Residual Learning from Demonstration: Adapting DMPs for Contact-rich Manipulation

Residual Learning from Demonstration: Adapting DMPs for Contact-rich Manipulation

Todor Davchev[†•*] Kevin Sebastian Luck[⊙] Michael Burke[◦] Franziska Meier[◇]
Stefan Schaal[‡] and Subramanian Ramamoorthy[†]

***Abstract*—Manipulation skills involving contact and friction are inherent to many robotics tasks. Using the class of motor primitives for peg-in-hole like insertions, we study how robots can learn such skills. Dynamic Movement Primitives (DMP) are a popular way of extracting such policies through behaviour cloning (BC) but can struggle in the context of insertion. Policy adaptation strategies such as residual learning can help improve the overall performance of policies in the context of contact-rich manipulation. However, it is not clear how to best do this with DMPs. As a result, we consider several possible ways for adapting a DMP formulation and propose "residual Learning from Demonstration" (rLfD), a framework that combines DMPs with Reinforcement Learning (RL) to learn a residual correction policy. Our evaluations suggest that applying residual learning directly in task space and operating on the full pose of the robot can significantly improve the overall performance of DMPs. We show that rLfD offers a gentle to the joints solution that improves the task success and generalisation of DMPs and enables transfer to different geometries and frictions through few-shot task adaptation. The proposed framework is evaluated on a set of tasks. A simulated robot and a physical robot have to successfully insert pegs, gears and plugs into their respective sockets. Other material and videos accompanying this paper are provided at *https://sites.google.com/view/rlfd/*.**

***Index Terms*—Learning from Demonstration; Reinforcement Learning; Sensorimotor Learning**

I. INTRODUCTION

PART insertion, e.g. plugs, USB connectors, house keys or car refuelling nozzles, is a manipulation skill required in a variety of practical applications, ranging from the home to manufacturing environments [1]. It remains surprisingly difficult to find robust and general solutions for this class of tasks without depending on specialised fixtures or other aids. Engineers have long devised ingenious methods for part insertion. However, these are either highly case-specific in manufacturing or not very robust in the face of hardware wear and tear or other environmental uncertainties. Humans solve such tasks without difficulties using visual or tactile perception, dexterous manipulation, and learning. This problem scenario is the focus of our work.

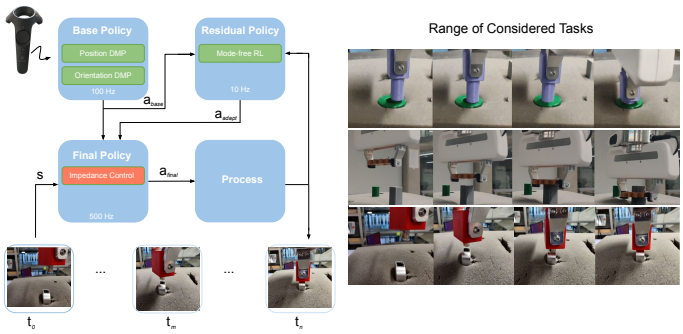

Fig. 1: Left: outline of the proposed framework. We collect demonstrations using an HTC Vive tracker, and extract an initial full pose policy using Dynamical Movement Primitives (DMPs, running at 100Hz). The control command produced by the DMP is corrected by an additional residual policy trained using model-free RL (run at 10Hz). The resulting motor command is then fed into a real time impedance controller (running at 500Hz) in a Franka Panda arm that performs peg, gear or LAN cable insertion in our physical setup. Right: The peg (top), gear (middle) and LAN cable (bottom) insertion tasks considered in this work.

Learning from demonstration (LfD) [2] is a popular approach for effective and rapid skill acquisition in physical robots. Existing solutions to LfD [3], [4] have shown that robots can gain proficiency on tasks even when we may not have detailed models or simulations of complex environments. However, generalising to a range of scenarios without explicitly modelling and identifying the effects of contacts and friction remains difficult, hence inhibiting robot learning in these contact-rich settings. Model-free RL is beneficial for solving challenging tasks [5], [6], but requires large amounts of data and often impractical numbers of training episodes. Furthermore, many hours of training can result in higher levels of equipment wear and tear and increase the risk of more severe hardware damage due to the inherently exploratory nature of RL.

Manuscript received: September, 9, 2021; Revised: December, 4, 2021; Accepted: January, 24, 2022.

This paper was recommended for publication by Editor Tetsuya Ogata upon evaluation of the Associate Editor and Reviewers' comments.

This research was supported by: the EPSRC iCASE award with Thales Maritime Systems provided to Todor Davchev; EPSRC UK RAI Hub NCNR (EPR02572X/1) and Academy of Finland Flagship programme: Finnish Center for Artificial Intelligence funding Kevin S. Luck; the Alan Turing Institute, as part of the Safe AI for surgical assistance project, funding Subramanian Ramamoorthy and Michael Burke at the University of Edinburgh.

* Corresponding Author.

• Research done in part during a Google X AI Residency.

† Authors are with the School of Informatics, University of Edinburgh, 10 Crichton St, EH8 9AB, United Kingdom, {t.b.davchev, s.ramamoorthy}@ed.ac.uk

⊙ Author is with the Department of Electrical Engineering and Automation, Intelligent Robotics, Aalto University, 02150 Espoo, Finland, kevin.s.luck@aalto.fi

◦ Author is with ECSE, Monash University, Melbourne, Australia, michael.g.burke@monash.edu

‡ Author is with [Google X] Intrinsic, Mountain View, CA, USA, stefan.k.schaal@gmail.com

◇ Author is with Facebook AI Research, Menlo Park, CA fmeier@fb.com

Digital Object Identifier (DOI): see top of this page.

Roboticists have recognised the need for adaptive learning from demonstration schemes that occupy a middle ground [1]. This paper follows this spirit by combining LfD and model-free RL in a residual pose correction framework comprised of two complementing policies. DMPs [7] act as a base policy acquired by LfD. The DMP approach has always included an additive coupling term that allows for online modulation of a DMP policy – i.e., DMPs are naturally interpreted as a form of residual learning. We instantiate this coupling term as a state-based policy using variants of state-of-the-art on-policy and off-policy RL trained in an online fashion (Figure 1, left). This yields a learning approach that is more sample-efficient and generalises to a range of start positions, orientations, geometries as well as levels of friction. We study the efficacy of this framework using peg, gear and LAN cable insertions - both in simulation and on a physical robot (Figure 1, right).

DMPs are ubiquitous in behaviour cloning settings and a popular technique for learning from demonstration. Although existing work has explored directly adapting DMP or coupling terms to improve task generalisation [8]–[10], it is still unclear how best to do this for contact-rich insertion tasks. This paper explores DMP correction techniques and shows that a residual correction policy that adapts directly in task space, outside the canonical formulation, and uses reinforcement learning is the most effective strategy among all considered ways for learning insertion skills. Furthermore, this framework explicitly accounts for orientation-based policy corrections in task space. Our results show that orientation-based correction, which is typically not applied to existing residual learning frameworks such as [11], is essential for reliable and robust peg-in-hole insertion. This is partly challenging because additive orientations can introduce catastrophic singularities and locks, while formulating residual orientation-based corrections in quaternion space can be challenging to learn.

Contributions The key contributions of this paper are:

C1 *An extensive comparison of the adaptation of different parts of the DMP formulation using a range of adaptation and exploration strategies for contact-rich insertion;*

C2 *A framework for applying full pose residual learning on DMPs applied directly in task space, and the demonstration of its utility to three types of physical insertion tasks;*

C3 *Showing that using full-pose residual nonlinear policies (e.g. RL-driven) to adapt DMPs results in more accurate, gentle and more generalisable DMP solutions.*

II. RELATED WORK AND CONTRIBUTIONS

Insertion skills are essential in many robot manipulation applications. They are a crucial part of manufacturing assembly, home automation, laboratory testing and even surgical automation [1]. Existing literature on this problem can be organised into two broad categories based on whether or not the proposed methods split the task execution into phases based on contact times. We provide an overview of some of these methods and position our work in this context.

Modelling Contacts It is common practice in extant approaches to first model the time(s) of contact and then to structure the insertion strategy in terms of two separate sub-problems - contact-state recognition and compliant control [1]. Compliant control generally relies on a human engineer to characterise the underlying friction and contact model [12]. Recent approaches demonstrate high-precision assembly through the use of additional force information [13]. These methods require careful design of the manipulated objects and the external environment, e.g. assuming a flat surface surrounding the hole. These assumptions may not hold in many practical and day-to-day settings encountered in the real world, e.g. for RJ-45 connector insertion tasks (Figure 1, bottom row). Also, it tends to be challenging to obtain accurate models of nonlinear changes in contact dynamics over time, which affects reliability. In contrast, rLfD seeks to overcome these limitations by relying on strategies extracted from human demonstrations, and a learned policy that adapts and optimises the end-effector pose directly in task space.

Learning from Demonstration (LfD) Learning policies for complex robot tasks can benefit from expert demonstrations [14]. LfD [2] is a widely adopted approach formalising this idea and has been applied previously in the context of peg-in-hole insertion tasks [15], but does not account for model imperfections. DMPs [7] have been shown to improve the generality of demonstrations in a variety of manipulation tasks [3], [8]–[10], [16]–[22]. However, these works do not consider full pose online adaptation behaviours in task space.

In cases where the error signal is a linear combination of basis functions throughout movement executions, such as minimising force/torque feedback control error in bimanual manipulation tasks [18], [19], DMPs can be adapted by employing linear adaptive policies. Gams et al. [20] directly modify the forcing term (i.e. adaption in parameter space) of a DMP iteratively and apply it to the task of wiping a surface. Abu-Dakka et al. [21] consider adapting DMPs using fixed feedback models as constant gains and focus on learning the forcing term. However, this learns a representation of a nominal behaviour and does not address learning feedback models that are important in complex force-based tasks like the RJ-45 connector insertion used in this work.

Previous work has also used Reinforcement Learning of feedback models, e.g. PoWER [9] or FDG [10], to learn or adjust nominal behaviours within a few iterations in parameter space of the DMP formulation. However, these approaches are limited to linearly weighted combinations of phase-modulated features, which are less expressive and can lead to unsatisfactory performance for contact-rich manipulation tasks. Alternatives, such as eNAC [8] have proposed adapting DMPs in the phase-based coupling term space instead, which is helpful for tasks that require reactive movement such as wiping [3] or hitting a baseball [8]. This formulation does not allow for random local task space exploration (e.g. jiggling), which is helpful in contact-rich manipulation tasks (see Figure 2). Instead, we propose to adapt DMPs directly in task space avoiding these limitations.

In cases where selecting arbitrary desired points of a trajectory can be done without affecting performance - such as in smooth surface painting or handover [4], both translational and orientation-based behavioural cloning (BC) policies can be adapted with an analytical approach. However, large perturbations of a robot's start configuration in the context of insertion

often leads to dramatic differences between a demonstrated trajectory and the desired one. This can invalidate such modes of teaching due to the significant distance between both trajectories. Instead, we propose a generalisation of such methods' structure and describe it as a framework that allows learning to adapt DMPs directly in task space. We show that this allows us to rely on nonlinear model-free approaches, which can benefit contact-rich insertions. Our results show that using this residual learning formulation leads to a robust to perturbations and sample efficient solution.

Residual RL Acknowledging the difficulties of running RL on physical systems, approaches like Johannink et al. [23] combine conventional contact model-based control with model-free reinforcement learning but do not handle orientation-based corrections. Instead, they focus on sliding at low levels of friction. Schoettler et al. [11] propose using stronger priors and combine SAC with a proportional controller to solve industrial position-only insertion tasks. They do not scale to full pose corrections and encourage the use of higher forces via a dense reward function. This works well in Cartesian space settings with very low perturbations but is limiting in the context of insertion. Tuning such dense rewards requires additional insights and potentially increases the risk of hardware damage. Zeng et al. [24], employ adaptive policies for learning how to throw. However, none of these works shows how best to apply residual learning to DMPs, a broadly applicable and flexible class of models of significant interest in robotics. Our extensive evaluation sets out to answer this.

Summary It is clear that there has been substantial work on DMP adaptation using residual policies. However, it remains unclear how best to do so for contact-rich insertion tasks. This paper explores this and finds that a framework bridging reinforcement learning and DMP learning from demonstration in task space (rLfD) is most effective when paired with orientation aware corrections. This framework is evaluated in both simulated and physical systems, running under real-time constraints. Results show that the proposed formulation significantly reduces the sample requirements during training and allows for the use of a sparse reward while preserving the overall improved accuracy achieved by model-free RL. In practice, this is important as it limits the risk of hardware damage and thus makes this approach feasible for real-world applications.

III. METHODS

In this paper, we learn a base policy, π_b, from expert demonstrations, represented directly in task space. We extract separate policies for each individual task in (Figure 1, right). We use two separate formulations for π_b - namely translational and orientational DMPs [7], [16]. The learned behaviour is then executed by a suitable robot controller, i.e. in position or impedance control, to produce the final robot motion. The translation-based base policy $\hat{\pi}_b$ is described by position and velocity terms, $(\mathbf{x}, \dot{\mathbf{x}})$, in 3D Cartesian space. The orientation-based base policy, $\tilde{\pi}_b$ is described using the orientation in quaternion and angular velocity, $(\mathbf{q}, \boldsymbol{\omega})$. The resulting base policy, π_b is capable of imitating the full pose trajectory of the demonstrated behaviour. Such a formulation can generalise well to perturbations of the initialisation and timing conditions, especially when executed in free space. However, it cannot adapt well to previously unseen task settings and environmental perturbations, which are typical for contact-rich manipulation tasks. In this work, we study how to alleviate this limitation using an additional residual policy, π_θ by employing state-of-the-art model-free RL that can act directly on the end-effector twist.

A. Problem Formulation

We consider a finite-horizon discounted Markov decision process (MDP), $\mathcal{M} = (S, A, P, r, \gamma, T)$ with transition probability $P : S \times A \times S \mapsto [0, 1]$ and a reward function $r(x, a) \in \mathbb{R}$. Let $x \in S \subseteq \mathbb{R}^{n_x}$ be an element in the state space S, and $a \in A \subseteq \mathbb{R}^{n_a}$ denote the desired robot end-effector velocity (action). Then, a stochastic control policy parameterised by θ, $\pi_\theta(a|x)$ defines the probability of selecting an action a given a state x. Let $\zeta = (\mathbf{s}_{t_0}, \ldots, \mathbf{s}_T)$ be a trajectory with a discrete horizon T, a discount function $\gamma(\cdot)$ and a state-action tuple $s_t = (x_t, a_t)$ and a trajectory return $R(\zeta) = \sum_{t=t_0}^{T} \gamma r(x_t, a_t)$. In this context, we can define an optimal policy as $\pi^* = \arg\max_{\pi \in \bar{\pi}} J(\pi)$, where $J(\pi) = \mathbb{E}_{s_0 \sim p(s_0), a_t \sim \pi(s)}[R(\zeta)]$ and $\bar{\pi}$, the set of all policies.

We can use a policy gradient method to optimise the objective from the sampled trajectories. By relying on a DMP to extract a fixed and pre-computed offline continuous base policy, π_b, we significantly reduce the complexity of the task for π_θ. The residual policy has to learn how to deviate from the base policy by correcting model inaccuracies and potential environmental changes during execution. The final policy can compensate for system uncertainties through an adaptive term obtained from π_θ while using a base policy, π_b acquired through human demonstrations. A core question in this context is what part of the DMP formulation should π_θ be applied to. We propose to adapt base policies directly in task space as described in the following subsections.

B. Translation-based Residual Corrections for DMPs

Contact-rich manipulation has inherent non-linear dynamical effects between the robot and the surrounding environment. In most insertion tasks, slight perturbations can have drastic consequences on the performance of π_b. To this end, utilising model-free solutions can help relax the challenge of modelling contact dynamics.

Adapting DMP formulations is a well known approach as discussed in Section II. Consider a point-to-point movement with a DMP defined as, $\dot{y} = \frac{1}{\tau^2}(\alpha_v(\beta_v(g-x)-\tau y)+f_\omega+C_t)$. In this context, an online adaptation term can be learnt by using exploration noise η that is applied to different components of this equation, e.g. to the parameters, ω of the forcing term f_ω, the phase-modulated coupling term C_t or outside the phase modulated DMP formulation (e.g. in task space). For brevity, consider describing these different ways of applying η with colour-coding, and namely

$$\dot{y} = \frac{1}{\tau^2}(\alpha_v(\beta_v(g - x) - \tau y) + z f_{\omega+\eta} + C_t(\eta)) + \eta. \quad (1)$$

Most existing work focuses on adapting nominal behaviours by learning a forcing term with some exploration signal η,

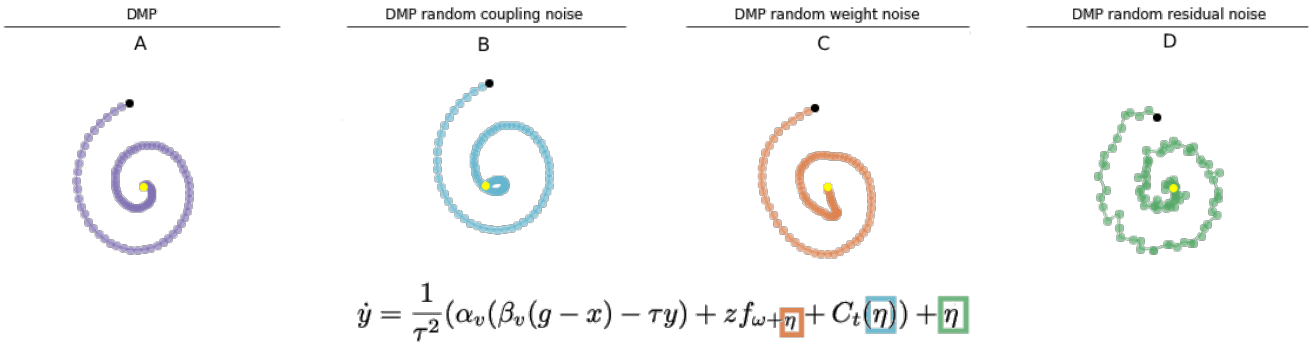

Fig. 2: Types of exploration perturbations with Gaussian noise, η, for a simple Archimedian spiral. Applied on a translational DMP policy (purple - A), shown as equation above. Perturbing directly in task space (green - D) results in local exploration that is important for contact-rich manipulation. In contrast, perturbing the phase-modulated coupling term, C_t (blue - B) or the parameters ω of the forcing term, f_ω (orange - C) results in locally smooth trajectories. Not perturbing the DMP is depicted in purple.

applied in parameter space (shown in orange) [9], [10], [20], [21]. At the same time, some have studied the effects of learning a phase-modulated coupling term[1] (shown in blue) [3], [8]. Recent work, [4] uses adaptive analytical models that can be applied directly in task space, too (shown in green). All of these different modes of exploration lead to learning different residual terms that have their benefits. However, very little has been done to compare the different residual signals that are learnt using the different colour-coded exploration noise from Equation 1. Similar in spirit to [25], this work studies the benefits of those different adaptive formulations and proposes correcting directly in task space. We conjecture that exploration in task space (Figure 2.D) encourages jiggling motion strategies that are useful in the context of insertion. A learnt policy that adapts in task space should thus be superior for contact-rich insertions.

a) Residual Learning in Task Space: In light of our conjecture, we propose to adopt recent advances of residual learning [11], [23] and extract a residual translational policy $\hat{\pi}_\theta$, parameterised by θ that adapts a base prior policy, $\hat{\pi}_b$ outside of the canonical system and directly in task space. The final translational policy can then be defined as $\hat{\pi}_f = \hat{\pi}_b + \hat{\pi}_\theta$. A key aspect of this formulation is that $\hat{\pi}_\theta$ is a complementary policy that operates in task space alongside the DMP policy π_b. We now extend this formulation to orientation-based DMPs.

C. Generalising to Full Pose Corrections

Given environmental uncertainties and assuming also some inaccuracies in the robot control itself, the translational residual formulation introduced above can correct this by combining an adapting policy $\hat{\pi}_\theta$ using the DMP as a prior policy $\hat{\pi}_b$, resulting in $\hat{\pi}_f = \hat{\pi}_b + \hat{\pi}_\theta$.

a) Orientation-based Corrections: The additive correction introduced above is unsuitable for orientations due to the risk for singularities or locks associated with Euler representations. Assuming fixed orientations or constant angular velocity could, in principle, relax this constraint. However, such approximations restrict applicability as motion is rarely fixed in orientation by nature. In order to address this, we introduce a residual formulation capable of accommodating orientations. Quaternion based representations lead to smooth interpolations that are compact and do not suffer from Gimbal locks. Therefore, we propose the following orientation-based residual framework.

b) Orientation based residual corrections in Task Space: In this context, a normalised quaternion $Q = [q^w, q^x, q^y, q^z]$ such that $||Q|| = 1$ is defined as a vector with a real scalar value q^w and a vector $[q^x, q^y, q^z]$ of imaginary components. We define composition $\circ$ between two quaternions using Shuster's notation [26]. An orientation Q_f along some orientation trajectory is produced from a policy $\tilde{\pi}$. In this work, $\tilde{\pi}$ is composed of two separate policies: a stochastic residual orientation policy $\tilde{\pi}_\theta$ which operates in task space and a base orientation policy $\tilde{\pi}_b$ from an orientational DMP [16].

c) Learning residual corrections in quaternion space: Residual orientation is composed in quaternion space and adapts the orientation of the end-effector. The policy predicts the parameters of an angle-axis representation which consists of a 3D unit vector $\mathbf{r}$ around which the robot end-effector is rotated by a scalar angle α. This results in a residual orientation vector $\{\alpha, \mathbf{r}\}$. Assuming that $\alpha \in [-\pi, \pi]$, which covers all rotations, it follows that $cos(\alpha/2)$ is strictly positive. Then, a correction $Q_\Delta = \{q^w_\Delta, q^x_\Delta, q^y_\Delta, q^z_\Delta\}$ can be computed where $q^w_\Delta = cos(\alpha/2)$ is the real part of the quaternion. The orientation-based adaptation relative to the base orientation can be described as quaternion, [27] by

$$Q_\Delta = [cos(\alpha/2), \ \frac{\mathbf{r}}{||\boldsymbol{r}||} sin(\alpha/2)], \tag{2}$$

where $||\boldsymbol{r}||$ is the L2 norm of the rotation vector. This allows us to obtain a quaternion correction term using real numbers. Then, adding the correction to the prior orientation is achieved with quaternion multiplication as

$$Q_f = Q_\Delta \circ Q_b. \tag{3}$$

The final quaternion Q_f is then converted to angular velocity using a log transform [16].

d) A Framework for Full Pose Residual Corrections: In summary, we propose a complete, twist policy π_f as

$$\pi_f = [\hat{\pi}_f, \tilde{\pi}_f]^{\mathrm{T}}. \tag{4}$$

[1]We will refer to this as adapting in coupling term space to differentiate between additive terms.

In practice, we combine the work introduced in [4] and [11] and extend it to a general framework for full pose residual learning. The individual components of our policy can be extracted independently from each other and can be of different nature. They can be learnt (e.g. by using RL) or analytically defined (e.g. by using recursive least squares (RLS) or similar to [4]). Like [11] we complement a base policy with a residual component. However, instead of using a basic proportional controller, we focus on improving DMPs for contact-rich tasks. Residual learning in task space happened to work best. We extend this by adding residual corrections and control with RL in real-time.

D. Execution Details

a) DMPs: We used a single demonstration to build the base policy. This was sufficient to extract 100% successful insertions when an episode starts up to 3mm away from that starting position. We used 40 and 70 basis functions for $\hat{\pi}_b$ and $\tilde{\pi}_b$ respectively. These values were chosen from a grid search of parameter sizes (see web page for details).

b) Model-free Policies: **SAC**: We found that this off-policy approach requires a recurrent neural policy to work in the lower frequency regime given the higher frequency controller. We used a recurrent policy with a cell state of 40. The actor policy comprised 400 and 300-dimensional feed-forward layers with ReLU activation functions. The critic had a single feed-forward layer of size 300. Those parameters were chosen in a grid search across policy network sizes. We used 32 policy update steps per iteration.

PPO We utilise the clipped objective as in Schulman et al. [28]. We used a curriculum for the physical experiments to reduce the sample requirements by varying the starting configuration. As the agent improves we increase task complexity until it becomes 1.5 cm away from the start configuration of the demonstration. PPO was less sensitive to network sizes.

c) Implementation Details: Both environments are built using Tf-Agents [29]. Training is performed using a sparse reward. All policies rely on a normal projection for continuous actions, which uses $\tanh$ to limit its output.

We use MuJoCo [30] and SL [31] for experimenting in simulation and the physical world, respectively. In simulation, we utilise Robosuite [32] and use their position controller applied on low-level actions obtained with inverse kinematics. To evaluate rLfD in the physical world, we used a Jacobian transpose Cartesian impedance control run in real-time (details on the website). Both controllers run at 500Hz, while π_b at 100Hz and π_θ at 10Hz (see Figure 1).

The framework supplies set-points to a real-time running controller and uses a real-time clock to synchronise the concurrently running modules. We extrapolate the DMP corrections using a standard fifth-order polynomial while assuming static repetition of the produced residual term.

IV. EXPERIMENTAL EVALUATION

We evaluate the effects of using different adaptive corrections in the context of skill acquisition for manipulation. We perform a sequence of empirical evaluations using a 7 DoF Franka Emika Panda arm. Our results show that rLfD with model-free reinforcement learning techniques is both sample efficient and improves the performance of DMPs on contact-rich manipulation tasks. We split this section into two main parts: studying the effects of using residual corrections on a simulated environment and applying rLfD in the physical world on three different complexity insertion tasks.

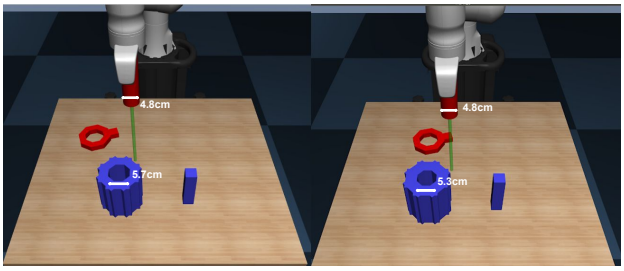

Fig. 3: An easy task (left) and a harder task (right). The robot is initialised with a position sampled uniformly within $\pm$12cm along all axes of the initial position of the demo. Difficulty is defined by the size of the hole. A task is complete when a peg is fully inserted.

A. Applying Residual Corrections

Next, we conduct a thorough study in simulation. We apply different types of corrections to a DMP in the context of contact-rich manipulation.

a) Experimental Details: We study the utility of applying residual corrections in the context of learning to insert in simulation. We utilise two separate tasks we refer to as 'easy' and 'hard'. The difficulty of a task is defined by the size of the hole a peg has to be inserted in (see Figure 3). A tighter hole indicates more difficulties, such as potential jamming due to friction during insertion. The task with a larger hole could be solved using only translation-based corrections. Here, we train on the easy task but evaluate on both.

TABLE I: Accuracy of applying exploration during learning on different parts of the DMP formulation (Type colour from Fig. 2). Eff. is efficiency- the number of episodes a model was trained for.

Type	Exploration Type	Model	Easy	Hard	Average	Eff.	Reward
A	No corrections	DMP [7]	24.0% $\pm$2.5	8.0% $\pm$1.4	16.0% $\pm$2.0	n/a	n/a
B	phase modulated coupling	eNAC [8]	7.2% $\pm$1.9	3.4% $\pm$2.2	5.3% $\pm$2.1	8K	$exp\{-L_1\}$
C	forcing-term parameters	FDG [10]	23.6% $\pm$4.3	16.2% $\pm$1.9	19.9% $\pm$3.1	8K	$exp\{-L_1\}$
C	forcing-term parameters	PoWER [9]	32.2% $\pm$2.8	14.4% $\pm$2.7	23.3% $\pm$2.8	8K	$exp\{-L_1\}$
D	**task space translation**	**rLfD (ours)**	**94.8%** $\pm$1.3	**55.0%** $\pm$2.7	**74.9%** $\pm$2.0	**700**	$\mathbb{1}[L_2 \leq \kappa]$

b) Adapting different parts of a DMP: We summarise the effect of using a range of RL approaches that apply exploration noise during learning to different components of the movement primitives formulation (see website for baseline details). Our findings, reported in Table I, further confirmed the conjecture from Figure 2 that exploring directly in task space is more effective for learning in terms of accuracy. As a result, the learnt task corrections can achieve more sample efficient solutions using only sparse rewards, which is of essential importance to applying RL in practical settings.

c) Adaptive strategy selection in task space: Choosing what part of a DMP to explore to train an RL agent is essential to their successful application in contact-rich settings. However, it is unclear whether using a nonlinear RL-based adaptive strategy is, in fact, necessary for contact-rich insertions. We compare using two different types of nonlinear correcting strategies to a recursive least-squares linear policy [33] and random adaptive noise [34] applied directly in task space.

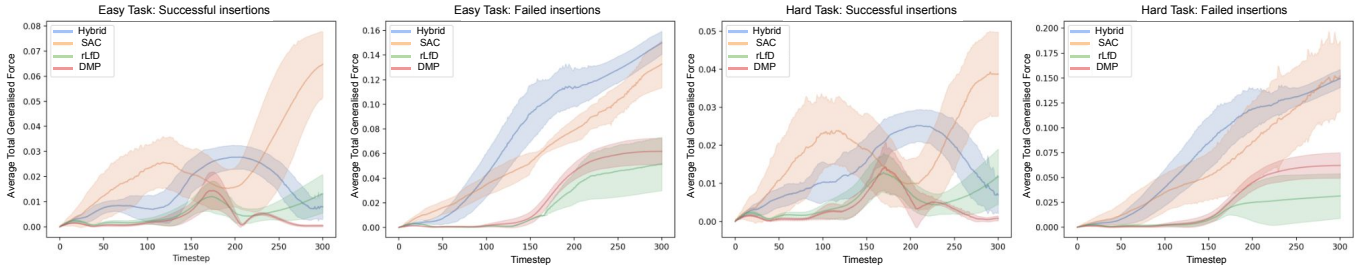

Fig. 4: Comparison between using residual, hybrid, DMP and model-free policies. The residual policy (green) consistently results in experiencing generalised forces comparable to the forces experienced using only a DMP (red) when succeeding. It experiences even smaller forces across both the easy and hard tasks when failing to insert. Lower is better.

Results (Table II) show that nonlinear adaptive strategies are, in fact, helpful in the context of adapting DMPs for contact-rich manipulation tasks. In all cases, only the DMP policy is used for the first half of an episode, with both the base and residual policies for the rest of the execution. While using PPO [28] performs better both on average and in terms of generalising to more challenging tasks, it requires a higher number of steps to reach this level of performance. In contrast, SAC [35] can produce similar results in only a fraction of the required steps but requires an increased number of update iterations and thus a longer time to train. The baselines perform worse on average but require less training as they did not rely on iterative exploration.

TABLE II: Residual adaptive policies in task space for DMPs (Exploration type colour from Fig. 2). Insertion accuracy table.

Type	Corrections	Adaptive Policy	Easy	Hard	Average	Eff.	Reward
A	translation	Random [34]	25.8% ±4.3	9.0% ±1.8	17.4% ±3.1	n/a	n/a
A	translation	Linear [33]	25.4% ±3.6	8.0% ±2.6	16.7% ±3.1	n/a	n/a
D	**translation**	**SAC**	**94.8%** ±1.3	55.0% ±2.7	74.9% ±2.0	**700**	$\mathbb{1}[L_2 \leq \kappa]$
D	**translation**	**PPO**	87.6% ±1.5	**69.0%** ±4.6	**78.3%** ±3.1	25.5K	$\mathbb{1}[L_2 \leq \kappa]$

d) Utilising nonlinear policies applied in task space: Nonlinear policies can significantly improve DMPs performance for contact-rich insertion tasks as shown above. In this section we compare rLfD against a hybrid and an RL-only solution applied in task space. We report results in Table III.

TABLE III: Comparing different ways of using nonlinear policies (Exploration type colour from Fig. 2). Insertion accuracy table.

Type	Corrections	Policy Type	Easy	Hard	Average	Eff.	Reward
A	None	DMP	24.0% ±2.5	8.0% ±1.4	16.0% ±2.0	n/a	n/a
D	translation	(pure) SAC [35]	94.4% ±1.2	41.8% ±7.2	68.1% ±4.2	12.5K	$-(\alpha * L_1 + \frac{\beta}{L_1 - \epsilon})$
D	translation	(hybrid) SAC	57.2% ±2.5	45.6% ±2.7	46.4% ±2.6	8K	$\mathbb{1}[L_2 \leq \kappa]$
D	**translation**	**(rLfD) SAC (ours)**	**94.8%** ±1.3	**55.0%** ±2.7	**74.9%** ±2.0	**700**	$\mathbb{1}[L_2 \leq \kappa]$

Using a model free RL solution is known to be appealing due to its tremendous success in improving accuracy [35]. Our findings confirm this too. Table III shows that using just SAC with no prior π_b can get comparable performance to rLfD. However, a pure RL agent requires carefully engineered rewards and significantly longer training to reach that level of performance. We used $\alpha = 10$, $\beta = 0.002$, $\epsilon = 0.0001$ for the engineered dense reward case reported in row 2 of Table III. In contrast, rLfD generalised better to the harder task and allowed for a sparse reward function using a fraction of the training requirements.

Using a hybrid formulation is a promising related approach. Lee et al. [36] learn policies from visual pre-trained state-space representations with SAC by switching between model-based and model-free policies. We compare against a similar switching strategy to the proprioceptive state representations used by rLfD, switching between a base policy (learned via DMP) and a model-free policy. Using hybrid switching is appealing as it does not require prior knowledge of combining RL with a DMP formulation. In our tasks, we found switching to perform worse than rLfD. This indicates that rLfD may perform well if the more informative, visual representations proposed in Lee et al. [36] are included. We leave this study for future work.

e) Gentle part insertion, analysis: Our analysis (Figure 4) shows that relying on SAC in both hybrid (blue) and purely model-free (orange) approaches results in a more forceful solution when compared to rLfD (green) possibly due to the larger magnitude of the action space required to compensate for the lack of a base policy. The DMP (red) failed at times by constantly pushing downwards, which contrasts with the less forceful searching of rLfD (green).

B. Pose Corrections with Concurrent Real-time Control

In this subsection, we scale rLfD to correct the full pose in a sequence of physical insertion tasks using a less precise impedance controller ran in real-time.

a) Tasks Description: We consider three tasks of differing complexity - peg insertion, similar in setup to the simulated tasks above; gear; and RJ-45 connector insertions (see Figure 1). Each of those tasks introduces an additional mode of complexity within the context of insertion. While physical peg insertion poses challenges due to the friction-heavy interactions with the highly nonlinear surrounding world, it can be solved by mainly relying on translational corrections. On the other hand, the gear insertion task requires perfect alignment of the squared peg with the hole. Due to the tight fit, it also requires a certain amount of downward force to achieve insertion. Finally, inserting an RJ-45 connector requires both perfect position and orientation. Moreover, the connector's plastic tip requires that a precise amount of force be applied to avoid breaking, further increasing task complexity (webpage for more details).

b) Experimental Details: In this subsection we provide additional experimental setup details for the tasks we use.

Peg Insertion The peg's width, depth and height are $28 \times 28 \times 77$mm with a hole with 0.4mm clearance. Evaluation was done on 500 starting points that were sampled randomly from six, 3D uniform distributions, each centred at up to +/- 3cm away along each axis from the demonstrated position.

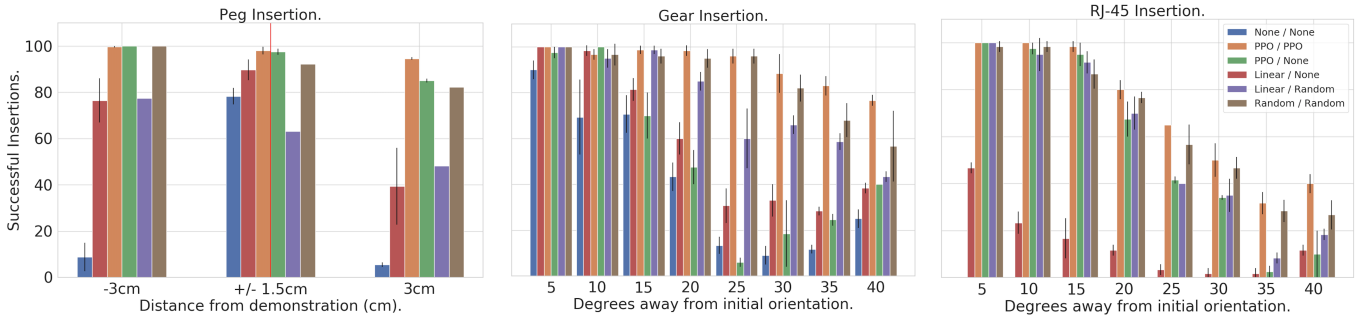

Fig. 5: Successful peg, gear and RJ-45 insertions. Peg insertion results along the x axis are cm away from the initial position (illustrated by a red vertical line). Gear and RJ-45 results are plotted along the x axis as degrees away from the initial orientation. Higher is better.

Gear and Lan Insertion The gear is of size $79.2 \times 79.85 \times 10.79$mm with a square hole of $23 \times 23 \times 10$mm. Lan cable is standard RJ-45. Evaluation uses 160 uniformly sampled unit vectors per task, split into 8 bins of 20 samples of up to 40° and $\pm 1cm$ away from the demo orientation.

Execution Details Each episode lasted at most 10 seconds. We used π_θ after 3.9 seconds of executing the base policy.

c) Using full pose corrections: Translation based corrections rely on unnatural assumptions such as moving on a 3D plane only. A full pose correction is more general and can address a broader range of tasks. We evaluate this next. We compare DMPs with no adaptive terms (None / None), translation-only linear adaptations, similar to [33] (Linear / None), translation-only RL policy (PPO / None), similar to [11]. We also compare against different types of residual adaptive full-pose terms. We use the same linear policy from above and add Uniform random orientation corrections (Linear / Random), a full pose random policy (Random / Random), similar to [34] and our full-pose RL policy (PPO / PPO). Table IV summarises our results. Full-pose rLfD performs

TABLE IV: Utilising full pose accuracy comparison. Pose baselines are named as: translation policy / orientation policy.

Type	Corrections	Adaptive Policy	Peg	Gear	RJ-45	Average
A	No corrections	None / None	52.6% ± 0.7	41.5% ± 1.5	0.0% ± 0.0	31.4% ± 0.7
A	translation	Linear / None	80.7% ± 4.0	58.7% ± 1.1	14.6% ± 1.6	51.3% ± 2.2
D	translation	PPO / None	94.2% ±0.9	50.0% ±0.4	57.2% ±2.1	67.4% ±1.1
A	full pose	Linear / Random (ours)	60.3% ± 2.9	76.2% ± 2.6	57.3% ± 2.5	64.6% ± 2.7
A	full pose	Random / Random (ours)	91.0% ±1.9	86.9% ±1.7	64.8% ±1.2	80.9% ±1.6
D	**full pose**	**PPO / PPO (ours)**	**97.9% ± 1.2**	**92.2% ± 2.6**	**70.6% ± 1.4**	**86.9% ± 1.7**

best. A translation-only RL policy can be useful for simpler tasks, such as the round peg, but not when orientation matters. Random can harm performance on translation-only tasks (such as peg), but it can be useful for tasks like Gear, which heavily depends on accurate orientations. Random noise may result in better accuracy but it requires larger magnitude actions which damages fragile tips like RJ-45 so it should be preferably avoided.

d) Utilising full-pose nonlinear policies: Linear and random solutions are susceptible to latent external factors in the environment, reducing performance. Compared to the perfectly levelled simulated surface, our physical set-up has a 1° slope of the surface a socket is positioned on. Such changes may be hard to notice, but coping with them is not always straightforward and is therefore important. Figure 5 disentangles the results from Table IV. It can be seen that RL significantly increases accuracy on out-of-distribution start configurations ($\pm$3cm and $\geq 20°$) across all three tasks. This indicates that an RL policy can be more effective at coping with latent external variability factors. This is likely due to the better generalisation of the full-pose RL corrections when compared to translation-only, analytical or non-residual solutions. While pure RL solutions may perform just as well, training them is challenging in real-world settings and often impractical, as discussed in Section IV-A.d As a result, we could not extract a successful pure RL policy on the physical robot. More details are available on the website.

e) Speed of Execution: The final speed of insertion is another important factor. Ideally, a successful policy will be highly accurate, safe and fast. We report our findings in Table V. The nonlinear adaptation policy was the fastest. rLfD

TABLE V: Achieved speed of insertion comparison. Pose baselines are named as: translation policy / orientation policy.

Type	Corrections	Adaptive Policy	Peg	Gear	RJ-45	Average
A	No corrections	None / None	7.0sec ±0.1	8.1sec ±0.2	10sec ±0.0	8.6sec ±0.1
A	full pose	Linear / Random (ours)	7.1sec ±0.2	6.6sec ±0.1	8.7sec ±0.1	7.5sec ±0.1
A	full pose	Random / Random (ours)	6.3sec ±0.1	6.2sec ±0.1	8.6sec ±0.1	7.0sec ±0.1
D	**full pose**	**PPO / PPO (ours)**	**5.1sec ±0.1**	**5.9sec ±0.1**	**8.4sec ±0.0**	**6.5sec ±0.1**

significantly improves the performance of a base DMP policy and generalisation to out of distribution start configurations.

f) Transferring Residual Policies across Tasks: A key benefit of rLfD is that it allows residual skill transfer in a few update steps, thanks to its jiggling exploration in task space. Next, we evaluate the performance of a policy trained on a source task src (either Gear or RJ-45) and then transfer to an associated target task $targ$ (either RJ-45 or Gear). Table VI

TABLE VI: Successful insertions on transfer, comparison. Pose baselines are named as: translation policy / orientation policy.

Type	Corrections	Adaptive Policy	Gear	RJ-45	Average	Eff.
D	full pose	(full training) π_{targ}	92.2% ± 2.6	70.6% ± 1.4	81.4% ± 2.0	500
D	full pose	(full training) π_{src}	85.4% ± 1.4	54.5% ± 3.2	69.9% ± 2.3	500
D	full pose	(3-shot) π_{targ}	70.3% ± 4.0	59.1% ± 3.1	64.7% ± 3.6	60
D	full pose	(3-shot) $\pi_{src \to targ}$	92.0% ± 2.1	70.6% ± 1.7	81.3% ±1.9	60

shows the performance of the transferred policies. Columns 4 5 refer to the target tasks. We use one demo for π_b and transfer residual policies trained on the src tasks to $targ$ using three update steps (or 60 episodes). This equates to ∼15 minutes of training, including resets. We denote transferred policy as $\pi_{src \to targ}$ and policies trained only on the $targ$ or src tasks as π_{targ} and π_{src}. We consider π_{targ} and π_{src} trained with the full budget of 500 episodes (or ∼2 hours) and also training π_{targ} from scratch for 3 update steps only (60 episodes).

Results show that rLfD allows for successful policy transfer on both tasks, requiring eight times less training.

V. CONCLUSIONS

This work explores DMP adaptation for contact-rich insertion tasks. Results show that residual learning from demonstration (rLfD) using RL adaptive policies in task space improves the generalisation abilities of DMPs both in simulation and real-world experiments. Results show that rLfD with full pose corrections is highly effective and produces a gentle to the joints solution that can transfer across tasks. Finally, rLfD with nonlinear policies was shown to find better solutions when compared to linear and random policies. Future extensions to this work include consideration of difficulty in terms of sequences of contacts, optimising the parameterisation of the solution and investigating ways to reduce the overall forces applied during insertion. We also envision using rLfD to other contact-rich manipulation tasks, such as painting [4], alongside visual policy inputs and learning generalised across skills policies.

ACKNOWLEDGMENT

The authors would like to thank Giovanni Sutanto, Ning Ye, Daniel Angelov, Martin Asenov and Michael Mistry for the valuable insights and discussions on drafts of this work.

REFERENCES

[1] O. Kroemer, S. Niekum, and G. Konidaris, "A review of robot learning for manipulation: Challenges," *Representations, and Algorithms.(2019)*, 2019.

[2] S. Schaal, "Learning from demonstration," in *Advances in neural information processing systems*, 1997, pp. 1040–1046.

[3] G. Sutanto, Z. Su, S. Schaal, and F. Meier, "Learning sensor feedback models from demonstrations via phase-modulated neural networks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1142–1149.

[4] Y. Huang, F. J. Abu-Dakka, J. Silvério, and D. G. Caldwell, "Toward orientation learning and adaptation in cartesian space," *IEEE Transactions on Robotics*, 2020.

[5] M. Večerík, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[6] J. Luo, E. Solowjow, C. Wen, J. A. Ojea, A. M. Agogino, A. Tamar, and P. Abbeel, "Reinforcement learning on variable impedance controller for high-precision robotic assembly," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3080–3087.

[7] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.

[8] J. Peters and S. Schaal, "Applying the episodic natural actor-critic architecture to motor primitive learning." in *ESANN*, 2007, pp. 295–300.

[9] J. Kober and J. R. Peters, "Policy search for motor primitives in robotics," in *Advances in neural information processing systems*, 2009, pp. 849–856.

[10] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural networks*, vol. 21, no. 4, pp. 682–697, 2008.

[11] G. Schoettler, A. Nair, J. Luo, S. Bahl, J. Aparicio Ojea, E. Solowjow, and S. Levine, "Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5548–5555.

[12] D. E. Whitney, "Quasi-static assembly of compliantly supported rigid parts," *Journal of Dynamic Systems, Measurement, and Control*, vol. 104, no. 1, pp. 65–77, 1982.

[13] T. Inoue, G. De Magistris, A. Munawar, T. Yokoya, and R. Tachibana, "Deep reinforcement learning for high precision assembly tasks," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 819–825.

[14] I. Havoutis and S. Ramamoorthy, "Motion planning and reactive control on learnt skill manifolds," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1120–1150, 2013.

[15] Z. Zhu, H. Hu, and D. Gu, "Robot performing peg-in-hole operations by learning from human demonstration," in *2018 10th Computer Science and Electronic Engineering (CEEC)*. IEEE, 2018, pp. 30–35.

[16] A. Ude, B. Nemec, T. Petrić, and J. Morimoto, "Orientation in cartesian space dynamic movement primitives," in *IEEE International Conference on Robotics and Automation*. IEEE, 2014, pp. 2997–3004.

[17] F. Stulp, E. A. Theodorou, and S. Schaal, "Reinforcement learning with sequences of motion primitives for robust manipulation," *IEEE Transactions on robotics*, vol. 28, no. 6, pp. 1360–1370, 2012.

[18] A. Gams, B. Nemec, A. J. Ijspeert, and A. Ude, "Coupling movement primitives: Interaction with the environment and bimanual tasks," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 816–830, 2014.

[19] N. Likar, B. Nemec, L. Žlajpah, S. Ando, and A. Ude, "Adaptation of bimanual assembly tasks using iterative learning framework," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 771–776.

[20] A. Gams, T. Petric, B. Nemec, and A. Ude, "Learning and adaptation of periodic motion primitives based on force feedback and human coaching interaction," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 166–171.

[21] F. J. Abu-Dakka, B. Nemec, J. A. Jørgensen, T. R. Savarimuthu, N. Krüger, and A. Ude, "Adaptation of manipulation skills in physical contact with the environment to reference force profiles," *Autonomous Robots*, vol. 39, no. 2, pp. 199–217, 2015.

[22] L. Koutras and Z. Doulgeri, "Dynamic movement primitives for moving goals with temporal scaling adaptation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 144–150.

[23] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6023–6029.

[24] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," *IEEE Transactions on Robotics*, vol. 36, no. 4, pp. 1307–1319, 2020.

[25] F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber, "Parameter-exploring policy gradients." *Neural Networks*, 2010.

[26] M. D. Shuster, "The nature of the quaternion," *The Journal of the Astronautical Sciences*, vol. 56, no. 3, pp. 359–373, 2008.

[27] C. Grubin, "Derivation of the quaternion scheme via the euler axis and angle," *Journal of Spacecraft and Rockets*, vol. 7, no. 10, pp. 1261–1263, 1970.

[28] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[29] S. Guadarrama, A. Korattikara, O. Ramirez, P. Castro, E. Holly, S. Fishman, K. Wang, E. Gonina, N. Wu, E. Kokiopoulou, L. Sbaiz, J. Smith, G. Bartók, J. Berent, C. Harris, V. Vanhoucke, and E. Brevdo, "Tf-agents: A library for reinforcement learning in tensorflow," https://github.com/tensorflow/agents, 2018, [Online; accessed 25-June-2019].

[30] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 5026–5033.

[31] S. Schaal, "The sl simulation and real-time control software package," Technical report, University of Southern California, Tech. Rep., 2009.

[32] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín, "robosuite: A modular simulation framework and benchmark for robot learning," in *arXiv preprint arXiv:2009.12293*, 2020.

[33] V. Kumar, E. Todorov, and S. Levine, "Optimal control with learned local models: Application to dexterous manipulation," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 378–383.

[34] C. Pottle, "The digital adaptive control of a linear process modulated by random noise," *IEEE Transactions on Automatic Control*, vol. 8, no. 3, pp. 228–234, 1963.

[35] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning*, J. Dy and A. Krause, Eds., vol. 80, 2018, pp. 1861–1870.

[36] M. A. Lee, C. Florensa, J. Tremblay, N. Ratliff, A. Garg, F. Ramos, and D. Fox, "Guided uncertainty-aware policy optimization: Combining learning and model-based strategies for sample-efficient policy learning," *arXiv preprint arXiv:2005.10872*, 2020.