2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 23-27, 2022, Kyoto, Japan

Hybrid LMC: Hybrid Learning and Model-based Control for Wheeled Humanoid Robot via Ensemble Deep Reinforcement Learning

Donghoon Baek[1], Amartya Purushottam[2], and Joao Ramos[1,2]

Abstract— **Control of wheeled humanoid locomotion is a challenging problem due to the nonlinear dynamics and underactuated characteristics of these robots. Traditionally, feedback controllers have been utilized for stabilization and locomotion. However, these methods are often limited by the fidelity of the underlying model used, choice of controller, and environmental variables considered (surface type, ground inclination, etc). Recent advances in reinforcement learning (RL) offer promising methods to tackle some of these conventional feedback controller issues, but require large amounts of interaction data to learn. Here, we propose a hybrid learning and model-based controller *Hybrid LMC* that combines the strengths of a classical linear quadratic regulator (LQR) and ensemble deep reinforcement learning. Ensemble deep reinforcement learning is composed of multiple Soft Actor-Critic (SAC) and is utilized in reducing the variance of RL networks. By using a feedback controller in tandem the network exhibits stable performance in the early stages of training. As a preliminary step, we explore the viability of *Hybrid LMC* in controlling wheeled locomotion of a humanoid robot over a set of different physical parameters in MuJoCo simulator. Our results show that *Hybrid LMC* achieves better performance compared to other existing techniques and has increased sample efficiency.**

I. Introduction

Humanoid robots have the potential to aid workers in physically demanding and dangerous jobs such as firefighting and disaster relief [1], [2]. In order to aid in these tasks, humanoid robots must be capable of manipulation and locomotion, while being robust to intermittent contact and disturbances. Wheeled-humanoid robots (WHR) are emerging as promising platforms for accomplishing these tasks by combining advantages of mobile robots with the dexterity of legged robots [3], [4].

However, inherent instability, nonlinearity, inaccurate modeling error, and strongly coupled mechanism pose challenges to control WHR. Specifically, balancing control of the WHR is a pivotal role for the robots to transverse various terrains in the real world.

The most common approach of control for these high dimensional nonlinear systems is to model a robot using reduced-order models (RoMs), such as Linear Inverted Pendulums (LIP) and Wheeled Inverted Pendulums (WIP), and adopt model-based linear quadratic regulator (LQR) [5], [6] or model predictive control (MPC) [7]. Alternatively, differential dynamic programming (DDP) and Nonlinear MPC (NMPC) are utilized to generate whole-body motion as a nonlinear approach [8], [9]. Despite their widespread usage among the robotic community, the stability and robustness of these controllers are limited by the fidelity of the robot model and of the surrounding environment. Besides, the performance of these methods depends on the accuracy of the model which has an inherent error.

Fig. 1: Wheeled Humanoid Robot System, SATYRR.

Deep reinforcement learning (RL)-based methods have garnered a growing amount of attention recently as an up-and-coming solution and have shown the success of tackling highly nonlinear locomotion problems [10], [11], [12]. They can overcome the limitations of prior model-based approaches by learning a policy directly from experience and automatically tuning the controller to optimize the given reward (or cost) function representing the task. However, standard RL methods require long interaction between the robot and an environment to learn complicated skills, which can be unsafe initially. Collecting the amount of data that is needed to learn a complex task is time-consuming. Although many Sim-to-Real techniques are suggested [12], [13], [14], reducing the domain gap between a simulation and reality is still challenging and takes an extensive amount of time, up to several days, to train. Exceptionally, control of WHRs solely with RL is challenging since they are inherently unstable at the initial stage during exploration, and re-setup of the robot every time is significantly inefficient and risky.

Meanwhile, the incorporation of the inductive bias or prior knowledge (e.g., analytical model, a conventional controller) with RL looks to address the issues of a conventional controller and RL-based methods by aiding the RL policy to be explored more safely and fast through increasing sample efficiency and reducing state space volume [15], [16], [17].

This work is supported by the National Science Foundation via grant IIS-2024775.

The authors are with the [1] Department of Mechanical Science and Engineering and the [2] Department of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign, USA.dbaek4@illinois.edu

2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-7927-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/IROS47612.2022.9981913

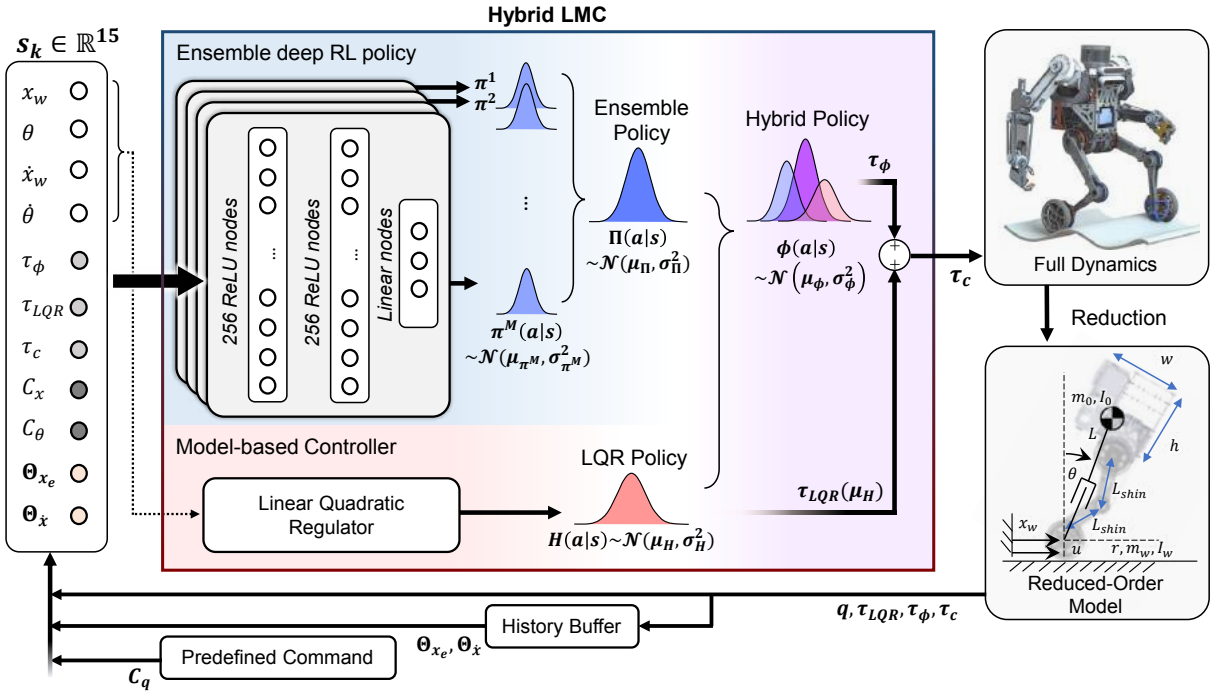


Fig. 2: **Overall Pipeline of Hybrid LMC.** *Hybrid LMC* generates the compensated torque τ_c directly (End-to-End) via a combination of the hybrid policy $\Pi(a|\boldsymbol{s})$ and a LQR controller $H(a|\boldsymbol{s})$. Hybrid policy $\Pi(a|\boldsymbol{s})$ is obtained by using an ensemble deep RL policy that is a mixture of a single SAC policy $\pi(a|\boldsymbol{s})$ with $\boldsymbol{s} \in \mathbb{R}^{15}$ at time k as an input. State $\boldsymbol{s}$ comprises the states of the WIP model of the robot as well as other augmented states including the predefined desired command C_q. All states are defined in the section III-B.

Although this approach has shown impressive results on manipulation and navigation tasks [15], [16], it has yet to be shown for locomotion that is a high dimensional and challenging to collect task-relevant data. Specifically, most controllers for WHRs map from the command to the resulting torque straightly without using a high-level trajectory or a learned policy, and this results in addressing the locomotion problem more challenging.

With this in mind, the goal of this paper is to develop a hybrid controller for a WHR that complements each of the RL and a model-based controller by starting exploration from a relatively stable controller as well as effectively reducing a residual error of the control part.

In this work, a hybrid learning and model-based controller (*Hybrid-LMC*) combining an optimal controller and ensemble deep reinforcement learning is proposed to enhance the locomotion control performance by reducing a residual error resulting from nonlinearities, modeling error, and a variety of environmental changes. The fundamental concept is the same with the residual reinforcement learning [15], but unlike the previous work, we utilized the ensemble RL policy that leverages multiple Soft Actor-Critic (SAC) [18] and distributional action torque provided by an optimal controller (LQR) to choose the compensated torque more carefully through broader exploration with low variance. Our approach allows generating a torque directly contrary to the existing works that use a policy network to build a high-level command such as a trajectory signal.

The contributions of this work are as follows: (1) Hybrid learning and model-based controller, taking the advantage of both a model-based controller and a deep reinforcement learning for increasing the control performance of wheeled-legged humanoid robots, is proposed. To the best of our knowledge, this is the first trial to apply the combined policy such as residual RL to humanoid locomotion. (2) Experimental results indicate that *Hybrid LMC* outperforms residual reinforcement learning and model-free reinforcement learning algorithms as well as compensates the residual error of a LQR controller even in the situation where the diverse physical parameter has changed. (3) Ablation study and additional experiments for investigating *Hybrid LMC* utilizing efficiently are carried out and analyzed carefully. (4) Experimental result using a human data shows the feasibility of *Hybrid LMC* applying to a teleoperated system.

TABLE I: SATYRR specification.

Parameter	Value	Parameter	Value	Parameter	Value
m_0	$6.8kg$	I_0	$0.16kgm^2$	m_w	$0.4297kg$
L	$0.28m$	r	$0.06m$	h	$0.26m$
I_w	$0.00278kgm^2$	L_{sine}	$0.15m$	w	$0.22m$

II. Method

In this section, we discuss the wheeled-legged humanoid robot platform of our choice, SATYRR, its RoM and LQR controller, and the proposed *Hybrid LMC* that combines an ensemble deep RL and LQR. The *Hybrid LMC* pipeline can be seen in Fig. 2. We explain its details in section II-B

A. Modeling and Feedback Control

SATYRR (Fig. 1) is an anthropomorphic biped robot with two powered wheels in place of its feet. We describe the main parameters of the SAYTRR model in Table. I.

Here we use the WIP [5], [19] RoM model that consists of the wheels and a lumped rigid body that represents the robots torso as seen in Fig. 2. The dynamics of this model are given by:

$$\begin{aligned}\left(m_o+m_w+\frac{I_w}{r^2}\right)\ddot{x}_w+m_oLs_\theta\dot{\theta}^2-m_oLc_\theta\ddot{\theta}=u\\ (m_oL^2+I_o)\ddot{\theta}-m_oLc_\theta\ddot{x}_w-m_ogLs_\theta=0\end{aligned} \tag{1}$$

where x_w denotes the traversed position of SATYRR calculated as an average of two wheel encoders, θ is the pitch angle. $\dot{x}_w$ and $\dot{\theta}$ represent their corresponding derivatives. The mass of the body and wheel indicates m_0 and m_w, respectively. u is the control input and torque applied to the wheel, r is the radius of the wheel, and g is gravity. The length between the center of the wheel and the center of mass (CoM) of the body is denoted by L, I_w is the inertia of the wheel, and I_0 is the inertia of the body.

Defining the state vector $\boldsymbol{q} = [x_w \quad \theta \quad \dot{x}_w \quad \dot{\theta}]^\top$, we linearize 1 around the upright equilibrium to obtain the state space equations and resulting optimal gains for the LQR: $\boldsymbol{K} = [-100, -315, -40, -40]$. For regulating yaw motion and the height of the robot conventional PD controller are used.

B. Hybrid Learning and Model-based Controller

Hybrid LMC shares fundamental concepts with residual RL [15] in that they combine a conventional feedback controller with a learned RL policy. In this manner, the two controllers complement each other and compensate for their individual shortcomings:

$$\tau_c = \tau_{LQR}(\boldsymbol{s}) + \tau_\phi(\boldsymbol{s}) \tag{2}$$

where $\tau_{LQR}(\boldsymbol{s})$ and $\tau_\phi(\boldsymbol{s})$ are the output action (torque) from the LQR and the hybrid policy $\phi(a|\boldsymbol{s})$ at given state $\boldsymbol{s}$, respectively. As seen in [15], [16], using prior knowledge of the system (e.g., its model and conventional controller) can aid the RL network in operating within safer bounds as well as increase its sampling efficiency. Conversely, RL policies can assist conventional controller in adapting to various environmental changes by interacting with the world. The proposed *Hybrid LMC* follows this outline but differs from previously explored residual RL frameworks - instead of a deterministic policy, a stochastic approach with distributional actions is utilized. We assume that a stochastic approach promotes the search - through randomly sampled behaviors - of the nearby action-space for more optimal torques. This ultimately result in better tracking of the desired states and in reduction of residual error, $\Delta\boldsymbol{q} = \boldsymbol{q}^{des} - \boldsymbol{q}$ created by unexpected disturbance, and environmental changes. The detailed procedure of *Hybrid LMC* is decribed in Algorithm 1. Also we note that our strategy builds its action as the sum of a stochastic policy and the conventional feedback controller (i.e the LQR), unlike BCF where the action is only sampled from the hybrid policy [17].

In order to take advantage of a stochastic approach and alleviate its drawback of large behavior variance, we use an ensemble technique that leverages multiple RL policy networks $\pi(a|\boldsymbol{s})$ in parallel [20]. The action of the $\phi(a|\boldsymbol{s})$ follows the composite Gaussian distribution $\phi(a|\boldsymbol{s}) \sim \mathcal{N}(\mu_\phi, \sigma_\phi^2)$ computed as follows:

$$\mu_\phi = \frac{\mu_\Pi\sigma_H^2 + \mu_H\sigma_\Pi^2}{\sigma_\Pi^2+\sigma_H^2}\ ,\ \sigma_\phi^2 = \frac{\sigma_H^2\sigma_\Pi^2}{\sigma_H^2+\sigma_\Pi^2} \tag{3}$$

where $\mu_H(\boldsymbol{s})$ denotes the mean of action from LQR and σ_H^2 is its variance. To acquire a distributional action from a conventional controller, we empirically assume the variance $\sigma_H^2(=0.4)$ for the LQR. The mean $\mu_H(\boldsymbol{s})$ is the same with an original action from LQR. We believe that the LQR, $H(a|\boldsymbol{s})$, can guide $\phi(a|\boldsymbol{s})$ in exploring more realistic torques during the early stages of training as the feedback controller is able to leverage prior information about the model and dynamics.

As seen in ensemble techniques [21], the mean μ_Π and variance σ_Π^2 of a uniformly weighted Gaussian mixture model $\Pi(a|\boldsymbol{s})$, Ensemble policy, are obtained by combining M number of single RL policy $\pi(a|\boldsymbol{s})$:

$$\mu_\Pi(\boldsymbol{s}) = M^{-1}\sum_{m=1}^{M}\mu_{\pi_m}(\boldsymbol{s}) \tag{4}$$

$$\sigma_\Pi^2(\boldsymbol{s}) = M^{-1}\sum_{m=1}^{M}(\sigma_{\pi_m}^2(\boldsymbol{s}) + \mu_{\pi_m}^2(\boldsymbol{s})) - \mu_\Pi^2(\boldsymbol{s}) \tag{5}$$

where $\mu_{\pi_m}(\boldsymbol{s})$ and $\sigma_{\pi_m}^2(\boldsymbol{s})$ denote the mean and variance of a single RL policy $\pi(a|\boldsymbol{s})$. Each RL policy $\pi(a|\boldsymbol{s})$ is trained with the use of the SAC algorithm [18] that has achieved state-of-the-art (SOTA) performance in simulated robotic systems by addressing the continuous action problem. SAC was determined suitable here because it is a stochastic policy that chooses an action by sampling from a Gaussian distribution. This enables exploration of a larger state-space and action-space area.

The hybrid policy $\phi(a|\boldsymbol{s})$ samples the appropriate torques mainly affected by $\sigma_\Pi^2(\boldsymbol{s})$ and $\sigma_H^2(\boldsymbol{s})$. We assume that the variance of $\Pi(a|\boldsymbol{s})$ gradually decreases so $\phi(a|\boldsymbol{s})$ follows the ensemble policy $\Pi(a|\boldsymbol{s})$ more, and the LQR less as the policy is learned over time. This is motivated by epistemic uncertainty estimation techniques [17], [21].

III. Experiment

A. Experimental Setup

Simulation Setup: All experiments for validating the *Hybrid LMC* were conducted using MuJoCo [22] simulation which is widely used to evaluate many learning-based methods. We modeled a wheeled humanoid robot, SATYRR using a Unified Robot Description Format (URDF) that has the same physical parameters (Table. I) as a real hardware platform. (Fig. 3)

Algorithm 1 Hybrid Leaning and Model-based Controller

Require: Learned M policies $(\pi_1(a_1|s_1),\ \pi_2(a_2|s_2),\ \ldots, \pi_M(a_M|s_M))$ from SAC models and LQR controller $H(a|s) \sim \mathcal{N}(\mu_H, \sigma_H^2)$

Ensure: Compensated torque τ_c

for $n = 0, \ldots$ epoch **do**
  Select a single agent randomly among M policies
  for $m = 0, \ldots, M$ multiple agents **do**
    Observe state s_m and act an action $a_m \sim \pi_m(\cdot|s_m)$
  end for
  Compute a single univariate Gaussian distribution $\Pi(a|s) \sim \mathcal{N}(\mu_\Pi, \sigma_\Pi^2)$ (see Equation 4 and 5)
  Compute the composite Gaussian distribution $\phi(a|s) \sim \mathcal{N}(\mu_\phi, \sigma_\phi^2)$ (see Equation 3)
  $a = H(a|s) + \tanh(\phi(a|s))$
  Execute a and Observe next state $s^{'}$, reward r, and done signal d
  Store $(s, a, r, s^{'}, d)$ in replay buffer $\mathcal{D}$
  If s' is terminal, reset environment state.
  If it's time to update **then**
    Compute targets and Update Q-function, policy, and target networks based on SAC algorithm [18]
  end if
until convergence

Experimental Variation: We tested *Hybrid LMC* on a diverse set of model parameters through changing of mass, friction, gear ratio, and CoM position values. The ranges of each parameter are described in Table. II.

Baselines: *Hybrid LMC* is compared to the following baselines:

1) LQR controller: LQR controller derived using a WIP model (a conventional feedback controller).

2) Model-free RL algorithms: SAC (stochastic approach) and Deep Deterministic Policy Gradient (DDPG) [23] (deterministic approach) have shown promising results in the continuous action space.

3) Residual RL: Residual RL framework [15] consisting of the sum of a deterministic residual policy and a feedback controller. To benchmark this framework's perfromance for comparision, we tested DDPG with LQR (*DDPG+LQR*) and SAC with LQR (*SAC+LQR*) in our experiments.

4) Bayesian controller fusion (BCF): A hybrid control strategy combining a model-free RL and a conventional controller [17] that motivates a basic structure of *Hybrid LMC*.

During experiments, we chose the best model from each method. All methods are trained with the same reward function, same state definition s, same LQR gains (equal as parameters described in section II-A), and system parameters mentioned in section II. Note that we only trained our model, *Hybrid LMC*, using velocity profiles of 5th-order polynomials. Other methods (e.g. DDPG, SAC+LQR, etc.) required training on full reference trajectories.

Evaluation Metric: To compare the performance of *Hybrid LMC* against baselines, we use root mean square error (RMSE) between the errors of respective desired states and position of x_w, velocity $\dot{x}_w$, and pitch angle θ.

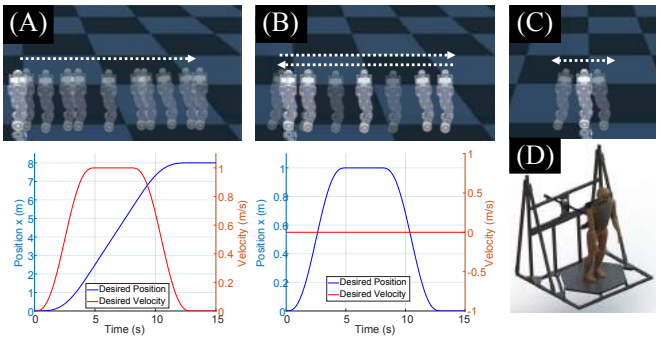


Fig. 3: **Experiments in MuJoCo Simulation:** (A) Task2: a 5th-order velocity trajectory is given as an input $\dot{x}_w^{des}$ ($x_w^{des} = x_w^{des} + \dot{x}_w^{des}\Delta t$). (B) Task3: a 5th-order position trajectory is given as an input x_w^{des} ($\dot{x}_w^{des} = 0$). (C) Task1: Balancing task in a fixed point ($x_w^{des}, \dot{x}_w^{des} = 0$). (D) Human Machine Interface (HMI) [24] to provide a human data. $\boldsymbol{q}^{des} = [x_w^{des}, \dot{x}_w^{des}, \theta^{des}, \dot{\theta}^{des}]^\top$ where both θ^{des} and $\dot{\theta}^{des}$ are 0.

B. Training Details

Defining State-Action Space and Reward Function: The state $\boldsymbol{s}$ at time k is defined as $\boldsymbol{s}^k = \langle \boldsymbol{q}^k, \boldsymbol{\tau}^{k-1}, \boldsymbol{C}^k, \boldsymbol{\Theta}^k \rangle$ and each components are defined as follows:

- State-space vector: $\boldsymbol{q}^k = \langle x_w^k, \theta^k, \dot{x}_w^k, \dot{\theta}^k \rangle$
- Applied torque vector at the previous time step $k-1$: $\boldsymbol{\tau}^{k-1} = \langle \tau_{LQR}^{k-1}, \tau_\phi^{k-1}, \tau_c^{k-1} \rangle$
- The desired position and pitch angle: $\boldsymbol{C}^k = \langle C_x^k, C_\theta^k \rangle \in \boldsymbol{q}^{des}$
- History of position error and velocity: $\boldsymbol{\Theta}^k = \langle \boldsymbol{\Theta}_{x_e}^k, \boldsymbol{\Theta}_{\dot{x}}^k \rangle$ $\boldsymbol{\Theta}_{x_e}^k = \langle \Delta x_w^{k-2}, \Delta x_w^{k-1}, \Delta x_w^k \rangle$, $\boldsymbol{\Theta}_{\dot{x}}^k = \langle \dot{x}_w^{k-2}, \dot{x}_w^{k-1}, \dot{x}_w^k \rangle$

where the symbol Δ indicates the error between the desired value and the actual value. The usage of $\boldsymbol{\tau}^{k-1}$ and $\boldsymbol{\Theta}_{x_e}^k$ is motivated by previous works [25], [12]. The key to generating an end-to-end (state-to-torque) policy was found in the inclusion and usage of both $\boldsymbol{\tau}^{k-1}$ and $\boldsymbol{\Theta}_{x_e}^k$ within the residual RL framework.

The reward function was designed to track the robot's desired position x_w^{des} and pitch θ to keep the robot stable. The resulting reward function R at time k is defined as follows :

$$R(s) = -K||\boldsymbol{e}_k^s||_2 + \mathbf{1}(|err(x_w)_k| < 1) * \mathbf{1}(|err(\theta)_k| < .35) + \mathbf{1}(|err(x_w)_k| < |err'(x_w)_k|) * \mathbf{1}(|err(\theta)_k| < |err'(\theta)_k|) \quad (6)$$

where $err(x)_k = x_w^{des}(k) - x_w(k)$ and $err(\theta)_k = 0 - \theta(k)$. Indicating the change pattern of the error denotes $err'(y)_{k-1} = y^{des}(k) - y(k-1)$. The scaling factor $K = [0.1, 0.1]$ and $\boldsymbol{e}_k^s = [err(x)_k, err(\theta)_k]^T$.

Learning the Ensemble Deep Reinforcement Learning: In this work, we use 10 single SAC policy $\pi(a|\boldsymbol{s})$ ($M = 10$). A single RL policy $\pi(a|\boldsymbol{s})$ is a 3-layer multi-layer perceptron (MLP), with input $\boldsymbol{s} \in \mathbb{R}^{15}$, and output $u \in \mathbb{R}^1$ that represents wheel torque for stabilization. We trained for total of 5,00

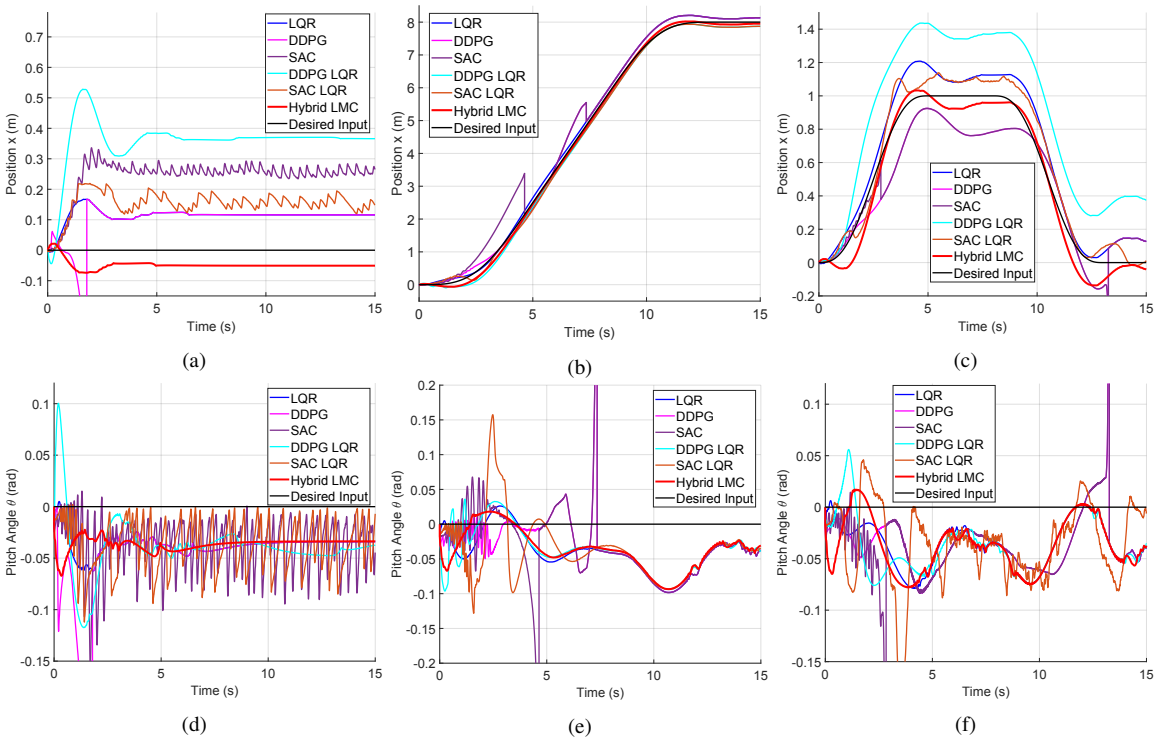


Fig. 4: **Result of Locomotion Benchmark in Different Tasks**. Two figures in each column (e.g., (a) and (e)) indicate the position tracking and balancing performance of each task. Each column corresponds to task1, 2, and 3 from the leftmost.

TABLE II: **Locomotion Performance Benchmark.** Each table shows the experimental result performed in different tasks: (1) Balancing task (2) Tracking task given 5th-order velocity trajectory (3) Tracking task given 5th-order position trajectory. Normal case is set the same physical parameter with training environment. Case 2 and 3 are set with different physical parameters. The result value indicates the average of a total of 5 trials. The number in parentheses represents the probability of the operation time was performed when it failed.

Model Settings (Mass, Gear ratio, Friction, CoM)	Metric (RMSE)	Method						
		LQR	DDPG	SAC	DDPG+LQR	SAC+LQR	BCF	Hybrid LMC
Case 1 (Normal) (4.05, 1 ,1, 0)	Position (m)	0.128	**0.025**	0.271	0.377	0.164	F	0.038
	Velocity (m/s)	0.06	0.283	0.279	0.159	0.209	F	**0.043**
	Pitch angle (rad)	0.042	0.051	0.048	0.048	0.049	F	**0.041**
Case 2 (8.05, 1.3 ,1.3, 0.12)	Position (m)	0.116	0.126 (0.12)	0.252	0.367	0.156	F	**0.05**
	Velocity (m/s)	0.05	0.456 (0.12)	0.287	0.156	0.209	F	**0.046**
	Pitch angle (rad)	0.038	0.181 (0.12)	0.045	0.047	0.044	F	**0.036**
Case 3 (14.05, 0.9 ,1.1, -0.12)	Position (m)	0.116	0.083 (0.12)	0.228	0.368	0.148	F	**0.053**
	Velocity (m/s)	0.081	0.323 (0.12)	0.289	0.177	0.205	F	**0.074**
	Pitch angle (rad)	0.039	0.146 (0.12)	0.063	0.050	0.047	F	**0.038**

Model Settings (Mass, Gear ratio, Friction, CoM)	Metric (RMSE)	Method						
		LQR	DDPG	SAC	DDPG+LQR	SAC+LQR	BCF	Hybrid LMC
Case 1 (Normal) (4.05, 1 ,1, 0)	Position (m)	0.147	0.269 (0.47)	0.482 (0.26)	0.137	0.116	F	**0.083**
	Velocity (m/s)	0.084	0.313 (0.47)	0.370 (0.26)	0.109	0.184	F	**0.074**
	Pitch angle (rad)	**0.049**	0.054 (0.47)	0.027 (0.26)	0.051	0.059	F	**0.049**
Case 2 (8.05, 1.3 ,1.3, 0.12)	Position (m)	0.136	0.285 (0.53)	0.583 (0.3)	0.146	0.122	F	**0.087**
	Velocity (m/s)	0.081	0.380 (0.53)	0.394 (0.3)	0.11	0.181	F	**0.074**
	Pitch angle (rad)	**0.046**	0.048 (0.53)	0.070 (0.3)	0.048	0.056	F	**0.046**
Case 3 (14.05, 0.9 ,1.1, -0.12)	Position (m)	0.129	0.265 (0.51)	0.05 (0.11)	0.14	0.122	F	**0.077**
	Velocity (m/s)	0.080	0.349 (0.51)	0.312 (0.11)	0.109	0.188	F	**0.073**
	Pitch angle (rad)	**0.045**	0.048 (0.51)	0.092 (0.11)	0.048	0.057	F	0.046

Model Settings (Mass, Gear ratio, Friction, CoM)	Metric (RMSE)	Method						
		LQR	DDPG	SAC	DDPG+LQR	SAC+LQR	BCF	Hybrid LMC
Case 1 (Normal) (4.05, 1 ,1, 0)	Position (m)	0.140	0.144 (0.88)	0.122 (0.3)	0.374	0.110	F	**0.060**
	Velocity (m/s)	**0.082**	0.202 (0.88)	0.336 (0.3)	0.120	0.175	F	0.087
	Pitch angle (rad)	**0.047**	0.051 (0.88)	0.091 (0.3)	0.049	0.057	F	**0.047**
Case 2 (8.05, 1.3 ,1.3, 0.12)	Position (m)	0.128	0.152 (0.88)	0.04 (0.19)	0.362	0.100	F	**0.058**
	Velocity (m/s)	**0.080**	0.209 (0.88)	0.326 (0.19)	0.120	0.179	F	0.087
	Pitch angle (rad)	**0.043**	0.048 (0.88)	0.07 (0.19)	0.046	0.054	F	**0.043**
Case 3 (14.05, 0.9 ,1.1, -0.12)	Position (m)	0.122	0.154 (0.87)	0.041 (0.2)	0.359	0.101	F	**0.046**
	Velocity (m/s)	**0.082**	0.205 (0.87)	0.213 (0.2)	0.132	0.243	F	**0.082**
	Pitch angle (rad)	0.044	0.047 (0.87)	0.084 (0.2)	0.047	0.067	F	**0.043**

epochs with each epoch consisting of a maximum of 4,000 steps. The policy was updated at every 1,000 steps. The size of our replay buffer was $1e^6$. The learning rate was set to $1e^{-3}$. The discount factor was set to 0.99. We chose ρ, for updating our target network value, to be 0.995. The entropy regularization coefficient value is chosen using an auto temperature adjustment [26]. The batch size is 64 and the Adam optimizer was utilized for all our implementations. The structure of SAC was implemented by referring the OPENAI open source [27]. A single desktop with 1 GPU (RTX3060ti) was used for training RL algorithms. Training takes roughly half day on the desktop machine.

IV. Result and Discussion

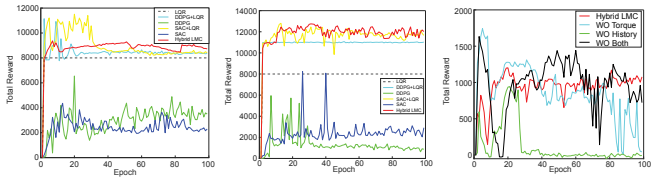
In this section we conducted three tests to gauge the performance of the proposed *Hybrid LMC*: A) A performance benchmark. B) *Hybrid LMC* with human control data test. C) An ablation study highlighting key design parameters of *Hybrid LMC*. Each experiment was conducted on different random seeds.

In section IV-A we fix our model parameters and reward function and compare against different controllers ranging from the standard LQR, model-free RLs, a residual RLs, to *Hybrid LMC*. We showcase our results and discuss why we believe *Hybrid LMC* demonstrates improved performance. In section IV-B, we highlight the performance of *Hybrid LMC* in the absence of hand-designed preconstructed trajectories, using human HMI commands for reference instead. Finally, in section IV-C we discuss an ablation study with varied design parameters, specifically using data history and previous time step torque values as an input of the network.

A. Performance Benchmark

We have summarized the performance of *Hybrid LMC* in Table II and described in Fig 4. Overall, *Hybrid LMC* achieves the highest performance (RMSE values less than 0.1) in all test cases. The proposed *Hybrid LMC* shows an average performance increase of 48% for position tracking compared to just an LQR. There was no significant improvement in tracking of the other states, $\dot{x}_w$ and θ. Interestingly, *Hybrid LMC* showed better overall performance for the 3 different tracking tasks, specified in II, despite being trained using only desired velocity trajectories and their integrated terms, as described in Fig. 3. Basic residual RL was not able to enhance performance and even performed worse than just the LQR in certain situations. Lastly, the policy with only BCF failed to generate the appropriate torques for completing the locomotion tasks and stabilizing SATYRR.

In our studies we consistently found that the LQR controller had larger position steady state error than steady state pitch error. *Hybrid LMC* is attempting to reduce the residual error by rewarding smaller deviations from the desired. Hence, we hypothesize that our policy effectively learned to complement the LQR and help reduce the largest source of error found in position tracking. As the other errors in pitch and velocity were already small, we noticed marginal improvement.



(a) (b) (c)

Fig. 5: **Learning Curves of the Total Reward.** (a) Total test reward graph given a 5th-order velocity trajectory. (b) Total test reward graph given command to stand upright in place. (c) Comparison total reward value for ablation study.

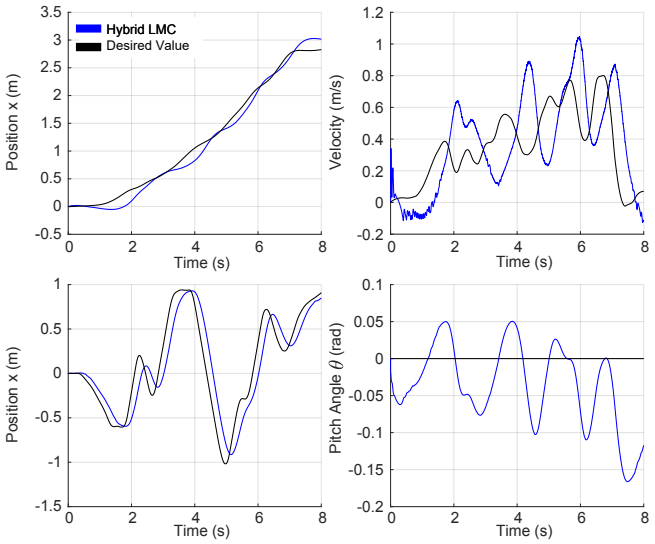


Fig. 6: **Results of the Performance of Hybrid LMC using a Human Data.** RMSE of each state is 0.124, 0.273, 0.064, 0.2419, corresponding to position, velocity, pitch angle, and yaw position.

For the balancing task, model-free DDPG had the best performance for Task 1, Case 1. In other cases, DDPG failed to successfully track the desired trajectories. Also, the performance of *DDPG+LQR* significantly decreased with changes in the model parameters (mass, friction, etc.). The stochastic approach, *SAC+LQR*, showed similar successful performance regardless of the model parameter changes. This suggests that a stochastic approach might be more advantageous in building a versatile policy against model changes for locomotion. We believe that stochastic policies promote exploration of the action space through random sampling of the output action distribution, which in turn enables discovery of potentially more ideal actions. However, methods using a single stochastic policy, i.e. SAC, are prone to having large output variance. Consequently, this degrades control performance and consistency. The performance difference between *Hybrid LMC* and *SAC+LQR* suggests that a deep ensemble RL is efficient in handling the high-variance issues highlighted above (see Fig. 4). Finally, from our comparisons of model-free RL we also see that these policies do not effectively learn end-to-end, state-to-torque values for the wheeled robot locomotion problem. These initial findings highlight the need for hybrid polices such as *Hybrid LMC*.

TABLE III: **Ablation Study for Hybrid LMC:** Performance result without a key component.

	Hybrid LMC	Metric (RMSE)		
		Position (m)	Velocity (m/s)	Pitch angle (rad)
Task1	W/O history data	0.132	0.074	0.042
	W/O LQR torque	0.108	0.067	**0.041**
	W/O Both	0.149	0.095	0.042
	W Both	**0.032**	**0.048**	**0.041**
Task2	W/O history data	0.134	0.09	**0.049**
	W/O LQR torque	0.146	0.09	**0.049**
	W/O Both	0.166	0.113	0.05
	W Both	**0.079**	**0.076**	**0.049**

TABLE IV: Verification with different parameters of LQR

	LQR parameter $K = [K_{xW}, K_{th}, K_{dxW}, K_{dth}]$	Metric (RMSE)	Method	
			LQR	HybridLC
Task 2	$[-150, -350, -50, -50]$	Position (m)	0.1	**0.058**
		Velocity (m/s)	0.069	**0.059**
		Pitch angle (rad)	0.046	**0.045**
	$[-50, -200, -20, -20]$	Position (m)	**0.169**	0.204
		Velocity (m/s)	**0.096**	0.114
		Pitch angle (rad)	**0.047**	0.049
	$[-25, -100, -10, -10]$	Position (m)	**0.162**	0.519
		Velocity (m/s)	**0.107**	0.289
		Pitch angle (rad)	**0.048**	0.211

All learning curves are shown in Fig. 5. We trained all models until they converged and the total reward no longer increased. The methods based on residual RL (e.g. *SAC+LQR* and *Hybrid LMC*) showed faster convergence than their model-free counterparts (SAC, DPPG). These hybrid controllers converged in approximately 20 epochs. Compared to *Hybrid LMC*, we found that *SAC+LQR* achieved a larger reward value but also had larger variance here. This high variance can result in undesired noisy output actions as shown in Fig. 4.

Hybrid LMC consistently showed improved performance in reducing residual error in simulation, but application to real hardware presents a final step in evaluating the efficacy and performance of the *Hybrid LMC*. We look forward to hardware implementation in future studies. To bridge the sim-to-real gap, we plan on utilizing an environmental encoder, a teacher policy [14], [13], and domain randomization techniques. We believe that a LQR (ideally tuned for hardware) within the *Hybrid LMC* will guide the policy in realizing more realistic torques and reduce the risk of undesirable actuation while exploring in the beginning stages of training.

B. Verifying Hybrid LMC with a Human Data

The goal of this experiment was to test the performance of *Hybrid LMC* for locomotion and tracking when given human command signals obtained directly from hardware as a step toward teleoperation (e.g.,HERMES humanoids [1]). This test scenario is important as it suggests the viability of using reference trajectories that are irregular and rapidly changing compared to those used in training - 5^{th} order velocity polynomials. The recorded human data - body tilt and twist - acquired from the Human Machine Interface [24] is mapped to the desired reference vector $\boldsymbol{q}_{des}$ and used for tracking here as seen in Fig. 3. Our three main findings were: 1) Compared to the standard LQR with human commands, the *Hybrid LMC* with human commands had a 20% improvement in position tracking. 2) Compared to the *Hybrid LMC* with preconstructed 5^{th} order polynomials, the *Hybrid LMC* with human commands presented slight degradation in position tracking. We believe that because irregular signals - such as those from the human - have high variance, they can negatively affect the RL policy's rate of convergence. 3) The improved *Hybrid LMC* performance was consistent despite being trained using only velocity trajectories. All results can be seen in Fig. 6)

C. Ablation Study and Analysis of Hybrid LMC

Performance comparison of key components: An ablation study is conducted to investigate and analyze how crucial components of the network affect the performance of *Hybrid LMC*. Based on our results we believe that feeding previous torque commands $\tau_{LQR}^{k-1}, \tau_{\phi}^{k-1}, \tau_{c}^{k-1}$ and a history of the states $\Theta_{x_e}, \Theta_{\dot{x}}$ as an input to *Hybrid LMC* is critical for achieving better performance, as seen in Table III. Here, *Hybrid LMC* is trained with the velocity trajectory, Fig. 5(c). Each component has a considerable impact on the convergence of the network and in achieving better performance. We hypothesize that leveraging the history state and the previous torque brings benefits to end-to-end (state-to-torque) learning in the broad residual RL framework, also utilized in *Hybrid LMC*.

Verifying Hybrid LMC with varied LQR gains: Here we test the dependency of *Hybrid LMC* on the LQR. The *Hybrid LMC* was trained using the original gains (described in section IV) but tested using varied LQR gains. From row 1 of Table IV, we see that increasing LQR gains results in similar performance and superiority of the *Hybrid LMC*. However, decreasing the gains generally resulted in an increase in RMSE error and lead to worse performance. This suggests that high performance of *Hybrid LMC* is dependant on the tuning and response of the feedback controller during training. In other words, switching the gains or feedback controller in deployment is not recommended. We believe that varying the gains randomly during training may help address this issue [13].

V. CONCLUSION

In this paper, we propose a hybrid learning and model-based controller, *Hybrid LMC*, that combines the strength of a conventional model-based LQR and deep reinforcement learning for more robust tracking in the presence of model uncertainty and parameter changes. Moreover, the ensemble deep reinforcement approach can augment the performance of a standard controller while reducing the variance of a single stochastic-based RL policy. In this manner we are able to perform end-to-end learning directly. By incorporating the ensemble deep RL and the LQR controller, LQR guides the RL policy in generating more appropriate torque within a bounded range, and results in an increase in the sampling efficiency. The ablation studies were conducted and analyzed

carefully, to offer a proper method of using *Hybrid LMC*. In all experiments, *Hybrid LMC* outperforms the previous methods and demonstrates generalized performance improvement in the presence of model changes and irregular desired trajectories.

In future works, we will apply *Hybrid LMC* on hardware to a wheeled humanoid robot system, SATYRR, to verify the performance of *Hybrid LMC* in the physical world. Based on our result, we expect that *Hybrid LMC* provides an efficient way to train the real system safely while enhancing the performance.

ACKNOWLEDGEMENTS

The authors are grateful to Youngwoo Sim and Guillermo Colin for their support in designing the figure and discussion for the paper.

REFERENCES

[1] Albert Wang, Joao Ramos, John Mayo, Wyatt Ubellacker, Justin Cheung, and Sangbae Kim. The hermes humanoid system: A platform for full-body teleoperation with balance feedback. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 730–737. IEEE, 2015.

[2] Joao Ramos and Sangbae Kim. Dynamic locomotion synchronization of bipedal robot and human operator via bilateral feedback teleoperation. *Science Robotics*, 4(35):eaav4282, 2019.

[3] Victor Klemm, Alessandro Morra, Ciro Salzmann, Florian Tschopp, Karen Bodie, Lionel Gulich, Nicola Küng, Dominik Mannhart, Corentin Pfister, Marcus Vierneisel, et al. Ascento: A two-wheeled jumping robot. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7515–7521. IEEE, 2019.

[4] Xu Li, Haitao Zhou, Songyuan Zhang, Haibo Feng, and Yili Fu. Wlr-ii, a hose-less hydraulic wheel-legged robot. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4339–4346. IEEE, 2019.

[5] Victor Klemm, Alessandro Morra, Lionel Gulich, Dominik Mannhart, David Rohr, Mina Kamel, Yvain de Viragh, and Roland Siegwart. Lqr-assisted whole-body control of a wheeled bipedal robot with kinematic loops. *IEEE Robotics and Automation Letters*, 5(2):3745–3752, 2020.

[6] Songyan Xin and Sethu Vijayakumar. Online dynamic motion planning and control for wheeled biped robots. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3892–3899. IEEE, 2020.

[7] Mert Önkol and Coşku Kasnakoğlu. Adaptive model predictive control of a two-wheeled robot manipulator with varying mass. *Measurement and Control*, 51(1-2):38–56, 2018.

[8] Munzir Zafar, Seth Hutchinson, and Evangelos A Theodorou. Hierarchical optimization for whole-body control of wheeled inverted pendulum humanoids. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7535–7542. IEEE, 2019.

[9] Ruben Grandia, Andrew J Taylor, Andrew Singletary, Marco Hutter, and Aaron D Ames. Nonlinear model predictive control of robotic systems with control lyapunov functions. *arXiv preprint arXiv:2006.01229*, 2020.

[10] Leilei Cui, Shuai Wang, Jingfan Zhang, Dongsheng Zhang, Jie Lai, Yu Zheng, Zhengyou Zhang, and Zhong-Ping Jiang. Learning-based balance control of wheel-legged robots. *IEEE Robotics and Automation Letters*, 6(4):7667–7674, 2021.

[11] Tuomas Haarnoja, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine. Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*, 2018.

[12] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019.

[13] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

[14] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

[15] Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6023–6029. IEEE, 2019.

[16] Krishan Rana, Vibhavari Dasagi, Ben Talbot, Michael Milford, and Niko Sünderhauf. Multiplicative controller fusion: Leveraging algorithmic priors for sample-efficient reinforcement learning and safe sim-to-real transfer. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6069–6076. IEEE, 2020.

[17] Krishan Rana, Vibhavari Dasagi, Jesse Haviland, Ben Talbot, Michael Milford, and Niko Sünderhauf. Bayesian controller fusion: Leveraging control priors in deep reinforcement learning for robotics. *arXiv preprint arXiv:2107.09822*, 2021.

[18] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[19] Hua Chen, Bingheng Wang, Zejun Hong, Cong Shen, Patrick M Wensing, and Wei Zhang. Underactuated motion planning and control for jumping with wheeled-bipedal robots. *IEEE Robotics and Automation Letters*, 6(2):747–754, 2020.

[20] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

[21] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[22] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[23] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[24] Sunyu Wang, Kevin Murphy, Dillan Kenney, and Joao Ramos. A comparison between joint space and task space mappings for dynamic teleoperation of an anthropomorphic robotic arm in reaction tests. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2846–2852. IEEE, 2021.

[25] Atil Iscen, Ken Caluwaerts, Jie Tan, Tingnan Zhang, Erwin Coumans, Vikas Sindhwani, and Vincent Vanhoucke. Policies modulating trajectory generators. In *Conference on Robot Learning*, pages 916–926. PMLR, 2018.

[26] Karl Pertsch, Youngwoon Lee, and Joseph J Lim. Accelerating reinforcement learning with learned skill priors. *arXiv preprint arXiv:2010.11944*, 2020.

[27] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.