Hazara, Murtaza; Kyrki, Ville

Transferring Generalizable Motor Primitives From Simulation to Real World

Published in:
IEEE Robotics and Automation Letters

DOI:
10.1109/LRA.2019.2900768

Published: 01/04/2019

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Hazara, M., & Kyrki, V. (2019). Transferring Generalizable Motor Primitives From Simulation to Real World. *IEEE Robotics and Automation Letters*, *4*(2), 2172-2179. https://doi.org/10.1109/LRA.2019.2900768

Transferring Generalizable Motor Primitives From Simulation to Real World

Murtaza Hazara and Ville Kyrki

Abstract—**Reinforcement learning provides robots with an autonomous learning framework where a skill can be learned by exploration. Exploration in real world is, however, inherently unsafe and time consuming, and causes wear and tear. To address these, learning policies in simulation and then transferring them to physical systems has been proposed. In this letter, we propose a novel sample-efficient transfer approach, which is agnostic to the dynamics of a simulated system and combines it with incremental learning. Instead of transferring a single control policy, we transfer a generalizable contextual policy generated in simulation using one or few samples from real world to a target global model, which can generate policies across parameterized real-world situations. We studied the generalization capability of the incremental transfer framework using MuJoCo physics engine and KUKA LBR 4+. Experiments with ball-in-a-cup and basketball tasks demonstrated that the target model improved the generalization capability beyond the direct use of the source model indicating the effectiveness of the proposed framework. Experiments also indicated that the transfer capability depends on the generalization capability of the corresponding source model, similarity between source and target environment, and number of samples used for transferring.**

Index Terms—**Learning and Adaptive Systems, Model Learning for Control.**

I. Introduction

REINFORCEMENT learning (RL) can provide robots with the capability to learn a skill autonomously. However, RL performs learning by exploration which is sometimes dangerous to execute on a physical system; causes wear and tear on the robot; and, is time consuming. In order to minimize the interaction with the physical robot, one can instead apply RL for learning a policy in the simulation. However, policies learned in the simulation often cannot be directly deployed in the real world because of discrepancies between dynamics of simulation versus real world. To address this problem, transfer learning has been proposed. The main focus of previous research has been on transferring a single control policy and robustifying it against the uncertainties of real world dynamics (e.g. friction dynamics) or adapting the dynamics of a task. However, few researchers have studied the generalization of the transferred policy to new situations.

Incremental learning provides another way of decreasing the amount of real-world exploration. When the context of learning can be controlled, a system can be exposed to increasingly difficult situations to build a contextual model that generalizes over those situations. For example, a global parametric model of a skill can be combined with RL to construct a database of motion primitives (MPs) incrementally [1]. When encountering a new situation, the global model predicts MPs by extracting the underlying regularities from the database, while RL optimizes the predicted MPs which have failed to re-enact the task successfully.

The main contribution of this letter is an incremental transfer framework which combines the incremental learning of a source task with transfer to the target environment. The proposed transfer approach is agnostic to the dynamics model of a system. Incremental learning is applied first in simulation to build a contextual model that captures the underlying regularities of the simulated (source) task with respect to measurable task parameters. Then, the contextual model is transferred to the real-world (target) environment using one or few real-world samples. The resulting global model can generate policies for any task parameter value in the real world. In other words, instead of transferring a single source policy optimized for only one specific situation, we transfer a generalizable model from simulation to real world. This allows the transfer approach to accommodate both unmodeled dynamics and generalization to new situations.

The proposed method is experimentally evaluated in basketball and ball-in-a-cup tasks to assess its generalization capability. Results show that the proposed incremental transfer framework can successfully transfer the source contextual model even with one sample from the real world. Furthermore, the experiments indicate that the transfer capability depends on the similarity between target and source environment, generalization capability of the source model, and the number of the samples from target environment used for transferring.

II. Related Work

Transfer learning (TL) has been extensively studied in different contexts [2]. However, in this section, we focus on simulation to real-world transfer. After that, we briefly review generalizable models.

Manuscript received September 10, 2018; accepted January 25, 2019. Date of publication February 21, 2019; date of current version March 7, 2019. This letter was recommended for publication by Associate Editor T. Inamura and Editor D. Lee upon evaluation of the reviewers' comments. This work was supported by Academy of Finland, decision 268580. *(Corresponding author: Murtaza Hazara.)*

The authors are with the Department of Electrical Engineering and Automation, Aalto University, Aalto 11000, Finland (e-mail: murtaza.hazara@aalto.fi; ville.kyrki@aalto.fi).

This letter has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Digital Object Identifier 10.1109/LRA.2019.2900768

A. Transfer Learning

One of the main assumptions in transferring control policies is that the source and target environment share the same characteristics. In this case, policies learned in the source is useful in the target. For example, in [3] utility of past policies is estimated using policy reuse to bias the exploration process of a simulated grid-based navigational domain. However, discrepancies between the simulation and real world makes it necessary to adapt the source policies with respect to the target.

Domain randomization has been proposed as a successful approach for adapting source policies to the target. In [4], visual features such as lighting and texture have been randomized in the source environment to robustify the transferred policies against the uncertainties of the visual input. However, these methods do not account for the uncertainties in the dynamics of a system. On the other hand, in [5], [6], they optimize source policies on an ensemble of perturbed dynamics models to robustify against the dynamics model uncertainties. In [7], [8], they randomize friction dynamics for learning policies which have been shown to be robust against the uncertainties of real-world friction dynamics. However, they have considered only the dynamics of a system which can be parametrized such as friction. In other words, it does not account for unmodeled dynamics. Moreover, the range of parameters (e.g. friction coefficient) need to be determined in advance of transferring. Above all, it does not account for generalizing transferred policies to new task parameters.

Instead of adapting the dynamics of the system, in [9], dynamics of the task is adapted using a data-efficient model-based approach [10], [11]. Even though, it can accounts for unmodeled dynamics of the system, it can only learn a single controller for one specific situation.

On the other hand, we propose incremental transferring of a global model which is generalizable to new situations. In other words, the transferred generalizable model generates target policies for any task parameter in the real world. In fact, our transfer approach is sample efficient, data-driven and agnostic to the underlying dynamics model of a system. Therefore, our transfer approach can accommodate for unmodeled disturbances.

B. Generalizable Models

Researchers have shown interest in generalizable motor primitives (MPs) to new situations using contextual policy search [12], [13], or a skill model such as Gaussian process regression [14] and locally weighted regression [15]. On the other hand, non-linear global models (GPDMP) with linear computational complexity [16] have been shown to outperform local models and the global linear models [17], [18] with respect to their extrapolation capability. Furthermore, GPDMP has provided the uncertainty of MPs for guiding the exploration process of RL in an incremental learning framework outperforming the covariance matrix adaptation [19]. The main contribution of this letter is to propose incremental transfer and combine it with incremental learning for transferring GPDMP from simulation to real world. To our best knowledge, this is the first letter proposing incremental transfer of a global model.

III. Method

We begin this section by describing the global parametric dynamic movement primitives (GPDMPs) method which extends dynamic movement primitives (DMPs) to varying contexts. We then describe the applied model selection approach and explain how a database of primitives can be learned incrementally in simulation. Finally, we describe the approach to transfer a GPDMP model learned in simulation to a physical system in an incremental fashion.

A. Global Parametric Dynamic Movement Primitives

DMPs [20] are a common policy representation for trajectory-based motions. The motion is encoded as a spring-damper system perturbed by a time variant forcing function. The forcing function can be written as

$$f(z;\mathrm{w}) = \mathrm{w}^T \mathrm{g}(z), \tag{1}$$

where g is a time-parameterized kernel vector and w the policy parameters. the n-th element of the kernel vector

$$[\mathrm{g}]_n = \frac{\psi^n(z)z}{\sum_{i=1}^{N}\psi^i(z)}(g - x_0) \tag{2}$$

is determined by a normalized basis function $\psi^n(z)$ multiplied by the phase variable z and the scaling factor $(g - x_0)$ allowing for the spatial scaling of the resulting trajectory.

Using DMPs, a task can be imitated from a human demonstration; however, the reproduced task cannot be adapted to different environment conditions. To overcome this limitation, we have integrated a parametric model to DMPs capturing the variability of a task from multiple demonstrations. We transform the basic forcing function (1) into a parametric forcing function [16]

$$f(z,\mathrm{l};\mathrm{w}) = \mathrm{w}(\mathrm{l})^T \mathrm{g}(z), \tag{3}$$

where the kernel weight vector w is parametrized using a parameter vector l of measurable environment factors.

We model the dependency of the weights with respect to parameters as a linear combination of J basis vectors v_i with coefficients depending on parameters in a non-linear fashion [16]

$$\mathrm{w}(\mathrm{l}) = \sum_{i=0}^{J-1}\phi_i(\mathrm{l})\mathrm{v}_i = \mathbf{V}^T\phi(\mathrm{l}), \tag{4}$$

where $\mathbf{V}$ is a $J \times N$ matrix of parameters with N referring to the number of kernels g, and J denotes model complexity. The basis function $\phi(\mathrm{l})$ is a J dimensional column vector with elements $\phi_j(\mathrm{l})$. For example, the non-linear basis $\phi_j(\mathrm{l})$ for a polynomial model in one parameter is $\phi_j(l) = l^j$. The formulation captures linear models such as [18] as a special case.

For a chosen non-linear basis (known functions ϕ_i), the basis vectors can be determined as a least squares problem

$$\arg\min_{\mathbf{V}} \sum_{k=1}^{K} \|\mathrm{w}(\mathrm{l}_k) - \mathrm{w}_k\|_2 , \tag{5}$$

where K represents the number of training sample represented by w_k which denotes target weights determined from human demonstration [20] or learned using policy search [21]. In either

case, reproducing an imitated task using $\mathbf{w}_k$ should lead to a successful performance in an environment parametrized by $\mathbf{l}_k$. The solution of (5) is

$$\hat{\mathbf{V}} = \mathbf{\Phi}^{+}\,\mathbf{W}, \tag{6}$$

where

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{l}_1) & \phi_1(\mathbf{l}_1) & \dots & \phi_{J-1}(\mathbf{l}_1) \\ \phi_0(\mathbf{l}_2) & \phi_1(\mathbf{l}_2) & \dots & \phi_{J-1}(\mathbf{l}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{l}_K) & \phi_1(\mathbf{l}_K) & \dots & \phi_{J-1}(\mathbf{l}_K) \end{bmatrix} \tag{7}$$

is the design matrix and $\mathbf{\Phi}^{+}$ denotes it pseudoinverse and

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_K^T \end{bmatrix} \tag{8}$$

is the target matrix.

B. Model Selection

The best generalization using a parametric regression model can be achieved by choosing an optimal order of complexity J for the model, which is addressed in model selection. We have selected a penalized log-likelihood model selection method which chooses a model by minimizing

$$B_M = tr((\mathbf{W} - \mathbf{\Phi}\hat{\mathbf{V}})^T(\mathbf{W} - \mathbf{\Phi}\hat{\mathbf{V}})\mathbf{\Sigma}_M^{-1}) + J\log K \tag{9}$$

where $\mathbf{\Sigma}_M$ represents a constant covariance matrix which needs to be determined prior to the model selection process (see [16] for more detail).

C. Incremental Learning of the Source Model

We utilize the incremental learning framework proposed in [1] to construct a database of MPs in a source (simulation) environment, denoting the database DB_s. The database is used to estimate a corresponding global parametric model $GPDMP_s$. Whenever the application of $GPDMP_s$ fails to reproduce a skill successfully, we apply the PoWER [21] policy search to optimize the primitive. The optimized primitive is then added to DB_s and the global model $GPDMP_s$ is re-estimated.

To perform exploration in the policy search, we use pre-structured covariance matrix [16] for the Gaussian distribution from which a correlated smooth noise vector is sampled providing safe exploration. Our complete setup of the PoWER is described in [16].

When re-estimating the $GPDMP_s$, the model selection method (9) is used. This may cause the model order J to change when more data is acquired. In particular, the model order can increase to allow describing more complex relationships when enough evidence is gathered.

In a nutshell, $GPDMP_s$ provides policy search a good starting policy for a new situation; RL, on the other hand, optimizes the predicted MPs in a new situation; thus, providing $GPDMP_s$ with more training samples. In this way, DB_s is built incrementally and in an online manner.

D. Incremental Transfer of the Source Model

Real world cannot be simulated perfectly. Thus, the policies learned in the simulation need to be adjusted to the real world. This problem is addressed in transfer learning. The policies can be transferred independently for every single task parameter. Although the policy learned in simulation is a good starting policy for transfer learning, this requires excessive robot interaction with the real world because learning is necessary for each task parameter. Instead, we transfer the generalizable model which can generate target policy for any task parameter, thus speeding up the transfer and minimizing the interaction with the real world.

Transferring a source global model $GPDMP_s$ can be performed incrementally (see Algorithm 1). The basis vectors of the transferred policy $\mathbf{V}_t$ are initialized from the source model in lines 1-7. Task parameter value is chosen in line 9, depending on application. For a one dimensional task parameter, an initial value is selected and then decreased or increased by a chosen amount (see Section IV-C for an example). Next, policy parameters are determined using the target basis vectors (line 10) and the policy is re-enacted in the real world. If the re-enacted policy is unsuccessful, it is optimized using reinforcement learning. The optimized policy parameters are then added to a database of target MPs DB_t (line 13).

Now that DB_t contains a MP for one task parameter, we can transfer $GPDMP_s$. We assume that the underlying regularities of the real-world dynamics are similar to the simulated dynamics since the tasks are the same. Therefore, we assume that the transferred global model $GPDMP_t$ has the same model complexity as the source model $GPDMP_s$, while it also matches MPs in the DB_t, collected in matrix $\mathbf{W}_t$. In order to calculate the transferred basis vectors $\mathbf{V}_t$, one needs to solve a system of linear equations:

$$\mathbf{W}_t = \mathbf{\Phi}_t \mathbf{V}_t, \tag{10}$$

where the target design matrix $\mathbf{\Phi}_t$ is defined in (7).

Initially, the system may be under-determined, since $\mathbf{W}_t$ contains only one sample. The set of solutions to (10) is

$$\mathbf{V}_t = \mathbf{V}_0 + \mathbf{V}_N \mathbf{\Lambda}_N, \tag{11}$$

where

$$\mathbf{V}_0 = \mathbf{\Phi}_t^{+}\,\mathbf{W}_t \tag{12}$$

and where each column of

$$\mathbf{V}_N = \text{null}(\mathbf{\Phi}_t) \tag{13}$$

denotes a basis vector for the null space of the target design matrix $\mathbf{\Phi}_t$.

The coefficient matrix $\mathbf{\Lambda}_N$ can be determined by minimizing the distance between the target basis vectors $\mathbf{V}_t$ and the source basis vectors $\mathbf{V}_s$

$$\min \|(\mathbf{V}_0 - \mathbf{V}_s) + \mathbf{V}_N \mathbf{\Lambda}_N\|^2, \tag{14}$$

where the basis vectors of the simulation is computed using

$$\mathbf{V}_s = \mathbf{\Phi}_s^{+}\, \mathbf{W}_s. \tag{15}$$

This case, however, gives equal attention to all dimensions of the basis vector space. This may lead to a transferred contextual model which is noticeably different from the contextual model in simulation because different basis functions may have effects in different scales. Therefore, instead of minimizing the sum of squares (14), we minimize the weighted sum of squares

$$\min \|(\mathbf{V}_0 - \mathbf{V}_s) + \mathbf{V}_N \mathbf{\Lambda}_N\|_{\mathbf{F}}^2, \tag{16}$$

where $\mathbf{F}$ is a diagonal weight matrix reflecting scales similar to a covariance matrix of a Gaussian prior. The solution to (16) is

$$\mathbf{\Lambda}_N = \left(\mathbf{V}_N^T \mathbf{F} \mathbf{V}_N\right)^{-1} \mathbf{V}_N^T \mathbf{F} \left(\mathbf{V}_s - \mathbf{V}_0\right). \tag{17}$$

For the polynomial base functions, we select the diagonal terms to be of the form $F_{ii} = c_F^{i-1}$. In this case, the coefficients associated with higher order terms in the global parametric model will contribute more to the adjustment. Consequently, the transferred contextual model will be more similar to the source contextual model, increasing the safety of generalization to new situations.

Algorithm 1: Incremental transfer of source model $GPDMP_s$.

Input: $DB_s = \{\{\mathrm{l}_i; \mathbf{w}_{\mathrm{l}_i}\} \mid 1 \leq i \leq K_s\}$
Output: transferred global model, that is $GPDMP_t$ with corresponding parameters, that are $\mathbf{V}_t$.
Initialisation :
$K_s \leftarrow |DB_s|$
$J_s \leftarrow \underset{J \in 1 \ldots K_s}{\operatorname{argmin}} \{B_M\}$, where B_M is computed using (9)
$\mathbf{W}_s \leftarrow [(\mathbf{w}_{\mathrm{l}_1})^T; \ldots (\mathbf{w}_{\mathrm{l}_{K_s}})^T] \in \mathbb{R}^{K_s \times N}$
$\mathbf{\Phi}_s \leftarrow [(\phi(\mathrm{l}_1))^T; \ldots (\phi(\mathrm{l}_{K_s}))^T] \in \mathbb{R}^{K_s \times J_s}$
$\mathbf{V}_s \leftarrow (\mathbf{\Phi}_s^T \mathbf{\Phi}_s)^{-1} \mathbf{\Phi}_s^T \mathbf{W}_s$
$\mathbf{V}_t \leftarrow \mathbf{V}_s$
$DB_t \leftarrow \{\}$
repeat
  select a task parameter l
  $\mathbf{w}_{\mathrm{l}} \leftarrow (\mathbf{V}_t)^T \phi(\mathrm{l})$
  if $\mathbf{w}_{\mathrm{l}}$ is not successful for task parameter l **then**
    optimize $\mathbf{w}_{\mathrm{l}}$ for l using policy search RL
    $DB_t \leftarrow DB_t \cup \{\mathrm{l}; \mathbf{w}_{\mathrm{l}}\}$
    $K_t \leftarrow |DB_t|$
    $\mathbf{W}_t \leftarrow [(\mathbf{w}_{\mathrm{l}_1})^T; \ldots (\mathbf{w}_{\mathrm{l}_{K_t}})^T] \in \mathbb{R}^{K_t \times N}$
    $J_t \leftarrow \underset{J \in 1 \ldots K_t}{\operatorname{argmin}} \{B_M\}$, B_M is computed using (9
    $\mathbf{\Phi}_t \leftarrow [(\phi(\mathrm{l}_1))^T; \ldots (\phi(\mathrm{l}_{K_t}))^T] \in \mathbb{R}^{K_t \times J_s}$ (see (7))
    $\mathbf{V}_0 \leftarrow (\mathbf{\Phi}_t^T \mathbf{\Phi}_t)^{-1} \mathbf{\Phi}_t^T \mathbf{W}_t$
    $\mathbf{V}_N \leftarrow \mathrm{null}(\mathbf{\Phi}_t)$
    $\mathbf{F} \leftarrow diag(c_F^0 \ldots c_F^{J_s - 1})$
    $\mathbf{\Lambda}_N \leftarrow \left(\mathbf{V}_N^T \mathbf{F} \mathbf{V}_N\right)^{-1} \mathbf{V}_N^T \mathbf{F} \left(\mathbf{V}_s - \mathbf{V}_0\right)$
    $\mathbf{V}_t \leftarrow \mathbf{V}_0 + \mathbf{V}_N \mathbf{\Lambda}_N$
    Evaluate the generalization capability of the transferred model, that is $g_t(GPDMP_t)$
  end if
until The required generalization capability $g_t(GPDMP_t)$ is achieved and while $J_t < J_s$
return $\mathbf{V}_t$ and DB_t

IV. Experimental Evaluation

We studied experimentally the generalization performance of the proposed incremental transfer framework using ball-in-a-cup and basketball tasks on KUKA LBR 4+. We utilized DMPs as the policy encoding since it provides us with a low-dimensional policy representation which is a less data-demanding model than high-dimensional policy representations such as deep RL. In this section, we first explain the tasks and the incremental transfer process. The experiments study five hypotheses on the relationship between generalization capability and factors such as number of target samples and similarity of source and target environments.

A. Ball-in-a-Cup Task

The ball-in-a-cup game consists of a cup, a string, and a ball; the ball is attached to the cup by the string (see Fig. 2). The objective of the game is to get the ball in the cup by moving the cup. We chose the ball-in-a-cup game because variation in the environment can be generated by changing the string length. The string length is observable and easy to evaluate, thus providing a suitable task parameter. Nevertheless, changing the length requires a complex change in the motion to succeed in the game. Hence, the generalization capability of a parametric LfD model can be easily assessed using this game. Similar to our previous set-up in [16], the trajectories along y and z axes were encoded using separate DMPs. Utilizing 20 kernels per DMP, in total $N = 40$ parameters are needed to describe the motion model for a single task parameter value. In order to vary the similarity between source (simulated) and target (real-world) environment, air density was changed in simulation from the real world value of 1.2 $\frac{\mathrm{kg}}{\mathrm{m}^3}$. For this task, five values of air density were used: 1.2, 1.5, 2.0, 2.5, and 3.0 $\frac{\mathrm{kg}}{\mathrm{m}^3}$ for environments $i = 1, \ldots, 5$ respectively. Changing the air density causes air drag, introducing viscous friction and generating differences between the dynamics of environments.

To compare to a common sim-to-real transfer approach, we implemented domain randomization (DR) [7] for the ball-in-cup task. We optimized the policy over randomized environment dynamics (air density). Results showed that the optimized policy was successful only in one of five environments. The results were consistent over different task parameters (string length). Thus, DR as a sim-to-real transfer method was insufficient to account for environment differences in our experimental scenario.

B. Basketball Task

The basketball game consists of a ball holder, a basket, and a ball; the holder is attached to the end-effector of KUKA LWR 4+ (see Fig. 1) and the basket is set at a certain distance from the robot. The objective of the game is to throw the ball at the

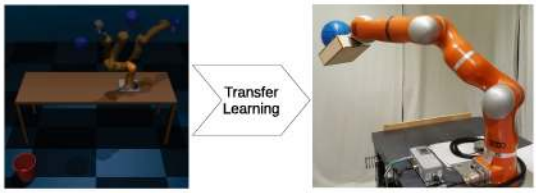

Fig. 1. Transferring basket ball skill from a simulated environment in MuJoCo to KUKA LBR 4+.

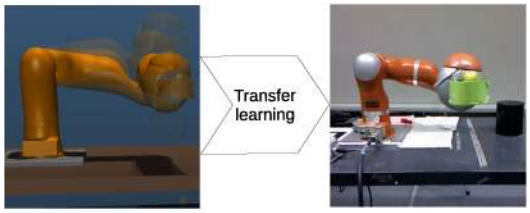

Fig. 2. Transferring ball-in-a-cup skill from a simulated environment in MuJoCo to KUKA LBR 4+.

basket. In this case, the task parameter is the distance of the basket from the base of the robot. KUKA LBR 4+ has seven DOF but only joints 2, 3, and 6 were used; the rest of the joints were kept fixed. Using 20 kernels per DMP, total of $N = 60$ parameters need to be determined for a task parameter value. To vary similarity between source and target environments, densities 1.2, 3.0, 5.0, 6.0, and 7.0 $\frac{\text{kg}}{\text{m}^3}$ were used for environments $i = 1, \ldots, 5$ respectively.

C. Learning Process

In this section, we give an example how a contextual model is learned in the source (simulation) environment and how it is then transferred to the real world, using the ball-in-a-cup as the example task. The policy is represented as DMPs; thus, policy parameters are DMPs shape parameters w (see equation (1)). The task parameter l is the string length. Furthermore, the global model $GPDMP$ maps task parameters l to policy parameters w. Let $GPMDP_s^i$ denote a particular source model learned under certain conditions.

To learn a source database DB_s in the simulated environment $i = 2$ with air density 1.5 $\frac{\text{kg}}{\text{m}^3}$, we started the incremental learning process with reasonable initial policy parameters (DMP shape parameters) for the string length of 34 cm. We optimized these policy parameters using PoWER and added the resulting policy w_{34}^2 to the source database DB_s^2, where superscript 2 refers to the index of the source environment and w_l^i denotes the policy parameters optimized for string length l in environment i. Next, we decreased the string length to 32 cm, which significantly changed the dynamics of the ball. As a result, we could not re-enact the task successfully using w_{34}^2. The policy was then optimized using PoWER using the previous policy as a starting point, and the resulting policy parameters w_{32}^2 was added to DB_s^2. The source database of two MPs, $DB_s^2 = \{\{32; \text{w}_{32}^2\}, \{34; \text{w}_{34}^2\}\}$, was then used to determine the global model $GPDMP_s^2$, model selection indicating a linear model. Moving to string length 30 cm, an initial policy was determined using the contextual model, and after optimization using PoWER, the resulting MP w_{30}^2 was added to the database. The contextual model was then re-estimated, where model selection indicated a second order model for $GPDMP_s^2$.

To study the generalization capability of a contextual model with respect to the task parameter, we define $g_s(m)$ as the range of task parameter values for which the contextual model m is successful, that is, the MP generated by the contextual model leads to a successful re-enactment of the task. For example, if model $GPDMP_s^2$ is successful for task parameter values $l = 29, 30, \ldots, 37$ cm, we say that the the generalization capability of the source model $g_s(GPDMP_s^2)$ is 8 cm. The subscript s in $g_s(\cdot)$ refers to the generalization capability of a model in its corresponding source environment; whereas, $g_t(\cdot)$ denotes the generalization capability of the model in the target environment. $g_s(\cdot)$ and $g_t(\cdot)$ were evaluated experimentally (in simulation or in real world, respectively) by varying the task parameter one unit at a time.

To transfer $GPDMP_s^2$ to the real world, Algorithm 1 was applied to generate a transferred model $GPDMP_t^2$ as follows. After success in using unmodified source policy for string lengths 27 cm to 35 cm ($g_t(GPDMP_s^2) = 8$ cm), w_{36}^2 failed to reproduce the task successfully in the real world for string length 36 cm. The algorithm then optimized the primitive using PoWER and added the resulting MP w_{36}^r to target database DB_t^2, superscript r indicating the real world. Using this primitive in transfer, the generalization capability of the resulting transferred model $GPDMP_t^2$ increased to $g_t(GPDMP_t^2) = 10$ cm, being successful for $l = 26, \ldots, 36$ cm.

D. Hypotheses and Results

We studied 5 hypotheses about the transfer capability:

1) Transfer improves the generalization capability of a source model beyond its direct use, that is $g_t(GPDMP_t^i) > g_t(GPDMP_s^i)$.
2) As the source environment becomes more dissimilar from the target environment, the transfer capability deteriorates, that is $g_t(GPDMP_t^{i+1}) \leq g_t(GPDMP_t^i)$ for $i = 1, \ldots, 4$.
3) The larger the generalization capability of the source model, the larger the generalization of the transferred model.
4) The more samples are used for transferring, the better the transfer capability will be.
5) Incremental transfer enhances the transfer capability beyond the non-transfer incremental learning, that is $g_t(GPDMP_t^i) > g_t(GPDMP_r^i)$.

1) Direct use of source policy vs transferred one: In order to study the benefit of transfer, we compared the generalization capability of the source $GPDMP_s^2$ versus the transferred $GPDMP_t^2$ model. The experiment was repeated twice for the ball-in-a-cup and 5 times for the basketball task. On average, 7 roll-outs were required for ball-in-a-cup and 15 roll-outs for basketball to converge, while learning from demonstration required 100 roll-outs on average. Results shown in Figs. 3a and 3b show consistent increase of generalization capability in both

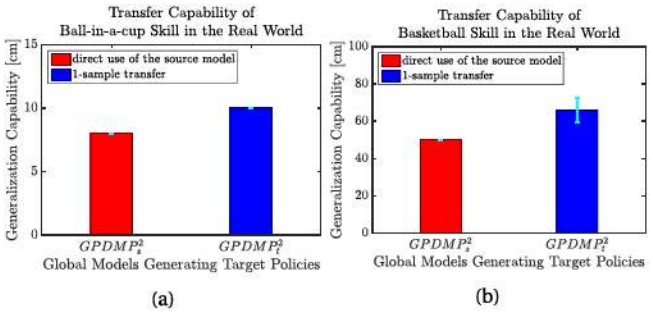

Fig. 3. Generalization capability of $GPDMP_s^2$ (in red) versus $GPDMP_t^2$ (in blue) for: (a) ball-in-a-cup and (b) basketball task. Note that $g_t(GPDMP_t^2) > g_t(GPDMP_s^2)$ for both tasks. Cyan bar denotes standard deviation.

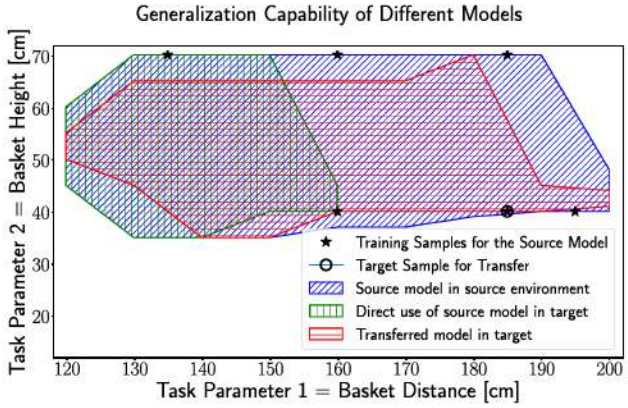

Fig. 4. The generalization performance in environments characterized by two context parameters. Note that only the one sample denoted by circle was used to transfer the skill.

cases, on average 25% increase for ball-in-cup and 32% for basketball even though only one sample MP was used for transfer. We also studied the transfer capability for basketball skill with two task parameters, namely basket height and basket distance. We achieved a generalization of 2435 cm^2 in the source environment. Direct use of the source model in the target resulted in a generalization of 1100 cm^2. Using a single sample transfer, the generalization increased to 1690 cm^2, a 53% improvement over the direct use (see Fig. 4).

The experiments demonstrate that the proposed transfer approach has indeed increased the generalization capability in the target environment further than the direct use of the source model indicating its effectiveness.

2) Similarity Between Source and Target Environments: To study how the level of similarity between the source and target environment influences the transfer capability, we created various source environments by changing their air density in the simulation. In other words, we managed to make a source environment more dissimilar from real world (target) by making its air more dense. Then, we applied 1-sample transfer from these different source environments and evaluated their transfer capability. 1-sample transfer refers to transferring a source global model $GPDMP_s^i$ using only one MP from the target (real-world) environment. In other words, target database $DB_t^i = \{\{l; \mathrm{w}_l^r\}\}$ has only one successful MP, that is $|DB_t^i| = 1$.

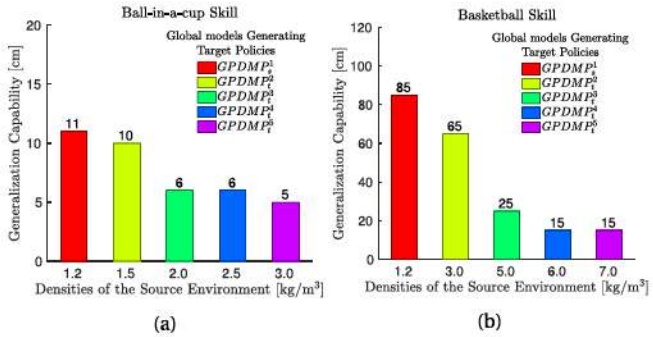

Fig. 5. Generalization capability of 1-sample transfer from different environments for (a) ball-in-a-cup and (b) basketball. Note that $g_t(GPDMP_t^i) \geq g_t(GPDMP_t^{i+1})$ for i=2, 3 and 4 and that $g_t(GPDMP_s^1)$ is the largest in both tasks.

We only considered zero-sample transfer from environment 1 because it has the same air density as the real world. Furthermore, models $i = 2, \ldots, 5$ have been trained in environments $i = 2, \ldots, 5$ respectively. We observed that the direct use of source model $GPDMP_s^1$ (zero-sample transfer) has led to the best generalization performance in the real world (target) for both tasks (see red bars in Fig. 5a and Fig. 5b). Zero-sample transfer from environment 1 has achieved this great performance because the density of the air in the source environment is the same as the real world. On the other hand, 1-sample transfer capability has decreased as the density of the source environment has increased (see Fig. 5). In other words, $g_t(GPDMP_t^i) \geq g_t(GPDMP_t^{i+1})$ for i=2, 3 and 4. Therefore, the transfer capability depends on the similarity between the source and the target environment.

3) Generalization Capability in Source Tasks: We studied how the generalization performance of the source model $GPDMP_s^i$ in its corresponding source environment $g_s(GPDMP_s^i)$ influences the generalization capability of the corresponding transferred model $GPDMP_t^i$ in the target environment $g_t(GPDMP_t^i)$ after transfer with one sample. We trained three models ($i = 5, 6, 7$) with air density set to 3.0 $\frac{\mathrm{kg}}{\mathrm{m}^3}$ in environment 5 for ball-in-a-cup task and two models ($i = 5, 6$) with air density set to 7.0 $\frac{\mathrm{kg}}{\mathrm{m}^3}$ in environment 5 for basketball. Because of stochastic nature of the training, the generalization capabilities of the models differ.

Figure 6 shows the generalization capability of these models in addition to the transfer capabilities, blue showing the capability in the source and red in the target. For the ball-in-a-cup task, models $GPDMP_t^6$ and $GPDMP_t^7$ have 2 cm larger transfer capability than $GPDMP_t^5$ (see Fig. 6a). Meanwhile, their corresponding source models $GPDMP_s^6$ and $GPDMP_s^7$ also have better generalization capability than $GPDMP_s^5$. Similarly in the basketball task, the greater generalization capability of the source model indicates greater generalization of the corresponding target model.

However, the transfer capability is also influenced by the similarity between the source and the target environment as stated in Sec. IV-D2. For example, in the basketball task (Fig. 6b), although $g_s(GPDMP_s^3) < g_s(GPDMP_s^6)$, 1-sample transfer

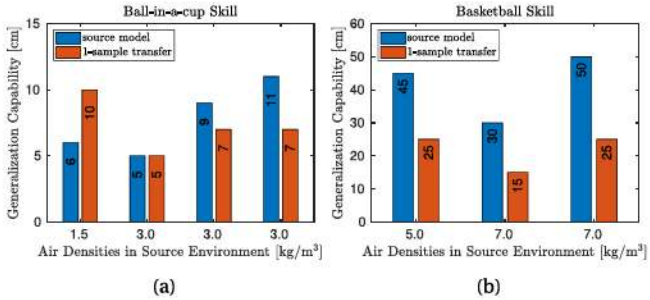

Fig. 6. Generalization capability of source models $GPDMP_s^i$ in their corresponding source environment, namely $g_s(GPDMP_s^i)$ (represented by blue bars) versus their 1-sample transfer capability in the real world, namely $g_t(GPDMP_t^i)$ (illustrated by brown bars) for: (a) ball-in-a-cup and (b) basketball task.

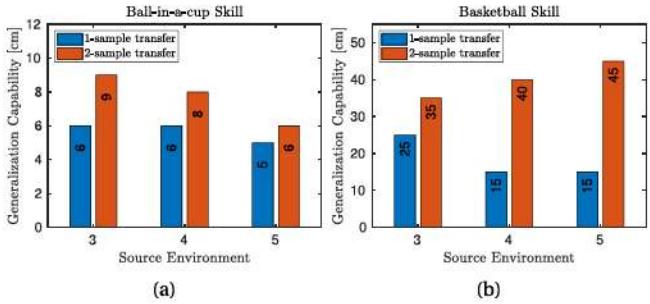

Fig. 7. Transfer capability of 1-sample versus 2-sample transfer from different environments in the real world for: (a) ball-in-a-cup and (b) basketball skill.

capability of model 6 is not better than model 3; this is mainly because the corresponding environment for model 3 (environment 3) is more similar to the real world than the corresponding environment for model 6 (environment 5). Similar behaviour can also be seen for the ball-in-a-cup task. This implies that in order to achieve the best transfer capability, several contextual models should be trained in the simulation, and the one with the best source generalization capability be selected as the candidate source model for transfer. Furthermore, if affordable, one should make source environment as similar as possible to the target to achieve the best transfer capability.

4) 2-Sample Transfer Improves Transfer Capability Over 1-Sample: We studied whether 2-sample transfer enhances the transfer capability over 1-sample transfer. In this experiment, we selected environments 3, 4 and 5 for both basketball and ball-in-a-cup tasks. It is noteworthy that 2-sample transfer has enhanced the transfer capability beyond the 1-sample transfer consistently across all these three environments in both tasks (compare blue vs brown in Figs. 7a and 7b). Therefore, the more number of samples used for transferring a source model, the better transfer capability its corresponding transferred model have achieved.

Furthermore, It is interesting to note that both 1-sample and 2-sample transfer capability of the ball-in-a-cup task decreased as the corresponding source environment became more dissimilar from the target environment making the transfer more difficult. On the other hand, 2-sample transfer capability of the basketball did not decrease unlike the ball-in-a-cup task. However, this was

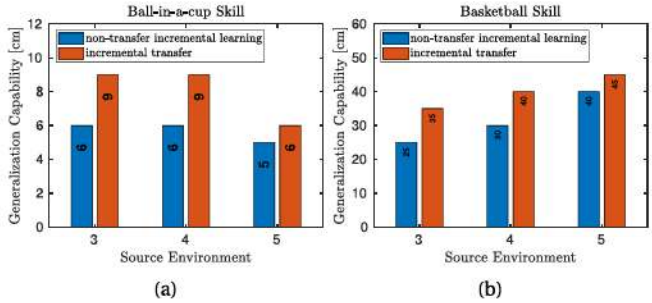

Fig. 8. Standard non-transfer incremental learning $GPDMP_r^i$ vs incremental transfer $GPDMP_t^i$ from different environments, namely $i = 3, 4, 5$ for: (a) ball-in-a-cup and (b) basketball skill.

because the velocity limit of the robot has exceeded the safety threshold for both $GPDMP_t^3$ and $GPDMP_t^4$, thus limiting their transfer capability. In fact, $GPDMP_t^3$ could generalize from task parameter, that is $l = 145$ cm up to 180 cm, but the velocity limit of joint $A2$ was exceeded by 35 deg/sec when generalizing to $l = 185$ cm. This happened at early part of the trajectory causing the robot to stop the movement. On the other hand, $GPDMP_t^4$ could generalize from $l = 165$ to 205 cm. It is noteworthy to mention that exceeding the velocity limit also occurred for $GPDMP_t^4$ when generalizing to $l = 205$ cm, but the robot stopped only after the ball was released into air.

5) Incremental Transfer vs Standard Incremental Learning: In this experiment, we have used two samples from the real world for transferring from a simulated environment. Now, it is natural to ask whether 2-sample transfer is more useful than the standard non-transfer incremental learning. Standard incremental learning refers to learning the global model $GPDMP_r^i$ only using the MPs in the target DB_t^i. In this case, we ignore the corresponding source database, that is DB_s^i and the underlying regularities which we have captured in the simulation. On the other hand, incremental transfer combines transfer with incremental learning and consider both the source DB_s^i and target DB_t^i for computing the transferred model $GPDMP_t^i$. In other words, we are adapting the underlying regularities of the source task to fit to the target task. In order to test whether the information which the source model is providing is beneficial for the target task, we conducted an experiment where we selected source environments 3, 4 and 5 for both ball-in-a-cup and basketball tasks.

It is noteworthy that incremental transfer has led to larger generalization capability throughout these three environments and in two different tasks (see Fig. 8); that is, $g_t(GPDMP_t^i) > g_t(GPDMP_r^i)$ for environments i=3, 4 and 5. This is mainly because the target database DB_t^i is not capable enough to capture the required underlying regularities of the target task due to the limited number of MPs in the target DB_t^i.

V. Conclusion

In this letter, we proposed an incremental transfer framework for transferring a generalizable model from simulation to real world. The proposed framework consists of incremental learning of a source model and its incremental transfer to target.

Incremental learning in source environment results in a source database of motion primitives that is used to learn a contextual model capturing the underlying regularities of the simulated task. The real-world target model is learned incrementally by combining the contextual source model with one or few samples from the real world. Experiments demonstrated that learning a global model in simulation is useful as it leads to a generalizable model. In fact, using the global model, learning is not required for every new situation as the model can generalize successfully to new task parameter values, thus reducing the interaction with the physical system. We observed that as the target becomes more dissimilar from the source environment the generalization capability of the direct use of the source model decreased. This indicated that the underlying regularities of the source task need to be adapted to better represent the underlying regularities of the target task. However, instead of adapting the underlying dynamics of the task or system, we proposed a transfer approach which is agnostic to the dynamics model and sample efficient. A single sample from target domain increased the generalization capability by 25–32%. Furthermore, the more samples we used from the real world, the larger transfer capability was achieved, indicating that the transferred model represents better the underlying regularities of the target task. As a side effect, the transferred model makes learning for a new situation easier and faster as it provides RL with a better initial policy to optimize.

We also compared incremental transfer versus standard incremental learning where a contextual model is trained using only data from target domain. Experiments revealed that incremental transfer was superior in terms of generalization capability. This indicates that the information from the source domain improves generalization even when the dynamics between source and target differ.

Experiments revealed also that the transfer capability of the contextual model depends on two additional factors, namely the generalization capability of the source model and the level of similarity between the source and target environment. This implies that due to the stochastic nature of learning, several contextual models can be trained in simulation and the one with the best source generalization capability selected as the candidate model for transfer. Besides that the source environment should be made as similar as possible to the target environment to achieve the best transfer capability.

All things considered, we proposed a novel sample-efficient transfer approach which is agnostic to the dynamics of a system, thus accommodating the unmodeled dynamics. As a future work, we will study how to speed up incremental transfer and the incorporation of active learning in the proposed incremental transfer framework.

REFERENCES

[1] M. Hazara and V. Kyrki, "Speeding up incremental learning using data efficient guided exploration," in *Proc. IEEE Int. Conf. Robot. Autom.*, Brisbane, QLD, Australia, 2018, pp. 1–8, doi: 10.1109/ICRA.2018.8461241.

[2] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, no. Jul, pp. 1633–1685, 2009.

[3] F. Fernández and M. Veloso, "Learning domain structure through probabilistic policy reuse in reinforcement learning," *Prog. Artif. Intell.*, vol. 2, no. 1, pp. 13–27, 2013.

[4] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*, 2017, pp. 23–30.

[5] I. Mordatch, K. Lowrey, and E. Todorov, "Ensemble-cio: Full-body dynamic motion planning that transfers to physical humanoids," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 5307–5314.

[6] A. Rajeswaran, S. Ghotra, S. Levine, and B. Ravindran, "Epopt: Learning robust neural network policies using model ensembles," 2017.

[7] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2018, pp. 1–8.

[8] R. Antonova, S. Cruciani, C. Smith, and D. Kragic, "Reinforcement learning for pivoting task," 2017, arXiv:1703.00472.

[9] J. C. G. Higuera, D. Meger, and G. Dudek, "Adapting learned robotics behaviours through policy adjustment," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 5837–5843.

[10] M. Saveriano, Y. Yin, P. Falco, and D. Lee, "Data-efficient control policy search using residual dynamics learning," in *Proc. Int. Conf. Intell. Robots Syst.*, 2017, pp. 4709–4715.

[11] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 2, pp. 408–423, Feb. 2015.

[12] A. G. Kupcsik *et al.*, "Data-efficient generalization of robot skills with contextual policy search," in *Proc. 27th AAAI Conf. Artif. Intell.*, 2013, pp. 1401–1407.

[13] A. Abdolmaleki, N. Lau, L. P. Reis, J. Peters, and G. Neumann, "Contextual policy search for generalizing a parameterized biped walking controller," in *Proc. IEEE Int. Conf. Auton. Robot Syst. Competitions*, 2015, pp. 17–22.

[14] D. Forte, A. Gams, J. Morimoto, and A. Ude, "On-line motion synthesis and adaptation using a trajectory database," *Robot. Auton. Syst.*, vol. 60, no. 10, pp. 1327–1339, 2012.

[15] A. Ude, A. Gams, T. Asfour, and J. Morimoto, "Task-specific generalization of discrete and periodic dynamic movement primitives," *IEEE Trans. Robot.*, vol. 26, no. 5, pp. 800–815, Oct. 2010.

[16] M. Hazara and V. Kyrki, "Model selection for incremental learning of generalizable movement primitives," in *Proc. 18th IEEE Int. Conf. Adv. Robot.*, Hong Kong, 2017, pp. 359–366.

[17] A. Carrera, N. Palomeras, N. Hurtós, P. Kormushev, and M. Carreras, "Learning multiple strategies to perform a valve turning with underwater currents using an I-AUV," in *Proc. OCEANS*, Genova, Italy, 2015, pp. 1–8.

[18] T. Matsubara, S.-H. Hyon, and J. Morimoto, "Learning parametric dynamic movement primitives from multiple demonstrations," *Neural Netw.*, vol. 24, no. 5, pp. 493–500, 2011.

[19] F. Stulp and O. Sigaud, "Path integral policy improvement with covariance matrix adaptation," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012.

[20] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, vol. 2, pp. 1398–1403.

[21] J. Kober and J. R. Peters, "Policy search for motor primitives in robotics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 849–856.