecting an appropriate point in the movement parameter space. Our aim is to develop a sample-efficient approach that avoids

trials which are not useful, by not doing random or exhaustive exploration during learning, intended for systems where trial execution is expensive. Moreover, during the learning phase, we do not provide any prior information about the task (e.g. goal position or cost function) or the environment to the agent, in order to reduce inputted domain knowledge and aim to make the approach “task-agnostic”.

To this end, we introduce a novel iterative and online active-learning approach, which performs informed search in the movement parameter space defining the trials, in order to sample datapoints. The proposed learning framework consists of a *task model* and an *exploration* component. The task model is implemented as a Gaussian Process (GP) Regression (GPR) (Rasmussen and Williams 2006) function that maps the movement parameters as inputs, to the trial outcomes as outputs. The exploration component performs search in the movement parameter space to find a parameter vector that encodes a subsequent most informative trial for the task model. This component represents a composite query strategy in the Active Learning parlance, obtained via probabilistic modelling of previous trial data and uncertainty inherent to the GPR task model. It is implemented as a probability distribution over the movement parameter space, from which parameter vectors are sampled. During the learning

---

✉ Nemanja Rakicevic  
n.rakicevic@imperial.ac.uk

Petar Kormushev  
p.kormushev@imperial.ac.uk

<sup>1</sup> Robot Intelligence Lab, Dyson School of Design Engineering, Imperial College London, London, UK

phase, the exploration component iteratively finds datapoints in the parameter space used to fit the task model and thus lower the task model's posterior uncertainty. Actual performance of the trial outcomes, i.e. cost function, is not used by either component as the desired target outcomes are not provided. This renders the components independent from a specific task requirement.

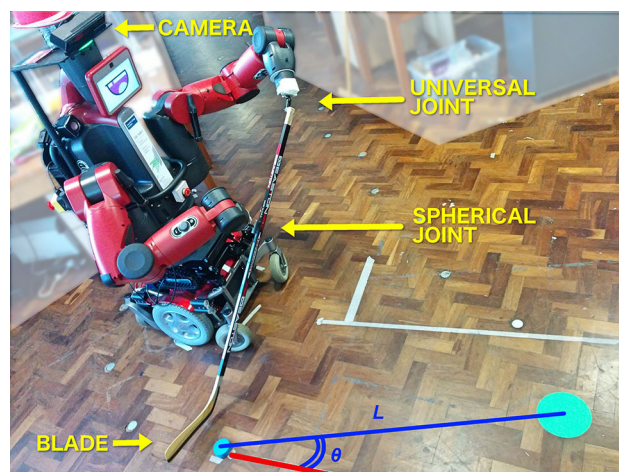
For transfer we consider tasks which can be different but have the same interface, i.e. response, from the environment, and the same parameter space. Meaning, the exploration and sampling of the datapoints for the task model is independent of a particular task and related to the agent's kinematic model. Therefore, the same exploration sequence would be applied in different environments. Since the exploration component maintains the information about the successful trials, these trials can be directly reproduced (transferred) in different environments, in order to gather data and fit the task model for the new environment. As a consequence, new task models can be learned from scratch with significantly less trial evaluations.

To present and analyse the performance of the proposed framework we use the MuJoCo (Todorov et al. 2012) simulation environment as well as a physical robot. Both the simulated and real robot task are similar, in that they employ an agent which tries to learn how to move another object (puck) to arbitrary locations using its body. During the testing phase, the agent is presented with a set of previously-unseen arbitrary target positions. It is expected to automatically generate an appropriate movement action, based on the learned task model, to hit the puck so that it lands on the target. This is the actual task that the agent needs to perform well. For evaluation on the real robot we have selected the ice hockey puck-passing task, as shown in Fig. 1. 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. The number of input parameters is given (which represent the displacement of each degree of freedom) and their ranges, without contextual information regarding their influence or importance.

To summarise, the main contributions of this work are:

- The probabilistic framework for trial-and-error robot task learning, based on a task-agnostic and sample-efficient search of the trial parameter space. This is achieved



**Fig. 1** Experimental setup: robot DE NIRO uses both arms to maneuver the ice hockey stick and learns the skills needed to pass the puck (blue) to user-specified target positions (green). Estimation of the polar coordinates  $\theta$  and  $L$  is done using the head-mounted Kinect camera. The red line in the bottom is parallel to the robot heading direction and is the zero-angle reference axis (Color figure online)

through the exploration component which is a novel composite query function consisting of the model uncertainty and the penalisation function.

- As a consequence of decoupling the task model and exploration components, efficient task transfer to new environments is possible, as shown experimentally. The robot successfully learns the task models in the new environments in significantly less trials, by executing only successful trials generated in the previous environment.

The rest of the paper is organised as follows: Sect. 2 gives an overview of the related work. Section 3 formulates the problem we are addressing and in Sect. 4 we present the proposed framework. The proof of concept on a simulated task is given in Sect. 5, and robot experiment is presented and results are discussed in Sect. 6. Finally, we conclude and discuss the future directions in Sect. 7.

## 2 Literature review

### 2.1 Active learning and Bayesian optimisation

The proposed approach can be characterised as an Active Learning (AL) approach (Settles 2012), a field similar to Bayesian Optimisation (BO) (Mockus 1994) and Experimental Design (ED) (Santner et al. 2013). The purpose of these approaches is to efficiently gather datapoints used to fit a model, which are most informative about the underlying data distribution. Such datapoints enable learning a good model in a sample-efficient way. The idea behind ED is that all the

datapoints are defined offline, before the execution, which limits the flexibility of the approach. The difference between AL and BO is rather subtle but important, which is why we focus more on them in this section. Both approaches model a low-fidelity surrogate function, which is usually obtained as a posterior over the unknown function. The mean and variance of this surrogate function are used, through a set of rules, in order to query a new input from the domain to evaluate the unknown true function over. In BO terminology, this set of rules is called an Acquisition Function, while in AL it is called a Query Strategy. In BO the function that is evaluated needs to be optimised, while in AL the query strategy is actually a mechanism for obtaining labels for input data to further improve the function estimate. Therefore, the end-goal of BO is to optimise an underlying function (whence the name), while for AL it is not. Consequentially, the nature of the acquisition and query functions slightly differ as in BO they also need to improve the evaluated function's value. The query function in AL focuses solely on querying inputs that will be most informative for a supervised learning problem, and minimise the uncertainty in such a model. Such query functions do not have explicit exploitation components, as opposed to their counterparts in BO, thus no explicit function optimisation is being done.

Some of the most popular BO Acquisition Functions are: probability of improvement (Kushner 1964), expected improvement (Moćkus 1975), GP upper confidence bound (GP-UCB) (Srinivas et al. 2010), entropy search (Hennig and Schuler 2012) and predictive entropy search (Hernández-Lobato et al. 2014). The objective function usually quantifies the model performance on a specific task. If the acquisition function needs to “know” the actual value of an objective function in order to select the next parameter to evaluate, this selection inherently carries task information embedded in the objective function. As opposed to BO, our proposed approach tries to avoid this. Several interesting examples in the literature use BO in robotic applications (Lizotte et al. 2007; Martinez-Cantin et al. 2009; Tesch et al. 2011; Calandra et al. 2016).

When AL Query Functions are implemented with GPs, similarly to BO acquisition functions, they provide uncertainty-based exploration (Seo et al. 2000; Kapoor et al. 2007; Kroemer et al. 2010; Rodrigues et al. 2014). However, they do not necessarily need to rely on the surrogate's posterior, one example being empirically estimating the learning progress (Lopes et al. 2012) which requires performance evaluation. Other examples include uncertainty sampling introduced by Lewis and Gale (1994), similar to our approach where the authors use the classifier prediction uncertainty. However, our uncertainty measure is derived from the GP posterior distribution and combined with the penalisation function. Another interesting approach to querying is based on maintaining multiple models for prediction and select-

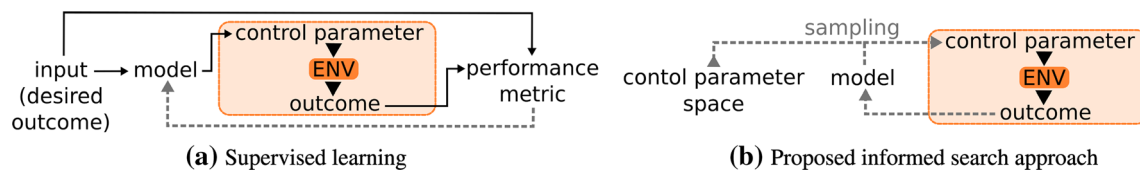
ing those points over whose prediction the models disagree the most Bongard and Lipson (2005) which is related to the notion of query by committee (Seung et al. 1992). Otte et al. (2014) and Kulick et al. (2015) present examples of applying AL to robotics, mostly for learning the parameters of the controller by probing environment interactions. Other robotic applications include Thrun and Möller (1992), Daniel et al. (2014), Dima et al. (2004), Baranes and Oudeyer (2013) and Kroemer et al. (2010) where AL helps relieve the sample complexity—one of the main limitations imposed by hardware for robotic experiments. Most of the above-mentioned AL sample query strategies, which rely on prediction uncertainty, do not take into account the actual order of acquiring datapoints explicitly, which is important to understand the boundaries within the parameter space. This is particularly needed in robotics, where physical constraints play a crucial role. Therefore, we explicitly include such information within our exploration component. Including safety constraints within the BO framework has been done through the optimisation constraints in the objective function (Englert and Toussaint 2016; Berkenkamp et al. 2016). Gelbart et al. (2014) and Schreiter et al. (2015) model safety constraints as a separate GP model, but this approach requires additional computational resources.

There have been several approaches in the literature employing GPs to learn mapping functions similar to our task model (Nguyen-Tuong et al. 2009; Nemeć et al. 2011; Forte et al. 2012). The latter two generate full trajectories encoded via DMPs and introduce constraints that guide the new policy to be close to the previously demonstrated examples in the trajectory database.

## 2.2 Parameter space exploration

The concept of good exploration strategies is crucial in supervised learning, as well as RL, where it can improve sample selection and sample-efficiency. 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. This can reduce the variance caused by noisy trajectories, and generally avoids premature convergence to suboptimal solutions (Rückstieß et al. 2010; Plappert et al. 2017). Evolutionary Strategy-based methods Hansen et al. 2003; Heidrich-Meisner and Igel 2009; Wierstra et al. 2014; Salimans et al. 2017) introduce noise in the parameter space to guide exploration, acting as a black-box optimiser, but have poor sample-efficiency.

The main inspiration for the proposed work is to shift away from the common utilitarian paradigm of task learning through optimising some utility (cost) function. Some of the approaches in this direction develop exploration which tends to be decoupled from the actual task definition embodied in the cost function. A recent parameter space search approach



**Fig. 2** Diagrams comparing the information flow in the **a** supervised learning paradigm (maps outcome to control parameter) and **b** proposed informed search approach (maps control parameter to outcome). Solid

line is the data pipeline, while the dashed line indicates updates. The orange box is the task execution, i.e. environment interaction

uses the notion of curiosity Pathak et al. (2017) where an intrinsic curiosity module is implemented to promote exploration, by learning to distinguish changes in the environment caused by the agent from random ones. The Quality Diversity (Pugh et al. 2016) family of approaches such as MAP-elites (Mouret and Clune 2015; Cully et al. 2015) and Novelty Search with Local Competition (Lehman and Stanley 2011b) perform exploration by encouraging diversity in candidate behaviours and improving fitness over clusters of behaviours in the behaviour space. However, in our presented problem formulation we do not aim to derive diverse behaviours, rather to find those for which the system is uncertain about and which avoid dangerous situations. More importantly, there is no notion of relative task fitness involved, as the proposed exploration component of our method generates points which are informative for the model, unrelated to their actual fitness as the fitness is task specific. The notion behind the proposed framework is akin to the concept of objective-free learning Lehman and Stanley (2011a) which promotes diversifying the behaviours as an alternative to having the objective as the only mean of discovery, which can in fact lead to deceptive local optima (Pugh et al. 2016). As opposed to promoting novelty, our approach actually selects behaviours which are most useful for the task model. Methods relying on techniques like Motor Babbling (Demiris and Dearden 2005; Kormushev et al. 2015), Goal Babbling (Rolf et al. 2010) and Skill Babbling (Reinhart 2017) can learn the robot's forward/inverse model by iteratively performing random motor commands and recording their outcomes. However, these methods are usually data-inefficient due to random exploration. Kahn et al. (2017) use neural networks with bootstrapping and dropout, to obtain uncertainty estimates of the observations for predicting possible collisions and adapting the robot control accordingly. These estimates are not further used to explore alternative control policies. Deisenroth et al. (2015) show that using GPs within model-based Reinforcement Learning (RL) helps in improving the sample-efficiency of learning the task. The posterior mean and variance are used to address the exploration/exploitation trade-off during policy learning. Still, the above-mentioned approaches require an explicit cost function optimised by the agent in order to learn the task. Learning robotic tasks with complex kinematics, by exploring the low-level control

space is presented in Kormushev et al. (2015). Additional elements such as links and leverage points are incorporated into the original kinematic chain to skew the mapping of motor torques to end-effector pose. The robot adjusts to these modifications, without an explicit model of the robot's kinematics or extra links provided, but such approach would have difficulties when scaled.

### 2.3 Bimanual skill learning

There are few examples in the literature of learning to play bimanual ice hockey, but none of them simultaneously address: bimanual manipulation, and using a tool which distorts/modifies the original robot kinematics. Relevant example of single-arm robot learning to play hockey using RL is presented in Daniel et al. (2013) where the robot learns to send the puck into desired reward zones and gets feedback after each trial. Kinaesthetic teaching is required to extract the shape of the movement which is then improved. Recently, Chebotar et al. (2017) combined 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 to provide 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

The main problem we are trying to solve is efficient high-dimensional parameter search. We employ the proposed exploration component, to search for movement parameter datapoints used to fit our task model component. The goal is to eventually have a good model performance during testing, on the task of reaching the desired outcomes. Figure 2 compares the information flow diagrams for the standard supervised learning paradigm and our proposed approach. The model outputs a *control* (e.g. movement) parameter vector, given an *input*, and the trial *outcome* is the produce of this output when applied in the environment. The *performance metric* is a cost function comparing trial *outcome* and target

*desired outcome*, and is used in supervised learning (Fig. 2a) to update the model. In our case (Fig. 2b), the “input” can be seen as the whole movement parameter space from which the *model* samples and outputs a movement parameter vector. The proposed approach does not use the model *performance metric* to update the *model*, rather the trial *outcome*, since *desired outcomes* are not provided nor needed. To demonstrate the proposed approach, we consider a task in which the agent needs to perform a movement that displaces its body from a fixed initial configuration. This movement can potentially lead to a contact with another object (e.g. a puck) moving this object to a certain location. One such executed event is called a *trial*. The movement of the object is governed by the dynamical properties of the environment (object mass, surface friction coefficient, obstacles etc) which are unknown to the agent. The only feedback that the agent receives, is whether the movement it performed successfully made contact with the object (successful trial) or not (failed trial), and in the former case, what is the final resting position of the object (i.e. trial/task outcome).

The action that the agent performs is defined by a vector of  $D$  movement parameters  $x = [\Delta q^1, \dots, \Delta q^{D-1}, s]$  that define the whole motion sequence as a “one-shot” action. This movement parameter vector contains the displacements  $\Delta q$  for each of the actuators w.r.t. a fixed starting configuration, and the speed of the overall action execution  $s$ . We assume that there already exists a position controller that translates the goal position to a trajectory. The set of all movement parameter vectors that encode actions (trials) is the *movement parameter space*. Even though this space is continuous, we discretise it to obtain a finite set of possible combinations. This allows us to perform fast and exact inference over the parameter space, without the need for approximate inference methods. In the simulation experiments we use revolute joints so the parameters are given in radians. Their units are the same even though their ranges might be different. The same holds in robotic experiments where the parameters are displacements in the Cartesian space measured in centimeters, with the exception of the wrist angle which is in radians. Although the wrist angle has different units, the effect it causes can be comparable to the displacements in centimeters. After the trajectory has been executed, in case of a successful trial, the obtained trial outcome can be any value (both continuous or discrete) and is used to fit the task model component. Both the successful and failed trials contribute to the exploration component. Under such a setup, the agent does not optimise for a particular task performance, but rather tries to avoid failed trials.

## 4 Proposed approach

The base assumption of our approach is that similar movement parameter vectors result in similar trial outcomes.

Therefore, the task regression mapping function is smooth, without hard discontinuities, i.e. Lipschitz continuous. In order to provide a sufficiently diverse sample distribution for the regression model to create a generic mapping during training, successful trials are necessary, i.e. agent needs to move the object. The main challenge is selecting the trial to evaluate next which will lead to the highest information gain. The proposed approach consists of two decoupled components updated using previous experience, i.e. previous trials—the task model and exploration components. They are implemented as functions over the movement parameter space, mapping each movement parameter vector to a certain value. The mathematical formulation, together with the underlying intuition behind the task model and exploration components is given in Sects. 4.1 and 4.2, respectively.

### 4.1 Task model component

The task model component uses the information from scarce successful trials, and creates a mapping between the movement parameter space ( $X$ ) as input, and the trial outcomes—puck’s final position  $(\theta_{puck}, L_{puck})$  as output. This component creates two independent task models for each of the puck’s polar coordinates, angle and distance (as depicted later on in Sect. 5 in Fig. 7a, b, 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. Using the notation from the previous section, let us define a point in the movement parameter space  $x \in \mathbb{R}^D$ . The main assumption is that for any finite set of  $N$  points  $X = \{x_i\}_{i=1}^N$ , the corresponding function evaluations (in our case trial outcome) at these points can be considered as another set of random variables  $\mathcal{F} = \{f_{x_i}\}_{i=1}^N$ , whose joint distribution is a multivariate Gaussian:

$$\mathcal{F} \sim \mathcal{N}(\mu(X), K(X, X))$$

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

$$\begin{aligned} \begin{bmatrix} f_{X^*} \\ f_X \end{bmatrix} &\sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{K}_{**} & \mathbf{K}_*^T \\ \mathbf{K}_* & \mathbf{K} \end{bmatrix}\right) \\ p(f_X | f_{X^*}, \mathbf{X}, \mathbf{X}^*) &\sim \mathcal{N}(\mathbf{K}_* \mathbf{K}_{**}^{-1} f_{X^*}, \\ &\quad \mathbf{K} - \mathbf{K}_* \mathbf{K}_{**}^{-1} \mathbf{K}_*^T) \end{aligned} \quad (1)$$

We assume a mean of 0 for our prior as we do not want to input any previous knowledge in our task model. Similarly to  $\mathbf{K}$ ,  $\mathbf{K}_{**}$  is the matrix of covariances for all the pairs from the set  $\mathbf{X}^*$ , and  $\mathbf{K}_*$  gives us the similarity of the successful parameter vectors  $\mathbf{X}^*$  to each point in the parameter space  $\mathbf{X}$ . Within the kernel definition we also consider zero mean Gaussian noise,  $\varepsilon \sim \mathcal{N}(0, \sigma_\varepsilon^2)$ , to account for both modelling and measurement inaccuracies. We evaluated the performance using the squared exponentiation (SE), Matern 5/3 and the rational quadratic (RQ) kernels. The best performing kernels are SE:  $\mathbf{K}_{SE}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 e^{-\frac{d^2}{2\sigma_l}}$  and RQ:  $\mathbf{K}_{RQ}(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \left(1 + \frac{d^2}{2\alpha\sigma_l^2}\right)^{-\alpha}$ , and these results are presented in Fig. 6. The distance measure  $d$  is defined as the Euclidean distance between the points in the parameter space  $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\| = \sqrt{\sum_{j=1}^D (x_j - x'_j)^2}$ . Even though the concept of a distance metric in a high-dimensional space is not straightforward to decide and interpret, we opt for the Euclidean distance based on the discussion from Aggarwal et al. (2001) who argue that in problems with a fixed high dimensionality, it is preferable to use a lower norm. Moreover, the presented kernel showed good empirical performance. From the similarity measure given by the kernel we get that for the points which are far away from each other, will have a higher variance associated with their prediction. The coefficients  $\alpha = D/2$ ,  $\sigma_f^2$  and  $\sigma_l^2$  are the scaling parameter, variance and the lengthscale of the kernel, respectively.

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

Even though it is possible to learn the GPR hyperparameters from data, we do not perform this because of: (i) Low number of samples; as the main goal of our approach is sample-efficiency, having a low number of samples and learning the hyper parameters with the marginal likelihood is very likely to give overfitting results (at least several dozens of samples are needed to learn something meaningful (Cully

et al. 2015)). (ii) Search instability; the order of acquiring samples is important and each subsequent point in the search depends on the previous ones. Changing the GPR hyperparameters after each step, would cause large variance in the sample acquisitions which may lead to instability. Therefore, we do extensive search of the hyperparameters, but keep them fixed throughout the training phase.

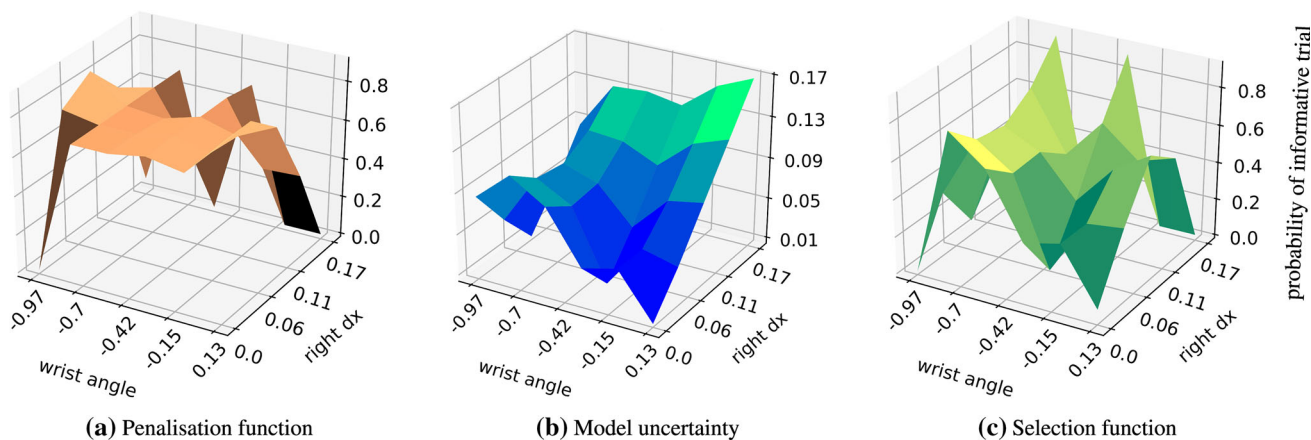
## 4.2 Exploration component

The exploration component exploits all the past 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. Since the movement parameters used as inputs for GPR are the same for both the distance and angle task model, their corresponding GPR uncertainty will also be the same. The penalisation function and the GPR model uncertainty 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. Therefore, multiplying these two functions acts as a kind of particle filter. Since we are interested in the relative “informativeness” of each point in the parameter space when sampling the next trial, the actual absolute values of these functions do not play a crucial role. An example of these IDFs is visualised in Fig. 3.

**Penalisation IDF (PIDF)** probabilistically penalises regions around the points in the movement parameter space which have led to failed trials. This inhibits repetition and reduces the probability of selecting parameters leading to failed trials. In our experiments, a trial is failed if the agent does not move the object. Additionally, in the simulation experiment, the trial is stopped if the agent contacts (hits) the wall. In the robotic experiment, fail cases are also when:

- Inverse kinematic solution cannot be found for the displacements defined by the movement parameters.
- The displacements would produce physical damage to the robot (self collision, ignoring the stick constraint or hitting itself with the stick).
- Mechanical fuse breaks off due to excessive force.
- Swing movement misses the puck (no puck movement).

The PIDF is implemented as a mixture of inverted  $D$ -dimensional Gaussians (MoG) (2), as they consider all failed trials evenly. Gelbart et al. (2014) and Englert and Toussaint (2016) chose GP for this, however, MoG provide better expressiveness of multiple modes in a distribution. PIDF is initialised as a uniform distribution  $p_p(\mathbf{X}) \equiv \mathcal{U}(\mathbf{X})$ . The uniform prior ensures that initially all the movement actions



**Fig. 3** The components used to generate movement parameters vectors that define the trials. The figures show: **a** penalisation function which inhibits the unsuccessful movements, i.e. failed trials, **b** task model uncertainty, and **c** selection function combines the previous two functions in order to get the distribution from which the parameter vector

is sampled. The visualisation of the 6-dimensional parameter space is done by fixing the remaining parameters and showing the variance of the model w.r.t. the wrist angle and right hand displacement along the x-axis

have an equal probability of being selected. Each of the  $K \ll N$  evaluated trials is represented by a Gaussian with a mean  $\mu_k^P$  coinciding with the parameter vector  $x_k$  associated with this trial. Coefficient *cov* is the covariance coefficient hyperparameter. Covariance matrix  $\Sigma_k^P$  is a diagonal matrix, calculated based on how often does each of the  $D$  parameters take repeated values, considering all the previous 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 with frequently repeating values, thus applying higher penalisation and forcing them to change when ‘stuck’. Parameters with evenly occurring values have wider Gaussians.

This procedure inhibits the selection of parameter values which are likely to contribute to failed trials, and stimulates exploring new ones. Conversely, the parameter vector leading to a successful trial is stimulated with a non-inverted and high variance Gaussian, which promotes exploring nearby regions of the space. PIDF can be interpreted as  $p(\text{successful\_trial} | \text{uncertain\_trial})$  i.e. the likelihood that the parameter vector will lead to a successful trial given that the model is uncertain about it.

$$p_p(X) = \mathcal{U}(X) + \sum_{k=1}^K \phi_k \mathcal{N}(\mu_k^P, cov\Sigma), \tag{2}$$

$$\begin{cases} \phi_k = -1, \Sigma = \Sigma_k^P & \text{failed trial} \\ \phi_k = +1, \Sigma = \mathbb{I} & \text{successful trial} \end{cases}$$

**Model uncertainty IDF (UIDF)** is intrinsic to GPR (3) and is used to encourage the exploration of the parameter space regions which are most unknown to the underlying task mod-

els. UIDF is updated for both successful and failed trials, as the exploration does not depend on the actual trial outcomes.

$$p_u(X) = K - K_\star K_{\star\star}^{-1} K_\star^T \tag{3}$$

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

$$p_{sel}(X) \propto p_p(X) p_u(X) \tag{4}$$

The trial generation and execution are repeated iteratively until the stopping conditions are met. Since we are not minimising a cost function, the learning procedure can be stopped when the average model uncertainty (i.e. entropy) drops below a certain threshold. This can be interpreted as stopping when the agent is certain that it has learned some task model. The pseudocode showing the learning process of the proposed framework is presented in Algorithm 1.

### 4.3 Learned task model evaluation

In order to evaluate the performance during the testing phase, it is necessary for the angle  $\theta(x)$  and distance  $L(x)$  task models to be invertible. Given the target coordinates (desired trial outcome), a single appropriate movement parameter vector  $\hat{x}$  defining the swing action that passes the puck to the target needs to be generated. It is difficult to generate exactly

**Algorithm 1** Task and Exploration component learning

---

```

1: Inputs:
   movement parameter space:  $X$ 
2: Initialize:
   PIDF  $\leftarrow \Pi_0$ 
   UIDF  $\leftarrow 1$ 
3: repeat
  ▷ Following Eq. (4)
4:   SIDF  $\leftarrow$  PIDF  $\times$  UIDF
  ▷ Sample movement parameter vector
5:    $x_t \sim$  SIDF
  ▷ Trial execution on the robot
6:   trial_outcome,  $\theta_{puck}$ ,  $L_{puck} =$  EXECUTE_TRIAL( $x_t$ )
7:   if trial_outcome = failed then
8:      $\Sigma_k^P[x_t] += 1$ 
     ▷ Following Eqs. (2) and (3), respectively
9:     PIDF  $-= \mathcal{N}(x_t, cov \Sigma_k^P)$ 
10:    UIDF = GPR( $x_t$ )
11:   else
12:      $\Sigma_k^P[x_t] -= 1$ 
     ▷ Following Eqs. (2) and (3), respectively
13:     PIDF  $+= \mathcal{N}(x_t, cov \mathbb{I})$ 
14:      $\theta_x$ ,  $L_x$ , UIDF = GPR( $x_t$ ,  $\theta_{puck}$ ,  $L_{puck}$ )
15:   end if
16: until stopping_conditions

```

---

one unique parameter vector which precisely realises both the desired coordinate values  $\theta_d$  and  $L_d$ . Therefore, 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  $\epsilon$  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, obtaining the optimal parameter vector could be achieved using any standard optimisation algorithm. More importance can be given to a specific coordinate by adjusting the weighing factor  $\delta$ .

$$\hat{x} = \underset{x}{\operatorname{argmin}} \left( \sqrt{(1 - \delta)(\theta(x) - \theta_d)^2 + \delta(L(x) - L_d)^2} \right) \quad \text{subject to: } p_p(x) < \epsilon \quad (5)$$

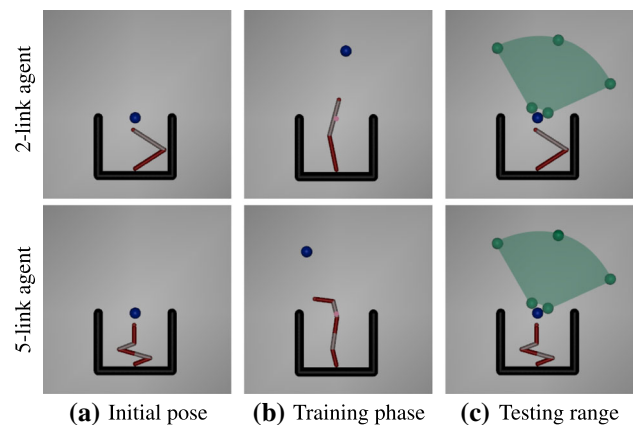
#### 4.4 Task transfer

After the task model is learned, if we were to repeat the approach in a different environment, the algorithm would still generate the same trials, both successful and failed. The new environment can be considered as a different task as long as the parameter space stays the same, but the trial outcomes values change. This is possible due to the fact that the proposed

approach does not take into account the actual puck position values when generating subsequent trials, but rather the binary feedback whether the trial was successful or failed. To reiterate, 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 separate 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. Experimental validation of the task transfer capability of the proposed approach is presented in Sect. 6.3.

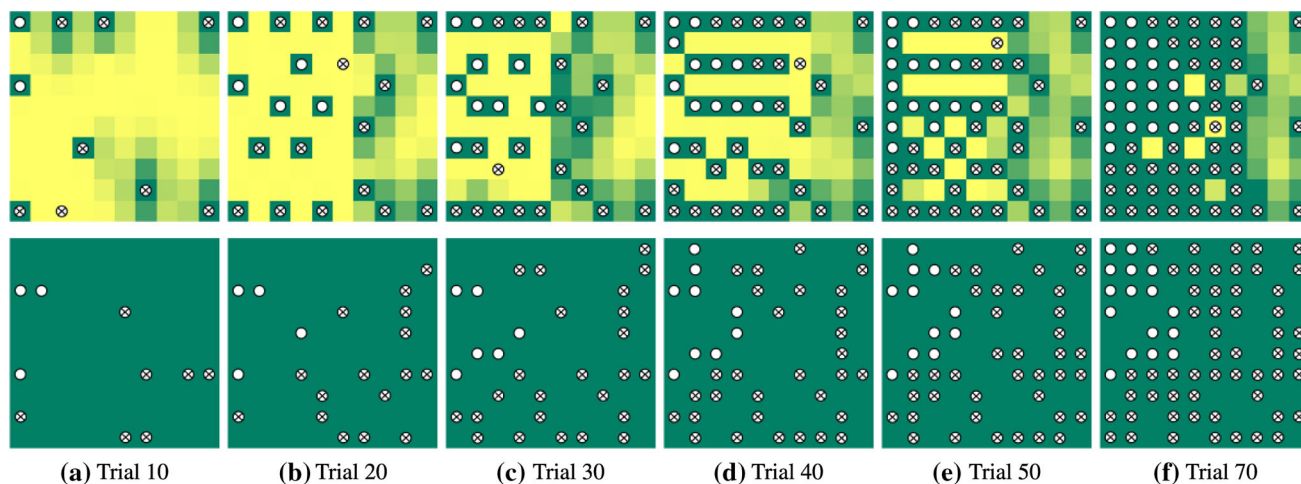
## 5 Simulation experiments and analysis

In order to present and analyse in detail the performance of the proposed approach on a simple task, we introduce the MuJoCo ReacherOneShot environment and evaluate the approach in simulation. We introduce two variants of the environment, with a 2-DOF agent consisting of 2 equal links, and a 5-DOF agent consisting of 5 equal links in a kinematic chain structure, as depicted in Fig. 4. The environment consists of an agent acting in a wall-constrained area, with the initial configuration of the agent and the puck location above it fixed. This environment is a physical simulation where the contacts between each of the components are taken into account, as well as the floor friction. Contact between the puck and the agent makes the puck move in a certain direc-



**Fig. 4** The MuJoCo ReacherOneShot-2link (top row) and ReacherOneShot-5link (bottom row) environments used for simulation. The links are shown in different colours for clarity, the walls are black and the puck the agent needs to hit is blue. Column **a** shows the initial position, **b** a training trial (no targets given) and **c** the testing phase with the testing area and sample targets in green (Color figure online)





**Fig. 5** Comparison of the sample selection progress for both the proposed informed approach (top row) and the random counterpart (bottom row), up to the 70th trial on a  $10 \times 10$  parameter space. The  $\otimes$  symbols represent failed trials while  $\circ$  are the successful trials. The grid

colours represent the probability of selecting a particular point, with green being 0 and yellow 1. For the random approach the probabilities are not applicable as all points are equally probable to be sampled (Color figure online)

tion, while the collision between the agent and the wall stops the simulation experiment and the trial is classified as failed.

The agent learns through trial and error how to hit the puck so it moves to a certain location. The hitting action is parameterised, with each parameter representing the displacement of a certain joint w.r.t. the initial position, and is executed in a fixed time interval. During the training phase, the agent has no defined goal to which it needs to optimise, but just performs active exploration of the parameter space to gather most informative samples for its task model. We have chosen this task as it is difficult to model the physics of the contacts, in order to estimate the angle at which the puck will move. Moreover, the walls on the side act as an environmental constraint to which the agent must adapt.

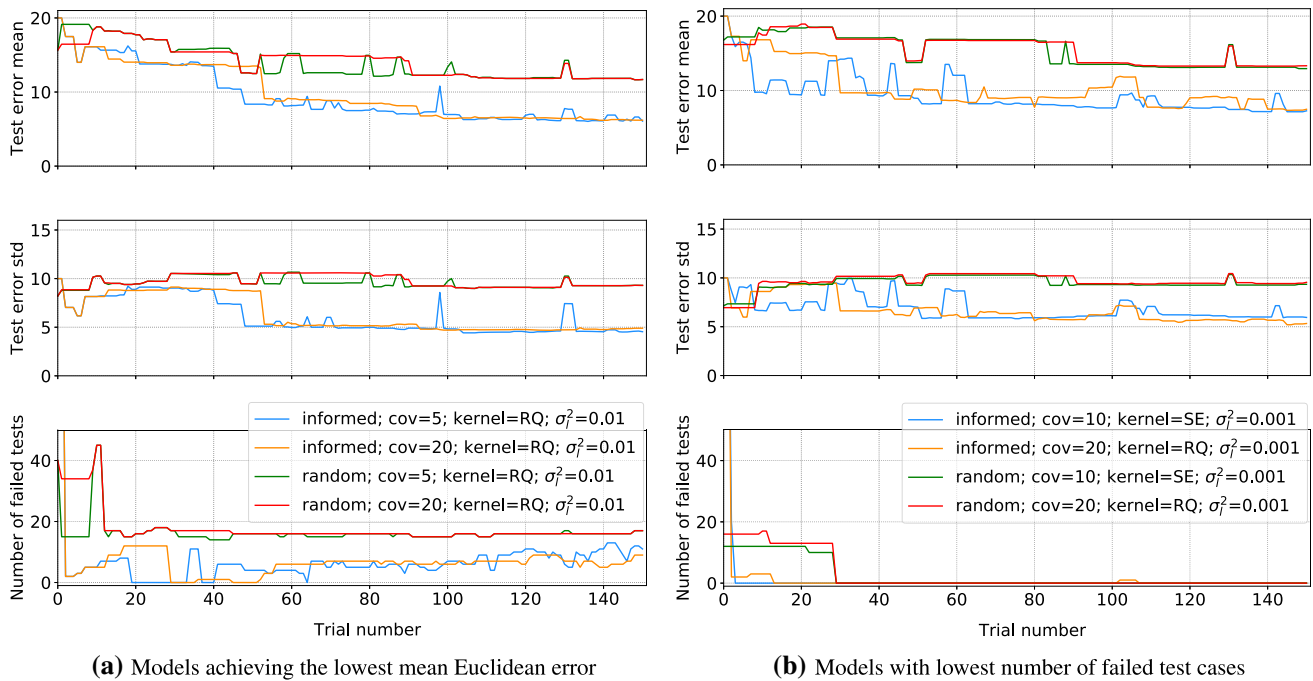
## 5.1 Experiment with 2-link agent

In order to properly visualise all the components of the proposed framework to be able to analyse them, we introduce a 2-link agent with only two parameters defining the action. The range of base-link joint ( $joint_0$ ) is  $\pm \frac{\pi}{2}$ , while for the inter-link joint ( $joint_1$ ), the range is  $\pm \pi$  radians. The action execution timeframe is kept constant, thus the speed depends on the displacements.

To demonstrate the efficacy of the proposed informed search approach, we compare and analyse its trial selection process to random trial sampling. We show this on a crude discretisation where each joint parameter can take one of 10 equally spaced values within its limits, producing the parameter space of 100 elements. Figure 5 shows side-by-side the trial selection progress up to the 70th trial, for both the proposed informed approach and the random counter-

part. In the beginning both approaches start similarly. Very quickly, the proposed informed approach appears to search in a more organised way, finding the ‘useful region’ of the parameter space that contains successful trials and focuses its further sampling there, instead of the regions which produce the failed trials. After 50 trials we can see that the distribution of the points is significantly different for the two approaches even in this simple low-dimensional example. The number of sampled successful trials with the proposed informed approach is 16, as opposed to 11 obtained by the random approach, while the remaining are failed which do not contribute to the task models. This behaviour is provided by the PIDF which penalises regions likely to lead to failed trials, and the sampling in the useful region is promoted by the UIDF which seeks samples that will improve the task model. We note that the highest concentration of the failed trials from the informed approach is actually in the border zones, around the useful region and at the joint limits. These regions are in fact most ambiguous and thus most interesting to explore. After the 70th trial, the informed approach already sampled all the points that lead to successful trials, and the next ones to be sampled are on the border of the useful region as they are most likely to produce further successful trials. Conversely, the random approach would need at least 6 more samples to cover the whole useful region.

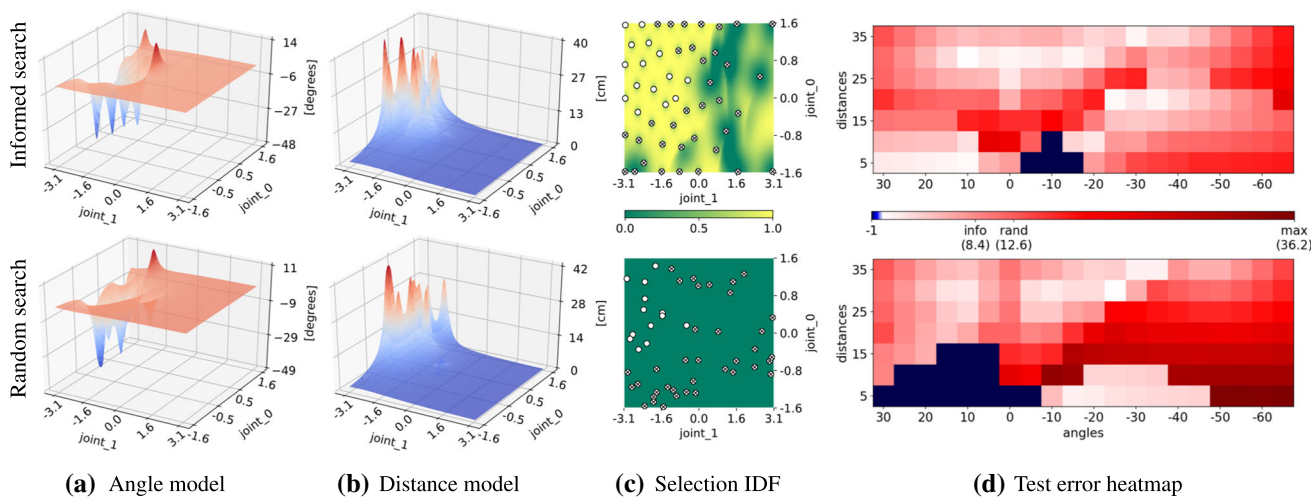
We further analyse the proposed approach by discussing the role of the hyperparameters and their influence on the performance. For this purpose, the two joint angle ranges are discretised with the resolution of 150 equally spaced values, which creates a much more complex parameter space of 22,500 combinations, over which the task models need to be defined. This finer discretisation makes the search more



**Fig. 6** Performance of the top performing hyperparameters on the test set after each trial, for 150 trials. The models that are better at lowering the Euclidean error are shown in **a** while those minimising the number of failed cases are shown in **b**, for both the informed and random approach (Color figure online)

difficult and emphasises the benefits of the informed search approach. To analyse the task model performance as the learning progresses, after each trial we evaluate the current task models on the test set. This is only possible in simulation, because testing on a real robot is intricate and time consuming. We perform the evaluation on 140 test target positions, with 20 values of the angle ranging from  $-65^\circ$  to  $30^\circ$  with  $5^\circ$  increments, and 7 values of the distance starting from 5 to 35 distance units from the origin, as shown in Fig. 4. The test error is defined as the Euclidean distance between the desired test target position, and the final resting position of the object the agent hits when executing the motion provided by the model. For the test positions which are complicated to reach and the learned task model cannot find the proper inverse solution, we label their outcomes as failed, exclude them from the average error calculation and present them separately. Figure 6 features plots showing the influence of different hyperparameter values, where PIDF covariance coefficients (cov), kernel functions (kernel) and the kernel's lengthscale ( $\sigma_l^2$ ) parameter are compared. The top plots show the mean Euclidean error, middle plots the error's standard deviation over the test set positions and the lower ones show the number of failed trials. The PIDF covariance values tested are 2, 5, 10 and 20 and they correspond to the width of the Gaussians representing failed trials in the PIDF. Making the covariance smaller (wider Gaussian) leads to faster migration of the trial selection to the regions of the parameter space

leading to successful trials. This hyperparameter does not affect the random approach as the random approach does not take into account the PIDF. Regarding the kernel type and its lengthscale hyperparameter, this affects the task model for both the proposed informed approach and the random trial generation. Smaller lengthscales imply that the influence of the training point on other points drops significantly as they are farther away from it, and thus the uncertainty about the model predictions for these other points raises quicker than with larger lengthscales. The actual effect is that the UIDF produces much narrower cones around trial points for small lengthscale values, which promotes sampling other points in the vicinity of the successful trials. In order to define the best performing model, we need to take into account the metrics presented in the plots in Fig. 6. Thus, models which do not produce failed tests and also have the minimal Euclidean error are required. Based on this criteria, both informed models from Fig. 6b, showed superior performance. However, for visualisation purposes, in Fig. 7 we show the learned task models and the final selection function of the best performing informed model together with its random counterpart from Fig. 6a at trial 50, having the same hyperparameters (RQ kernel with  $\sigma_l^2 = 0.01$  and  $\text{cov} = 5$ ). Moreover, in the same figure we present the evaluation of these models on 140 test target positions in the form of a heatmap together with the error mean. The failed test cases are shown in dark blue.



**Fig. 7** Comparison of the informed (top row) and random (bottom row) approach with the 2-link simulated agent at trial 50. The hyperparameters used are: RQ kernel with  $\sigma_f^2 = 0.01$  and  $\text{cov} = 5$ . Column **a** shows the learned angle task models, **b** distance models, **c** SIFD used to generate trials, where  $\otimes$  represent failed trials and  $\circ$  successful ones, with the colourmap indicating the probability of selecting the next trial.

Column **d** shows the performance on 140 test target positions with the colourmap indicating the Euclidean error for each of the test cases. The mean error (excluding unreachable positions) is indicated on the colourbar for both approaches. Notice that for the random approach there are 12 more unreachable positions marked in dark blue (Color figure online)

In Fig. 7 we can see how the SIFD was shaped by the penalisation function and the model uncertainty. The PIDF influenced the trial sampling to move away from the regions leading to failed trials, and focus on the region where the informative samples are, similarly to the previous experiment shown in Fig. 5. Furthermore, we can again see that most of the failed trials are in fact at the border between the failed and successful trial regions, as well as at the joint limits, which are the areas that need to be explored thoroughly.

Regarding the learned task models, we can see a clear distinction in the angle model that defines whether the puck will travel to the left (positive angle) or to the right (negative angle) and *joint\_0* influences this the most. The other, *joint\_1* mostly influences the intensity of the angle, i.e. how far will the puck move. This is possible because the joint space has a continuous nature which implies that the samples which are close in the parameter space produce similar performance. In the case of the learned angle model, it is easy to see the difference between what the informed and random approaches learned. While for the informed approach it is clear that the positive values of the *joint\_0* parameter lead to the positive angle values, within the random approach this relationship does not hold.

### 5.2 Experiment with 5-link agent

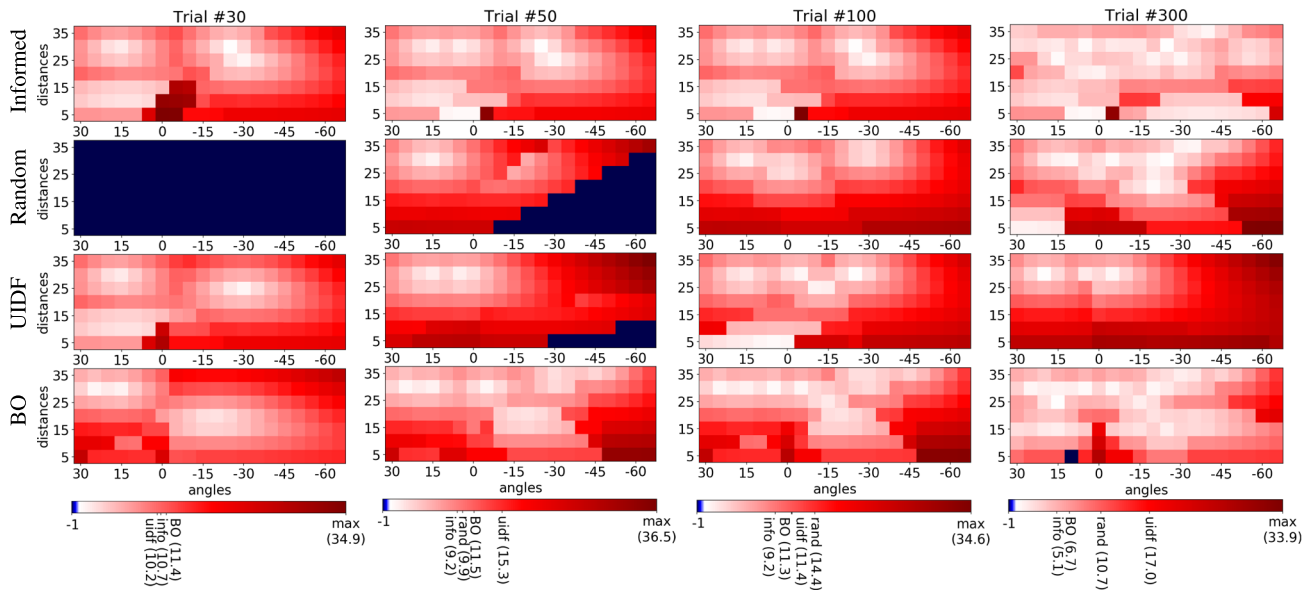
We further evaluate the performance of the proposed approach on a more complex task by using a 5-link agent as depicted in Fig. 4. The parameter space is 5 dimensional, discretised with 7 values per parameter dimension. The action execution

speed, base-link and inter-link joint ranges are as described in the previous section. Even though the discretisation is crude as mentioned in Sect. 3, we show the task is learned efficiently and shows good empirical performance. We evaluate the performance of the proposed informed search w.r.t. a random sampling approach. We also add an ablative comparison with the case where the PIDF is not included in the exploration component, but just the UIDF. UIDF uses the GPR model’s variance which can be considered proportional to the entropy measure, as the entropy of a Gaussian is calculated as  $\frac{1}{2} \ln(2\pi\sigma^2)$ , where the log preserves the monotonicity and the variance is always positive. In addition to this, we evaluate the performance of a modified version of the state-of-the art BO method presented in Englert and Toussaint (2016). Our problem formulation does not provide an objective function evaluations needed in BO, because the movement parameters are not model parameters which influence the final model performance. Instead of the model performance, we provide the decrease in model uncertainty as a measure of performance which is dependent on the movement parameter selection. This setting is then in line with our problem formulation and represents a good method for comparison. In Fig. 8a we show the mean (solid line) and standard deviation (shaded area) of the test performance error as well as number of failed test trials, based on several top performing hyperparameters. Below, in Fig. 8b, we show the heatmaps with errors for each test target position, at trials 30, 50, 100 and 300.

First significant observation, which was not obvious in the 2-link example, is that the random approach needs to sample almost 40 trials before obtaining a partially useful



(a) Performance plots



(b) Test error heatmaps

**Fig. 8** Performance of various hyperparameters on the test set after each trial, for 300 trials. In **a** the plots show the mean and the standard deviation of the test Euclidean error, averaged over the 5 best performing models of the (orange) informed approach (  $cov = 5$ , RQ kernel with  $\sigma_l^2 = 0.01$ ;  $cov = 10$ , RQ kernel with  $\sigma_l^2 = 0.001$ ;  $cov = 10$ , SE kernel with  $\sigma_l^2 = 0.001$ ;  $cov = 20$ , RQ kernel with  $\sigma_l^2 = 0.001$ ;  $cov = 20$ , SE kernel with  $\sigma_l^2 = 0.01$ ), (green) random approach runs over 5 different seeds, top 3 performing (red) UIDF-only exploration approaches ( all

using RQ kernel with  $\sigma_l^2 = 0.01$ ,  $\sigma_l^2 = 0.001$  and  $\sigma_l^2 = 0.001$ ) and (blue) the modified BO approach from Englert and Toussaint (2016) (showing all combinations of RQ and SE kernels with  $\sigma_l^2$  values: 0.01, 0.001, 0.0001). The heatmaps in **b** show actual test errors for each of the 140 test positions at trials 30, 50, 100 and 300, using the best performing instance of each of the models. The legend colormap show the average values for each approach (Color figure online)



**Fig. 9** An example of a successful trial executed during the training phase. The blue arrow points to the puck (Color figure online)

task model, while the informed approach needs less than 5 trials. It is important to emphasise that the parameter space contains  $7^5 = 16,807$  elements, which could cause inferior performance of the random approach. Secondly, we can see from the graph that even after 300 trials the informed approach demonstrates a declining trend in the test error mean and standard deviation, while the random approach stagnates. The uncertainty-only-based exploration approach finds a simple well-performing task model after only few trials, even slightly outperforming the informed approach. However, this approach is unstable and very sensitive to hyperparameter choice. This can be explained by UIDF being hardware-agnostic and not taking into account failed trials, but purely exploring the parameter space. The modified BO approach (Englert and Toussaint 2016), as expected, shows good and consistent performance. Also, it can be seen that it is not sensitive to hyperparameter change as the variance in performance for different settings is low. Unlike with our proposed approach, at the end of the learning phase when testing the models, there are still some test target positions which are not reachable by this model. By adding this new experiment, we compared to a method that enforces the feasibility of the parameters as a constraint in the cost function. As opposed to having an explicit constraint selecting only successful trial parameters, our proposed approach implements a soft (probabilistic) constraint through the PIDF, which still allows sampling of failed trials occasionally. This allows us to obtain datapoints at the borders of the feasible regions which are useful for the task model.

## 6 Robot experiment and analysis

To evaluate our proposed approach on a physical system we consider the problem of autonomously learning the ice-hockey puck-passing task with a bimanual robot.<sup>1</sup> We use robot DE NIRO (Design Engineering Natural Interaction Robot), shown in Fig. 1. It is a research platform for bimanual manipulation and mobile wheeled navigation, developed

<sup>1</sup> The video of the experiments is available at <https://sites.google.com/view/informedsearch>.

in our lab. The upper part is a modified Baxter robot from Rethink Robotics which is mounted on a mobile base via the extensible scissor-lift, allowing it to change the total height from 170 to 205 cm. The sensorised head includes a Microsoft Kinect 2.0 RGB-D camera with controllable pitch, used for object detection. DE NIRO learns to hit an ice hockey puck with a standard ice hockey stick, on a hardwood floor and pass it to a desired target position. We are using a right-handed stick which is 165 cm long and consists of two parts: the hollow metal stick shaft and the curved wooden blade fitted at the lower end. The standard (blue) puck weighs approximately 170 g. 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. 1). 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 cm in front of the blade. We monitor the movement effect on the puck using the head-mounted Kinect camera pointing downwards at a 45° angle. 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 transformation obtained via homography. All elements are interconnected using ROS (Quigley et al. 2009).

### 6.1 Experiment description

The puck-passing motion that the robot performs consists of a swing movement, making the contact with the puck and transferring the necessary impulse to move the puck to a certain location (as shown in Fig. 9). 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. The trajectory is generated by passing the chosen parameters (displacements) that define the goal position, to the built-in position controller implemented in the Baxter robot's API.

During the training phase, 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 which then slides it from the puck's fixed initial position to some final position which is encoded via polar coordinates  $\theta$  and  $L$ , as shown in Fig. 1. Such a trial is considered successful and contributes to the task models. If the swing misses the puck, the trial is failed. Other cases in which a trial is considered failed are defined in Sect. 4.2.

During the testing phase, the robot is presented with target positions that the puck needs to achieve, in order to evaluate the task model performance. The target is visually perceived as a green circle which is placed on the floor by the user (Fig. 1). Having received the target coordinates ( $\theta_d$  and  $L_d$ ), the robot needs to apply a proper swing action ( $\hat{x}$ ) that passes the puck to the target.

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. The movement parameters represent the amount of relative displacement with respect 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. The rest of the joints are not controlled directly. In this way the swing movement is parameterised and can be executed as a one-shot action. In the proposed setup, the parameters take discrete values from a predefined fixed set, equally spaced within the robot's workspace limits. The initial configuration of the robot arms and the ranges of the movement parameter values are assigned empirically. Even though the approach could be extended to autonomously detect the limits for the parameters, it is done manually in order to reduce the possibility of damaging the robot while exploring the edges of the parameter space. This implicitly reduces the number of learning trials, especially the failed ones. However, this parameter definition does not lead to any loss of generality of the framework and preserves the difficulty of the challenge. Although the robot's kinematic model is implicitly used for the movement execution, via the inverse kinematics in the position controller, this information is not used within our framework. The discretisation resolution of the parameter values inside the range is due to the numerical approach to obtaining the task models whose domain is the whole

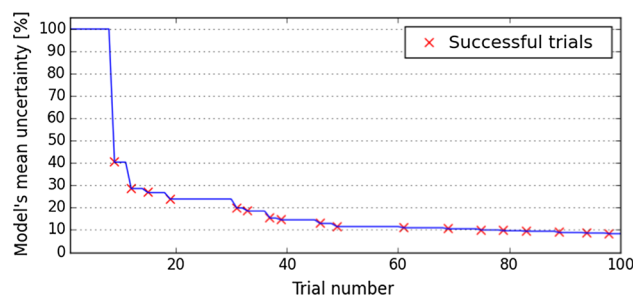
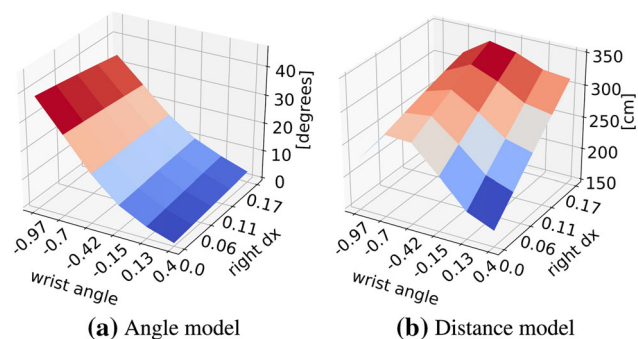


Fig. 10 Plot showing the decrease of GPR models' uncertainty with the number of trial evaluations

movement parameter space. The assigned parameter value sets are (in meters):  $l_x = \{-0.3, -0.2, -0.1, 0, 0.1\}$ ,  $l_y = \{-0.1, -0.05, 0, 0.05, 0.1\}$ ,  $r_x = \{0, 0.042, 0.085, 0.128, 0.17\}$ ,  $r_y = \{-0.1, 0.05, 0.2, 0.35, 0.5\}$ ,  $w = \{-0.97, -0.696, -0.422, -0.148, 0.126, 0.4\}$  and  $s = \{0.5, 0.625, 0.75, 0.875, 1.0\}$ . This produces a parameter space of size  $6 \times 5^5 = 18,750$ . 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 trials and 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 trial outcome amongst others. Figure 10 shows the uncertainty decrease over the sampling progress, and this can be interpreted as a learning curve showing how our task model decreases its uncertainty about its predictions. The overall training time including resetting is approximately 45 min. Figure 11a, b show the angle and distance models learned based on the datasamples from 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 in Fig. 11a shows how the wrist rotation angle greatly affects the final angle of the puck, for this particular swing configuration. This is in line with how professional hockey players manipulate the puck by rotating their wrist. The right hand displacement along the robot's  $x$ -axis does not contribute as much. The distance model in Fig. 11b 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



**Fig. 11** GPR task models learned during training based on successful trial data; learned task model for the **a** angle and **b** distance

value decreases, so does the influence of  $r_x$ . The range of motions that the puck achieves after training are from  $0^\circ$  to  $25^\circ$  for the angle, and the distance from 50 to 350 cm.

As a side note, one option could be to prune all the failed trials in simulation and perform only the successful ones on the robot. However, this would require having a precise kinematic model of the robot including the hockey stick and the passive joints which is not straightforward to model.

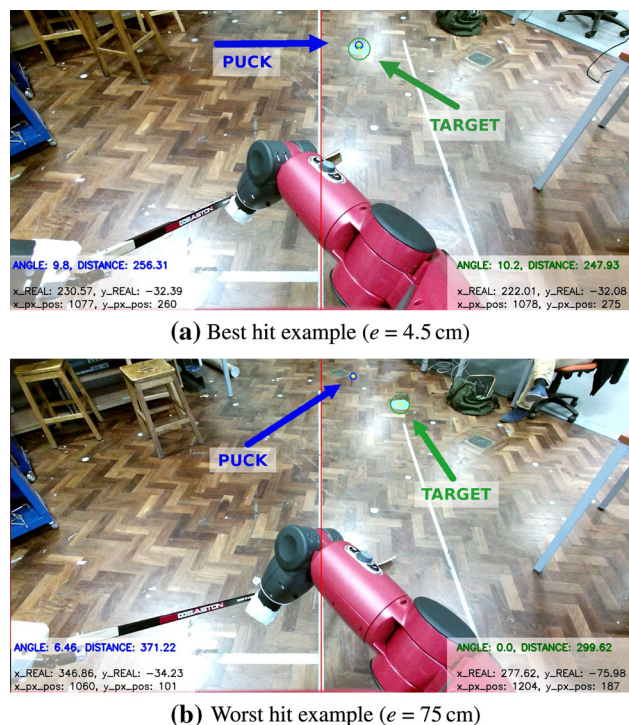
## 6.2 Experiment results and discussion

The essential interest here 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 assess the task model performance of our approach, we analyse the test execution accuracy, i.e. the ability to reach previously unseen targets.<sup>2</sup> During testing, the robot is presented with a target position (green circle as in Fig. 12) 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 Sect. 4.3. The final repeatability is the one achievable by the robot hardware ( $\pm 5$  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

<sup>2</sup> The code and experiment data will be made available on the project website upon publication.



**Fig. 12** View from the Kinect camera during the testing phase. The error  $e$  is the measured Euclidean distance between the puck and the target position, during the **a** best and **b** worst hit cases

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, based on the standard deviation, than the models produced by random search. 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 18,750 movement parameter combinations. Regarding the performance in the related work, in Daniel et al. (2013) the puck is sent to a target zone of 40 cm in width, while in Chebotar et al. (2017) there are only three fixed 25 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.

Secondly, we compare our approach to human-level performance. We asked 10 volunteers who had no previous ice hockey experience and 4 members of the college ice hockey club to participate, under the same settings as the robotic counterpart. The volunteers were placed at the same fixed position as the robot to maintain equal distance from the test targets, and the puck had the same starting position. No additional guidance was given regarding the stance, but they were shown in which regions of the stick they should place their grip in order to be comparable with the robot. The volunteers

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

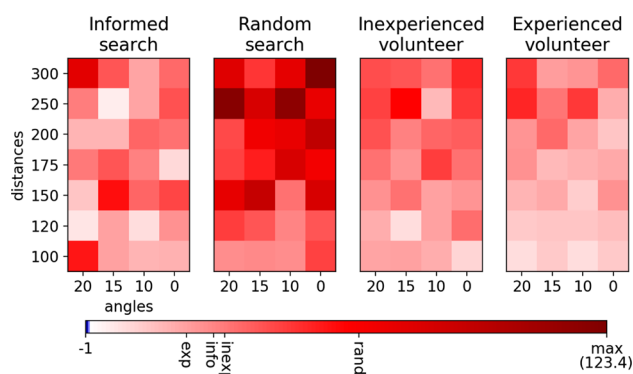
Movement generation method	Mean (cm)	STD (cm)
“Original” environment (blue puck, hardwood floor)		
Informed search	29.48	16.33
Random search	64.18	45.72
Inexperienced volunteers	32.16	27.82
Experienced volunteers	22.96	18.07
New environments		
“Original” model (blue puck, marble floor)	66.18	50.75
Re-trained 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

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. Experienced volunteers performed better than the robot and this can be explained with their domain knowledge. Even with a small sample size the within-group variance is low. By observing the heatmaps of these tests (Fig. 13) we can see the performance on each of the 28 test target position individually, averaged over all the candidates. It is noticeable that the human volunteers are more confident with targets that are closer, and to some extent the random approach as well. For the informed approach no such obvious pattern has emerged.

From qualitative observations we deduce that the inexperienced volunteers also need less time to acquire the basic skill level necessary to perform this task efficiently. This includes adjusting their grip and swing technique after a couple of trials, so it resembles that of experienced volunteers. We also 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 experienced volunteers and was also learned by the robot.

### 6.3 Task transfer

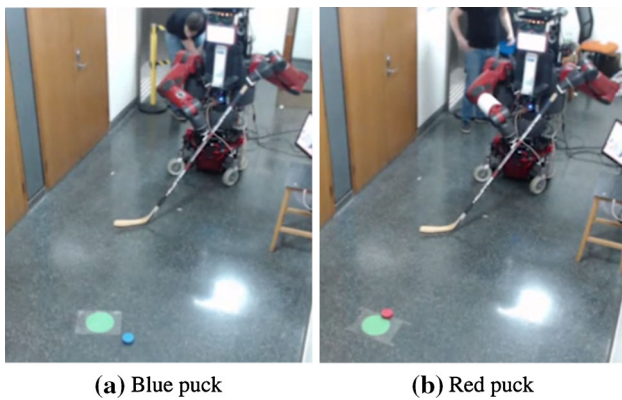
We demonstrate the task transfer aspect of the proposed framework by re-learning the task model for different environments. In this experiment we consider a task new, if it



**Fig. 13** Heatmaps showing the Euclidean distance error (in cm) over the test target positions, for tests in the upper part of Table 1

has a significantly different environment model, such as the object shape or weight and the floor friction. The main idea is that the trials generated are intrinsic to the robot hardware and are independent of the environment. Consequently, if the robot is placed on a different surface and given a ball instead of the puck, it would still generate the same trial movements. However, if the stick or other parts of the robot’s kinematic chain change significantly, this might not hold anymore. In that case, the training phase would have to be done from scratch as different kinematics generate different failed trial cases which need to be accounted for. Thus, if the kinematics are the same we just need to replicate the successful trials, and gather datapoints for the new environment-specific task model. Therefore the robot can adapt and perform the task in a new environment by executing only the set of 24 movement parameter vectors that generated successful trials in the “original” training session (standard puck on hardwood floor), not all 100 trials. The successful trials are independent of the environment and provide samples for the GPR task models. After executing the 24 trials and obtaining the trial outcomes, the actual model update is done in batch with the 24 datapoints, so the learning happens instantaneously.





**Fig. 14** Task transfer experimental setup on the marble floor, using the **a** standard ice-hockey puck made from vulcanised rubber (blue), and **b** a lighter puck made of wood (red) (Color figure online)

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 (approx. 80 g). The experimental setup for the task transfer is presented in Fig. 14. 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 can be explained due to the higher friction and thus decreased sensitivity, where changes in the movement parameters have a lower impact on the puck position. Therefore, 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 cm on average.

## 7 Conclusion and future directions

We have presented a probabilistic framework for learning the robot’s task and exploration models based solely on its sensory data, by means of informed search in the movement parameter space. The presented approach is validated in simulation and on a physical robot doing bimanual manipulation of an ice hockey stick in order to pass the puck to target positions. We compared our informed trial generation approach with random trial generation, as well as two more approaches in simulation, and showed superior performance of our proposed approach. In the robotic experiment, the robot learns

the task from scratch in approximately 45 min with an accuracy comparable to human-level performance and superior to similar experiments in the literature. Additionally, through our framework we demonstrated that the agent is capable of re-learning the task models in different new environments in significantly less time.

Future directions of the research include exploring the applicability of this approach to sequential tasks through the informed search in the policy or DMP parameter space. In particular with the emphasis on adapting the approach to continuous movement parameter spaces.

**Acknowledgements** The authors would like to thank Antoine Cully for useful comments that helped improve the manuscript. Nemanja Rakicevic is funded by the President’s PhD Scholarship from Imperial College London.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

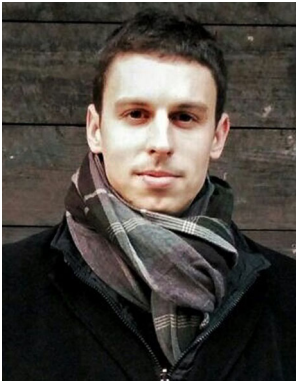
## References

- Aggarwal, C. C., Hinneburg, A., & Keim, D. A. (2001). On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*.
- Baranes, A., & Oudeyer, P. Y. (2013). Active learning of inverse models with intrinsically motivated goal exploration in robots. *Robotics and Autonomous Systems*, 61(1), 49–73.
- Berkenkamp, F., Schoellig, A. P., & Krause, A. (2016). Safe controller optimization for quadrotors with Gaussian processes. In *International conference on robotics and automation*.
- Bongard, J., & Lipson, H. (2005). Automatic synthesis of multiple internal models through active exploration. In *AAAI fall symposium: From reactive to anticipatory cognitive embodied systems*.
- Calandra, R., Seyfarth, A., Peters, J., & Deisenroth, M. P. (2016). Bayesian optimization for learning gaits under uncertainty. *Annals of Mathematics and Artificial Intelligence*, 76(1–2), 5–23.
- Chebatar, Y., Hausman, K., Zhang, M., Sukhatme, G., Schaal, S., & Levine, S. (2017). Combining model-based and model-free updates for trajectory-centric reinforcement learning. In *International conference on machine learning*.
- Cully, A., Clune, J., Tarapore, D., & Mouret, J. B. (2015). Robots that can adapt like animals. *Nature*, 521(7553), 503.
- Daniel, C., Neumann, G., Kroemer, O., & Peters, J. (2013). Learning sequential motor tasks. In *International conference on robotics and automation*.
- Daniel, C., Viering, M., Metz, J., Kroemer, O., & Peters, J. (2014). Active reward learning. In *Robotics: Science and systems*.
- Deisenroth, M. P., Fox, D., & Rasmussen, C. E. (2015). Gaussian processes for data-efficient learning in robotics and control. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(2), 408–423.
- Demiris, Y., & Dearden, A. (2005). From motor babbling to hierarchical learning by imitation: A robot developmental pathway. *Workshop on epigenetic robotics*.

- Dima, C., Hebert, M., & Stentz, A. (2004). Enabling learning from large datasets: Applying active learning to mobile robotics. In *International conference on robotics and automation*.
- Englert, P., & Toussaint, M. (2016). Combined optimization and reinforcement learning for manipulation skills. In *Robotics: Science and systems*.
- Forte, D., Gams, A., Morimoto, J., & Ude, A. (2012). On-line motion synthesis and adaptation using a trajectory database. *Robotics and Autonomous Systems*, 60, 1327–1339.
- Gelbart, M. A., Snoek, J., & Adams, R. P. (2014). Bayesian optimization with unknown constraints. arXiv preprint [arXiv:1403.5607](https://arxiv.org/abs/1403.5607).
- Hansen, N., Müller, S. D., & Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation*, 11(1), 1–18.
- Heidrich-Meisner, V., & Igel, C. (2009). Neuroevolution strategies for episodic reinforcement learning. *Journal of Algorithms*, 64(4), 152–168.
- Hennig, P., & Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research*, 13, 1809–1837.
- Hernández-Lobato, J. M., Hoffman, M. W., & Ghahramani, Z. (2014). Predictive entropy search for efficient global optimization of black-box functions. In *Advances in neural information processing systems*.
- Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., & Schaal, S. (2013). Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 25(2), 328–373.
- Kahn, G., Villaflor, A., Pong, V., Abbeel, P., & Levine, S. (2017). Uncertainty-aware reinforcement learning for collision avoidance. arXiv preprint [arXiv:1702.01182](https://arxiv.org/abs/1702.01182).
- Kapoor, A., Grauman, K., Urtasun, R., & Darrell, T. (2007). Active learning with Gaussian processes for object categorization. In *International conference on computer vision*.
- Kormushev, P., Demiris, Y., & Caldwell, D. G. (2015). Kinematic-free position control of a 2-dof planar robot arm. In *International conference on intelligent robots and systems*.
- Kroemer, O., Detry, R., Piater, J., & Peters, J. (2010). Combining active learning and reactive control for robot grasping. *Robotics and Autonomous Systems*, 58(9), 1105–1116.
- Kulick, J., Otte, S., & Toussaint, M. (2015). Active exploration of joint dependency structures. In *International conference on robotics and automation*.
- Kushner, H. J. (1964). A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *Journal of Basic Engineering*, 86(1), 97–106.
- Lehman, J., & Stanley, K. O. (2011a). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation*, 19(2), 189–223.
- Lehman, J., & Stanley, K. O. (2011b). Evolving a diversity of virtual creatures through novelty search and local competition. In *Conference on genetic and evolutionary computation*.
- Lewis, D. D., & Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Conference on Research and development in information retrieval*.
- Lizotte, D. J., Wang, T., Bowling, M. H., & Schuurmans, D. (2007). Automatic gait optimization with Gaussian process regression. In *International joint conference on artificial intelligence*.
- Lopes, M., Lang, T., Toussaint, M., & Oudeyer, P. Y. (2012). Exploration in model-based reinforcement learning by empirically estimating learning progress. In *Advances in neural information processing systems*.
- Martinez-Cantin, R., de Freitas, N., Brochu, E., Castellanos, J., & Doucet, A. (2009). A bayesian exploration-exploitation approach for optimal online sensing and planning with a visually guided mobile robot. *Autonomous Robots*, 27, 93–103.
- Moćkus, J. (1975). On bayesian methods for seeking the extremum. In *Optimization techniques IFIP technical conference*.
- Mockus, J. (1994). Application of bayesian approach to numerical methods of global and stochastic optimization. *Journal of Global Optimization*, 4(4), 347–365.
- Mouret, J. B., & Clune, J. (2015). Illuminating search spaces by mapping elites. arXiv preprint [arXiv:1504.04909](https://arxiv.org/abs/1504.04909).
- Nemec, B., Ude, A., et al. (2011). Exploiting previous experience to constrain robot sensorimotor learning. In *International conference on humanoid robots*.
- Newell, K. M. (1991). Motor skill acquisition. *Annual Review of Psychology*, 42(1), 213–237.
- Nguyen-Tuong, D., Seeger, M., & Peters, J. (2009). Model learning with local gaussian process regression. *Advanced Robotics*, 23(15), 2015–2034.
- Otte, S., Kulick, J., Toussaint, M., & Brock, O. (2014). Entropy-based strategies for physical exploration of the environment's degrees of freedom. In *International conference on intelligent robots and systems*.
- Pathak, D., Agrawal, P., Efros, A. A., & Darrell, T. (2017). Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*.
- Plappert, M., Houthoofd, R., Dhariwal, P., Sidor, S., Chen, R. Y., Chen, X., Asfour, T., Abbeel, P., & Andrychowicz, M. (2017). Parameter space noise for exploration. arXiv preprint [arXiv:1706.01905](https://arxiv.org/abs/1706.01905).
- Pugh, J. K., Soros, L. B., & Stanley, K. O. (2016). Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3, 40.
- Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., & Ng, A. Y. (2009). Ros: An open-source robot operating system. In *ICRA workshop on open source software*.
- Rasmussen, C. E., & Williams, C. K. (2006). *Gaussian processes for machine learning*. Cambridge, MA: MIT Press.
- Reinhart, R. F. (2017). Autonomous exploration of motor skills by skill babbling. *Autonomous Robots*, 41(7), 1521–1537.
- Rodrigues, F., Pereira, F., & Ribeiro, B. (2014). Gaussian process classification and active learning with multiple annotators. In *International conference on machine learning*.
- Rolf, M., Steil, J. J., & Gienger, M. (2010). Goal babbling permits direct learning of inverse kinematics. *Transactions on Autonomous Mental Development*, 2, 216–229.
- Rückstieß, T., Sehnke, F., Schaul, T., Wierstra, D., Sun, Y., & Schmidhuber, J. (2010). Exploring parameter space in reinforcement learning. *Paladyn*, 1, 14–24.
- Salimans, T., Ho, J., Chen, X., & Sutskever, I. (2017). Evolution strategies as a scalable alternative to reinforcement learning. arXiv preprint [arXiv:1703.03864](https://arxiv.org/abs/1703.03864).
- Santner, T. J., Williams, B. J., & Notz, W. I. (2013). *The design and analysis of computer experiments*. Berlin: Springer.
- Schreiter, J., Nguyen-Tuong, D., Eberts, M., Bischoff, B., Markert, H., & Toussaint, M. (2015). Safe exploration for active learning with Gaussian processes. In *Joint European conference on machine learning and knowledge discovery in databases*.
- Seo, S., Wallat, M., Graepel, T., & Obermayer, K. (2000). Gaussian process regression: Active data selection and test point rejection. In *International joint conference on neural networks*.
- Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 6, 1–14.
- Seung, H. S., Opper, M., & Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on computational learning theory*.
- Srinivas, N., Krause, A., Kakade, S., & Seeger, M. (2010). Gaussian process optimization in the bandit setting: No regret and experimental design. In *International conference on machine learning*.

- Tesch, M., Schneider, J., & Choset, H. (2011). Using response surfaces and expected improvement to optimize snake robot gait parameters. In *International conference on intelligent robots and systems*.
- Thrun, S. B., & Möller, K. (1992). Active exploration in dynamic environments. In *Advances in neural information processing systems*.
- Todorov, E., Erez, T., & Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *International conference on intelligent robots and systems*.
- Wierstra, D., Schaul, T., Glasmachers, T., Sun, Y., Peters, J., & Schmidhuber, J. (2014). Natural evolution strategies. *Journal of Machine Learning Research*, 15, 949–980.

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Nemanja Rakicevic** is a Ph.D. student in the Robot Intelligence Lab, Imperial College London. He holds a B.Sc. degree in Mechatronics from the Faculty of Technical Sciences, University of Novi Sad and a M.Sc. degree from the double-degree EMARO (European Masters on Advanced Robotics) program. His research interests lie within artificial intelligence and robotics, more specifically on uncertainty-based efficient exploration approaches for supervised and reinforcement learning meth-

ods, applied to robotic task learning.



**Dr. Petar Kormushev** is Director of the Robot Intelligence Lab and Lecturer in Robotics at the Dyson School of Design Engineering, Imperial College London, UK. Previously, he was Research Team Leader of the Robot Learning and Interaction Lab at the Department of Advanced Robotics, Italian Institute of Technology (IIT) in Genoa, Italy. In 2009 he obtained PhD in Computational Intelligence from Tokyo Institute of Technology, Japan. He holds two M.Sc. degrees in Artificial

Intelligence and Medical Informatics, and a B.Sc. degree in Computer Science from Sofia University, Bulgaria. Dr. Kormushev was a co-chair of the IEEE Technical Committee on Robot Learning from 2012–2015. He is the recipient of the 2013 John Atanasoff award by the President of Bulgaria, awarded for scientific achievements in the field of ICT. His research interests include machine learning and robot learning algorithms, especially reinforcement learning for intelligent robot behaviour.