

Adapting Rapid Motor Adaptation for Bipedal Robots

Ashish Kumar^{*1}, Zhongyu Li^{*1}, Jun Zeng¹, Deepak Pathak², Koushil Sreenath¹, Jitendra Malik¹

Abstract—Recent advances in legged locomotion have enabled quadrupeds to walk on challenging terrains. However, bipedal robots are inherently more unstable and hence it’s harder to design walking controllers for them. In this work, we leverage recent advances in rapid adaptation for locomotion control, and extend them to work on bipedal robots. Similar to existing works, we start with a base policy which produces actions while taking as input an estimated extrinsics vector from an adaptation module. This extrinsics vector contains information about the environment and enables the walking controller to rapidly adapt online. However, the extrinsics estimator could be imperfect, which might lead to poor performance of the base policy which expects a perfect estimator. In this paper, we propose A-RMA (Adapting RMA), which additionally adapts the base policy for the imperfect extrinsics estimator by finetuning it using model-free RL. We demonstrate that A-RMA outperforms a number of RL-based baseline controllers and model-based controllers in simulation, and show zero-shot deployment of a single A-RMA policy to enable a bipedal robot, Cassie, to walk in a variety of different scenarios in the real world beyond what it has seen during training. Videos and results at <https://ashish-kmr.github.io/a-rma/>

I. INTRODUCTION

The ability of legged animals to traverse a wide variety of terrains has inspired decades of research in legged robots to replicate this capability in artificial systems. Although great advances have been made via control theoretic methods [1]–[14], designing these controllers often requires expert parameter tuning for different terrain. With recent advances in learning, research interest has shifted towards methods that *learn* to walk. Impressive results have been achieved in the past few years, notably Lee et.al. [15] and RMA [16], for general-purpose legged locomotion in challenging real-world terrains. However, these results have largely been shown in quadrupeds which are relatively easier to control than bipedal robots.

Our goal is to see how well a learning-based approach works for bipedal robots. Unlike their quadruped or hexapod counterparts, bipeds are more dynamic and offer coverage of more terrains at the cost of being more prone to instability. Although there have been several investigations of applying learning methods to bipedal robots [17], [18], until now, the robustness performance has not matched their quadruped counterparts in terms of robustness to scenarios not seen during training (Figure 1).

We train our robot in simulation and start with Rapid Motor Adaptation (RMA) [16] as a baseline for sim2real

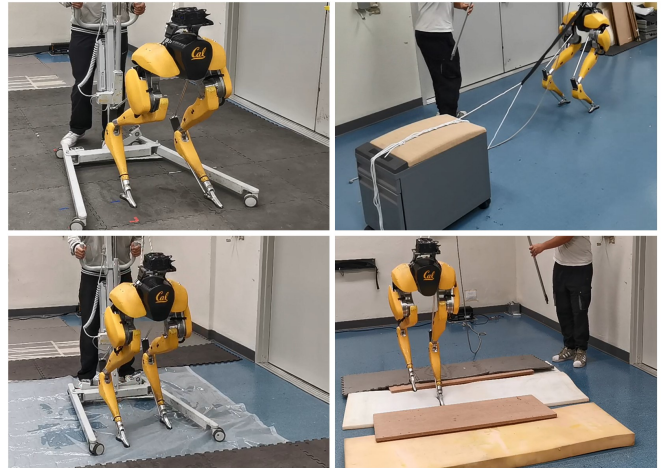


Fig. 1: We demonstrate the performance of A-RMA with a bipedal robot in several challenging setups which includes slippery surfaces, foam, and pulling a payload. Some of these were never seen during training and the policy was deployed without any calibration or real-world finetuning. Note that bipeds are inherently more unstable compared to quadrupeds, which makes each of these much more challenging than for a quadruped robot. Videos at <https://ashish-kmr.github.io/a-rma/>

transfer to the Cassie biped. In contrast to the popular idea of training a terrain-invariant policy [19], RMA trains an adaptive policy conditioned on latent *extrinsics* vector that encodes terrain-specific information in simulation. After training this base policy, the extrinsic vector is then estimated online by an adaptation module using the history of observed proprioceptive states. However, we found that scaling RMA as-is to complex bipeds is faced with several challenges.

The first challenge is training the base policy from scratch using reinforcement learning. To address this issue and learn naturally stable high-performance gaits, we bootstrap the base policy from a set of reference motions generated using a gait library. The second challenge, however, is a more fundamental one. For complex robots like bipeds, it is usually very challenging to precisely estimate the privileged extrinsics at deployment just from the observable states. This creates a large domain gap for the base policy at deployment which was trained with accurate extrinsics. The higher the magnitude of extrinsics estimation error, the worse the base policy performs. We bridge this gap by another round of finetuning the base policy using the imperfect extrinsics estimated from adaptation module instead of conditioning on the perfect extrinsics. We call our method *A-RMA* due to the adaptation of RMA policy itself, as outlined in Figure 2. Similar to the base policy and adaptation module in RMA,

^{*} Authors contributed equally.

¹ University of California Berkeley, {ashish.kumar, zhongyu.li, zengjun-sjt, koushils, malik}@berkeley.edu

² Carnegie Mellon University, dpathak@cs.cmu.edu

A-RMA *policy adaptation* step is also trained in simulation. We evaluate A-RMA on a complex Cassie bipedal robot and show evaluations on several challenging environments in both simulation and real world.

II. RELATED WORK

a) Model-based Control for Bipedal Robots: Locomotion of bipedal robots has been traditionally approached via notions of gait stability such as ZMP criterion [1] or Capturability [3] using robot’s reduced-order models [2], [4]–[6], [10], [20]. Although these methods can effectively control humanoid robots with flat feet, they often walk conservatively. Alternatively, Hybrid Zero Dynamics (HZD) [7], [9], [11]–[13] based techniques can also generate stable periodic walking based on input-output linearization using robot’s full-order model. However, HZD-based controllers for 3D robots typically need extensive parameter tuning in both simulation and in the real world, and are hard to adapt to the environment changes. In this work, we use reinforcement learning to learn our walking controllers, using the HZD-based walking gaits only as reference motions to produce natural looking gaits. This allows us to generate more diverse and effective stable walking behaviours since we don’t have hard constraints on periodicity and neither do we require a precise model.

b) RL-based Control for Legged Robots: Reinforcement learning for legged locomotion has shown promising results in learning walking controllers that can be successfully deployed in the real world [16], [17], [21]–[23]. Data-driven methods could either use reference motions [17], [23] to learn walking behaviours, learn residuals over predefined foot motions [15], [24], or learn without any motion priors and foot trajectory generators [16], [22], [25]. In this work, we use a hybrid approach where we warm start the learning process using a HZD-based gait library, but subsequently reduce the dependency on them by lowering the costs related to imitating the gait library.

Recent works have also built bipedal controllers using RL. Model-based RL in [26], [27] is used to learn a walking controller on Cassie in simulation to track a velocity. Alternatively, work in [28], [29] learns model free residuals over reference motions, while work in [30] learns walking policies that can track a commanded planar velocity on Cassie in the real world. Although residual control speeds up training, the corrections that can be applied to the reference trajectory is limited which limits the diversity of behaviours of the controller. This can be addressed by learning policies which don’t rely on reference motions during deployment, but instead add motion constraints during training by either using reference motions or through foot fall constraints via reward terms [17], [31], [32]. We follow this general approach in this paper. More specifically, we use reference trajectories for imitation via reward terms, except that we only use them to warm start the learning process and then reduce the dependency on them and use energy minimization [33] to allow for diverse behaviors which can be more general than the reference motions.

A) Training in Simulation

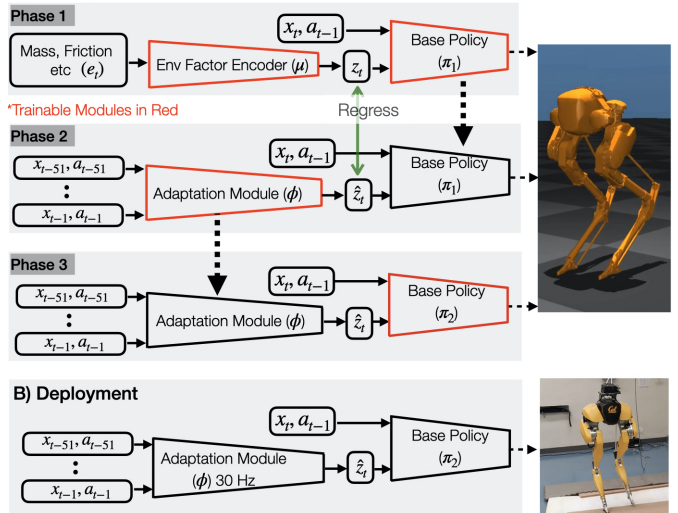


Fig. 2: We show the training and deployment phases of A-RMA. The first two phases are the same as RMA [16]. We additionally add a third phase in which the base policy is fine-tuned again with PPO while keeping the adaptation module fixed, to account for imperfect estimation of extrinsics. We found this to be critical for reliable performance in the real world.

c) Simulation to Real World Transfer: Sim-to-real methods allow deployment of walking policies in the real world after training them in simulation, which is a safe and inexpensive data source. Parameters finetuning on the hardware is one way to bridge the gap for model-based methods [11], [12], [34]. Alternatively, several works use domain randomization which varies the system properties in simulation in order to cover the uncertainty in the real world [17]–[19], [23], [31], [32], [35]–[38]. These methods achieve robustness at the cost of optimality, where they try to learn a single environment-agnostic behaviour for all deployment scenarios. Instead, we learn an adaptive policy which enables transfer to widely varying deployment scenarios.

d) System Identification and Adaptation: An adaptive policy conditions to environment variations instead of being agnostic to them. These variations can be explicitly estimated during deployment either through a module that is trained in simulation [39] or can be estimated by optimizing for high returns using evolutionary algorithms [40]. Predicting the exact parameters is both unnecessary and difficult, which in turn leads to poor empirical performance. Instead, a low dimensional latent embedding can be used [16], [23], [41] which contains an implicit estimate of the environment. At test time, this latent embedding can be optimized using policy gradients from real-world rollouts [23], Bayesian optimization [42], or random search [38]. An alternate approach is to use meta learning to learn to adapt online [43]. These methods tend to take multiple rollouts to adapt in the real world [44], [45] which is prohibitive for several scenarios. In this work, we follow the approach of [16] which learns a feed forward policy to estimate the latent environment vector, enabling rapid adaptation in fractions of a second.

III. GENERAL WALKING CONTROLLER

Our walking controller contains a base policy which produces the target joint points for the robot, and an adaptation module which uses proprioceptive history to continually estimate the extrinsics vector. Such a vector contains information about the environment that can be used by the base policy to adapt. In this section, we describe how we train the base policy for bipeds since trivially training the policy from scratch results in unnatural gaits, unlike prior work in quadrupeds [16]. Instead we use reference motions to bootstrap learning but unlike prior methods which use reference motions [15], [17], we reduce the dependence on these reference motions as training progresses to learn optimal behaviors in a data driven way.

A. Walking Policy

We train an adaptive walking policy (π_1) using model free reinforcement learning to learn a stable walking policy while imitating reference motions generated using a HZD-based gait library. Here, the gait library consists of a collection of periodic walking gaits at different walking velocities and walking heights. The base walking policy (π) takes two arguments as input 1) the current state x_t , 2) the extrinsics vector $z_t \in \mathbb{R}^8$ and then predicts the next action a_t . The predicted action a_t is the target position for the 10 actuated robot joints which is converted to torque using a PD controller. The extrinsics vector z_t is a compressed version of the environment vector $e_t \in \mathbb{R}^{147}$ generated by μ . The environment vector e_t includes things such as friction, inertia, center of mass, etc (see Sec III-C), but the extrinsics vector z_t only retains as much information from e_t as is necessary for the base policy to adapt. We thus have,

$$z_t = \mu(e_t), \quad (1)$$

$$a_t = \pi_1(x_t, a_{t-1}, g_t, z_t). \quad (2)$$

We implement μ and π_1 as MLPs and jointly train them end-to-end using model-free reinforcement learning to maximize the following expected return of the policy π :

$$J(\pi) = \mathbb{E}_{\tau \sim p(\tau|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t r_t \right],$$

where $\tau = \{(x_0, a_0, r_0), (x_1, a_1, r_1) \dots\}$ is the trajectory of the agent when executing policy π_1 , and $p(\tau|\pi_1)$ represents the likelihood of the trajectory under π_1 .

B. Reward Function

The reward function encourages the agent to track the goal commanded and to imitate the reference motion from the gait library. Let's denote the robot's actual motor positions as \hat{q}_m , pelvis translation position and velocity as \hat{q}_p and $\hat{\dot{q}}_p$, the pelvis rotation and rotation velocity as \hat{q}_r and $\hat{\dot{q}}_r$. Let the corresponding quantities for the reference motion be denoted by q_m^r , q_p^r , \dot{q}_p^r , q_r^r and \dot{q}_r^r respectively. We additionally denote the robot's torque as u and the robot's ground reaction force as F . The reward at each time step t is a weighted sum of:

- 1) Motor imitation: $\exp[-\rho_1 \|q_m^r - \hat{q}_m\|_2^2]$

- 2) Pelvis position imitation: $\exp[-\rho_2 \|q_p^r - \hat{q}_p\|_2^2]$

- 3) Pelvis velocity imitation: $\exp[-\rho_3 \|\dot{q}_p^r - \hat{\dot{q}}_p\|_2^2]$

- 4) Pelvis rotation imitation: $\exp[-\rho_4 (1 - \cos(q_r^r - \hat{q}_r))]$

- 5) Rotational velocity imitation: $\exp[-\rho_5 \|\dot{q}_r^r - \hat{\dot{q}}_r\|_2^2]$

- 6) Torque penalty: $\exp[-\rho_6 \|u\|_2^2]$

- 7) Ground reaction force penalty: $\exp[-\rho_7 \|F\|_2^2]$,

where the corresponding weights of each of the reward terms are $[0.3, 0.24, 0.15, 0.13, 0.06, 0.06, 0.06]^T$, and ρ_1 to ρ_7 are $[5.0, 0.1, 5.0, 5.0, 1.0, 5e^{-7}, 1.25e^{-5}]$. To stabilize the pelvis, we set the desired roll and pitch velocity to 0, and the desired yaw velocity is from the user command $c(t)$. We compute the desired pelvis translational and rotational positions by integrating the corresponding desired velocity. Moreover, we decay the coefficients of the imitation terms as the training progresses to reduce the dependence on reference motions and learn optimal data driven behaviours.

C. Environment Variations and Terrains

During the learning of the walking policy, we vary the ground friction, robot's motor friction, mass and center of mass of the robot and robot links. We train over fractal terrain, flat terrain, slopes and discrete terrains such as steps (see Table I).

IV. WALKING CONTROLLER WITH RAPID ADAPTATION

Following the process described in the previous section, we now have a base policy which can use the extrinsics vector to adapt. However, this extrinsics vector uses privileged simulation information and is unavailable in the real world. To resolve this, we generalize the idea proposed in [16] of learning an adaptation module using the history of commanded actions and robots proprioception to estimate these extrinsics online. The insight is that the discrepancy between what was commanded to the robot and the actual joint positions contains information about these extrinsics. However, such an estimation could be noisy as the extrinsics vector might not be fully observable. To fix this, we further finetune the base policy to learn to walk adaptively with an imperfect extrinsics vector. We now describe this in detail below.

A. Adaptation Module for Estimating Extrinsics

Privileged environment information e_t is not available in the real world, and consequently its encoded extrinsics vector z_t is not accessible during deployment. Similar to [16], we estimate the extrinsics online from the proprioceptive history $(x_{t-k:t-1}, a_{t-k:t-1})$ using the adaptation module ϕ . In our experiments, we use $k = 70$ which roughly corresponds to 2s. The estimated extrinsic vector is thus given by,

$$\hat{z}_t = \phi(x_{t-k:t-1}, a_{t-k:t-1}).$$

We can train the adaptation module in simulation via supervised learning to minimize: $\text{MSE}(\hat{z}_t, z_t) = \|\hat{z}_t - z_t\|^2$, where $z_t = \mu(e_t)$. We model ϕ as a 1-D CNN to capture temporal correlations.

We collect trajectories by unrolling the base policy π_1 with the \hat{z}_t predicted by the randomly initialized function ϕ , and

then pair it with the ground truth z_t to train ϕ . We iteratively repeat this until convergence.

B. Finetuning with Estimated Extrinsic

For our setting, the extrinsics estimated by the adaptation module ϕ is imperfect since z_t might not be fully observable from the proprioception history. We observed this as the regression error to estimate z_t did not become low enough during the training of the adaptation module. This causes the base policy π_1 to experience a significant drop in performance since it is trained with perfect extrinsics, and noisy estimation introduces a domain gap. To overcome this issue, we propose to further finetune the base policy π_1 with the imperfect extrinsics predicted by the adaptation module ϕ trained in phase 2, see Fig. 2. The adaptation module is kept frozen and the base policy is finetuned using model free reinforcement learning with the reward described in Section III-B. This gives us the final base policy π_2 which is used in deployment.

V. EXPERIMENTAL SETUP

A. Hardware

We use the Cassie robot for our experiments. It is a dynamic, life-sized, and underactuated bipedal robot which has 20 DoFs and is introduced in details in [12, Sec. II]. It has 10 actuated rotational joints $q_m = [q_{1,2,3,4,7}]^T$ (abduction, rotation, hip pitch, knee, and toe motors), and four passive joints $q_{5,6}^{L/R}$ (shin and tarsus joints). Its floating base, the robot pelvis $q_p = [q_x, q_y, q_z, q_\psi, q_\theta, q_\phi]^T$, has 6 DoFs for sagittal q_x , lateral q_y , vertical q_z , roll q_ψ , pitch q_θ , and yaw q_ϕ , respectively. The entire robot coordinate $q \in \mathbb{R}^{20}$ includes all robot DoFs whereas the observable DoFs $q^o \in \mathbb{R}^{17}$ excludes the pelvis translational position $q_{x,y,z}$ which can not be reliably obtained by the sensors on the robot hardware. There is an onboard Intel NUC on the robot, which runs our policy. The policy updates at 30 Hz and the joint-level PD controller operates at 2 kHz.

B. Simulation Setup

We use the simulation environment for reinforcement learning on Cassie developed in [17], which itself is based on an open source MuJoCo simulator [46], [47]. During training, each episode has a maximum number of time steps of $T=2500$ (83 s). In each episode, we resample a new command $c(t) = [\dot{q}_x^d, \dot{q}_y^d, q_z^d, \dot{q}_\phi^d]^T$ and randomize the environment parameters every 8 s. The randomization range of the command is between $[-1.0, -0.3, 0.65, -30^\circ]^T$ and $[1.0, 0.3, 1.0, 30^\circ]^T$. The episode will terminate early if the pelvis height falls below 0.55 m, or if the tarsus joints $q_6^{L/R}$ hit the ground [17].

C. State-Action Space

a) *Observation:* The observation $\mathbf{x}_t = (q_{t-4:t}^o, \mathbf{a}_{t-4:t-1}, \mathbf{q}_{m,t}^r, \mathbf{q}_{m,t+1}^r, \mathbf{q}_{m,t+4}^r, \mathbf{q}_{m,t+7}^r, c_t)$ at time t consists of 4 parts. There are observable robot states $\mathbf{q}^o = [q^o, \dot{q}^o]$ at the current and the past 4 time steps, and the actions \mathbf{a} from the past 4 time steps. There is

Parameter	Range	Unit
Link Mass	$[0.7, 1.3] \times \text{default}$	kg
Link Mass Center	$[0.7, 1.3] \times \text{default}$	m
Joint Damping	$[0.3, 4.0] \times \text{default}$	Nms/rad
Spring Stiffness	$[0.95, 1.05] \times \text{default}$	Nm/rad
Ground Friction Ratio	$[0.3, 3.0]$	1
Fractal Terrain Height	$[0.0, 0.12]$	m

TABLE I: The range of dynamics parameters that are extended from [17, Table II] and are varied in the phase 1 of A-RMA.

also reference motion $\mathbf{q}_m^r = [q_m^r, \dot{q}_m^r]$ which includes the reference motor position and motor velocity at current time t and future time steps (next 1, 4, 7 time steps) obtained from a HZD-based gait library [17]. Moreover, the command c_t is also part of the observation. c_t includes desired sagittal and lateral walking speed, walking height, and turning yaw rate ($c_t = [\dot{q}_x^d, \dot{q}_y^d, q_z^d, \dot{q}_\phi^d]$).

b) *Action:* The action $\mathbf{a}_t = q_m^d$ is the target position for the 10 actuated motors on Cassie which are first passed through a low-pass filter [17] before being sent to the PD controller to generate the torque u for each motor.

D. Environment and Terrain variations

We randomize the environment parameters that includes robot modelling errors, sensor noise and delay, and terrain variations during the training. Based on previous work [17], the range of the randomization that is newly introduced is presented in Table. I. Furthermore, we also introduce fractal-like terrain variations to make the policies robust.

E. Training Details

a) *Base Policy and Environment Encoder:* The base policy (π) is a MLP with 2 hidden layers of size 512 each. The environment encoder (μ) has 1 hidden layer of size 256, and output (z) size of 8. We train the policy with PPO using a batch size of 65536 and minibatch size of 8192.

b) *Adaptation Module:* The adaptation module (ϕ) contains 3 convolution layers (kernels sizes 8, 5, 5 and strides 4, 1, 1 respectively) followed by 1 hidden layer of size 256. The output layer (\hat{z}) is 8 dimensional. We train the policy with Adam optimizer using the same batch size, for a total of 2000 iterations, resampling a new batch in every iteration.

c) *Base Policy Finetuning Stage:* In the last finetuning stage of A-RMA, we freeze the adaptation module and finetune the base policy using PPO with a batch size of 65536 and minibatch size of 8192 for 2000 iterations.

VI. RESULTS AND ANALYSIS

We first validate A-RMA in both MuJoCo and MATLAB Simulink (a high-fidelity simulator), and then present the results of RMA deployed on a bipedal robot Cassie in the real world. All A-RMA results in simulation and the real world are from the same policy which was only trained in MuJoCo and was deployed in MATLAB Simulink and the real world without any finetuning during test time. In this paper, we focus more on the control performance with a nominal walking height $q_z = 0.98$ m, *i.e.*, we do not change the walking height q_z during the test though the policy is trained with variable walking height.

	Min Friction	Feasible Cmd Range	Tracking Error
HZD Controller [12]	0.3	39/147	[0.1554, 0.0811]
Robust MLP [17]	0.3	147/147	[0.0869, 0.1150]
RMA [16]	0.2	147/147	[0.0908, 0.0965]
A-RMA (Ours)	0.2	147/147	[0.0849, 0.0952]

TABLE II: **Comparison of A-RMA with baseline controllers in MATLAB Simulink.** We observe that all RL-controllers can track the entire command range. Of these, A-RMA and RMA generalize beyond the training friction range, maintaining stability in as low as a 0.2 friction coefficient. A-RMA has the best tracking performance in both $[\dot{q}_x^d, \dot{q}_y^d]$ compared to all the baseline controllers.

	MTTF (s)	Return	Tracking Err	Mean Jerk
A-RMA-static	7.1	158.5	0.79	0.64
RMA [16]	11.4	273.6	0.29	0.92
A-RMA	14.0	335.8	0.30	0.30
A-RMA-priv	14.2	340.0	0.27	0.28

TABLE III: **Comparison of A-RMA with baseline controllers in MuJoCo.** We observe that A-RMA’s performance is very close to A-RMA-priv, which has access to privileged simulation information, and is better than all the baseline controllers for the metrics: MTTF (max episode length = 20s), Returns and Mean Jerk experienced. The tracking performance (Mean Tracking Error) is similar to RMA. Note that RMA experiences a large drop compared to A-RMA-priv because of unobservability of the entire extrinsics vector from proprioception. A-RMA-static sees a sharp drop in performance compared to A-RMA, validating the importance of a continually and rapidly updating extrinsics vector.

A. Simulation Validation

The Simulink simulator provides us a safe validation environment to compare locomotion controllers while providing very high fidelity dynamics which is much more accurate than MuJoCo. We additionally show comparisons of A-RMA to its other adaptive variants in MuJoCo. Please note that the agent has no access to the Simulink data during the policy training in MuJoCo. We benchmark the A-RMA with the following locomotion controllers on Cassie as baselines: 1) a HZD-based controller (HZD) developed in [12] based on [11], 2) a robust RL-based locomotion controller represented by MLP (Robust MLP) developed in [17], 3) A-RMA-static in which we estimate the latent in the first time step and then freeze it for the rest of the episode, 4) RMA [16]. Finally as oracle, we also show the performance of A-RMA-priv which has access to privileged simulation information for reference. Please note that, for all the RL-based controllers, we utilize the same formulation of rewards and range of dynamics randomization presented in Sec. V-D during training.

We benchmark these controllers using the following metrics: 1) the converged return, 2) command tracking performance on a nominal ground, 3) Minimum friction successfully handled, 4) Mean time to Fall (MTTF), 5) Mean Jerk of all the joints. In MuJoCo, the reported metrics are computed over 10 episodes with an episodic timeout of 20s during which we randomize all the parameters shows in Table I.

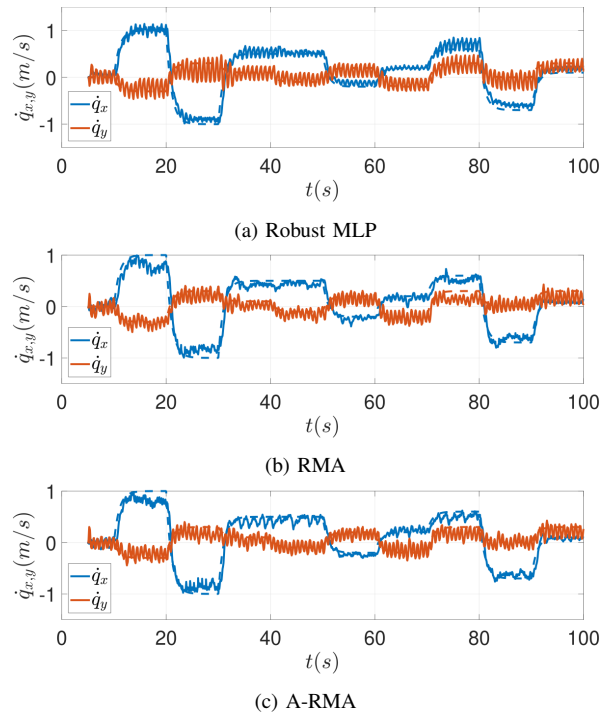


Fig. 3: Plot of tracking error in sagittal and lateral directions different controllers. RMA and A-RMA produce less oscillations than the Robust MLP once the lateral velocity has stabilized.

a) *Converged Return and MTTF:* We report the performance of A-RMA-priv to understand the maximum achievable performance. As shown in Table III, going from A-RMA-priv to RMA shows a sharp performance fall in both the metrics. This sharp fall is a consequence of non-negligible phase 2 regression loss as the extrinsics vector might not be fully observable from the proprioception history. We show that the proposed A-RMA recovers this performance drop and approximately matches the performance of the A-RMA-priv. We additionally see that A-RMA-static has a substantial performance fall indicating the importance of rapid online adaptation via estimated extrinsics.

b) *Command Tracking:* We compare the different locomotion controllers on command tracking performance where we sample different desired sagittal walking speed \dot{q}_x^d and lateral walking speed \dot{q}_y^d . The command tracking performance includes two parts: the range of the command for which a controller is able to maintain gait stability (Feasible Command Set [17]), and tracking error between the desired and actual robot walking velocities.

We test the range of command $[\dot{q}_x^d, \dot{q}_y^d]^T$ from $[-1.0, -0.3]^T$ to $[1.0, 0.3]^T$ with a resolution of $[0.1, 0.1]^T$ with a nominal walking height $q_z = 0.98$ on Cassie with different locomotion controllers in simulation. These count for 147 different commands and a command is considered feasible if the controller can maintain stability for 10 seconds. As demonstrated earlier in [17], HZD controller is only able to stably track a limited number of commands at nominal walking height. Specifically, the lateral walking speed cannot exceed 0.1 m/s while walking forwards, and

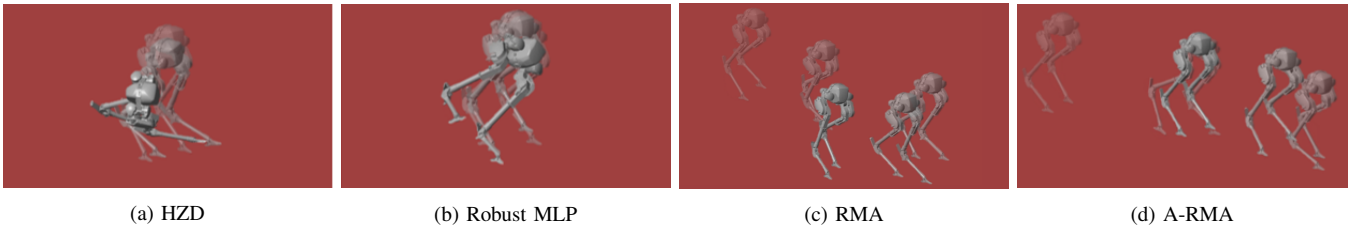


Fig. 4: Trajectories of Cassie on slippery ground with friction coefficient of 0.2. We observe that A-RMA has minimal lateral deviation and is more stable than RMA, which shows significant deviation in the lateral walking direction.

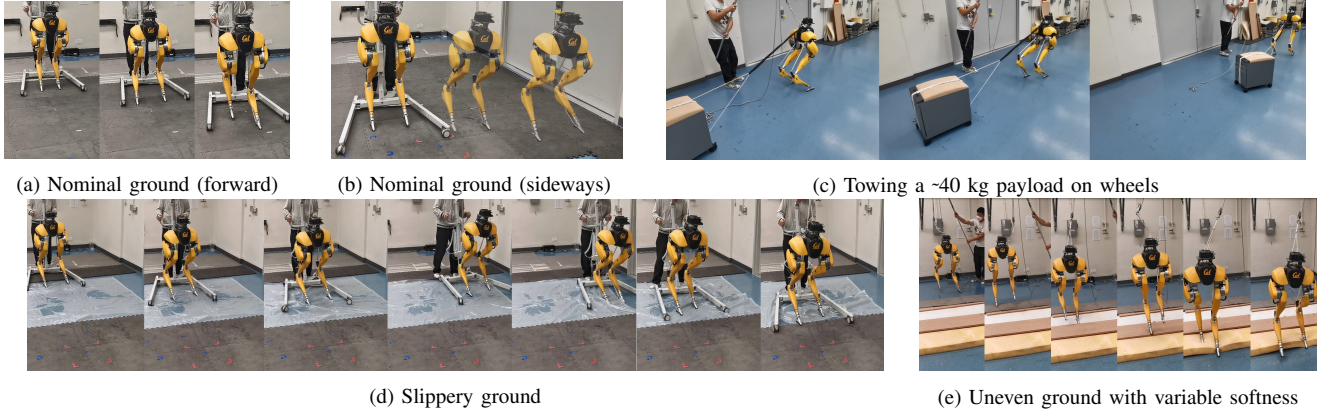


Fig. 5: Real world deployment of Cassie walking on nominal ground, towing a payload, walking on a slippery surface, and on rough terrain with variable softness of the ground. The payload and rough terrain tests were repeated twice with consistent results. The same A-RMA policy was deployed in all these scenarios without any real world fine-tuning, task specific tuning, or calibration. We observe that A-RMA maintains stability in all of these deployment scenarios despite never having seen some of them during training.

there are only 39/147 feasible commands using the HZD controller. However, all of the RL-based controllers can cover the entire range of the testing commands, *i.e.*, there are 147/147 feasible commands using locomotion controllers using Robust MLP, RMA, and A-RMA. In terms of the tracking errors, as shown in Fig. 3, A-RMA manifests the least tracking errors with the only $[0.0849, 0.0952]^T$ m/s compared to other RL-based controllers (Table. II and Table. III). Moreover, as seen in Fig. 3, RMA and A-RMA produce less oscillations than Robust MLP. A-RMA has an overall superior performance in command tracking.

c) Mean Jerk: We measure the jerk experienced in the motors in MuJoCo for the baseline controllers. We observe in Table III that A-RMA almost matches the smoothness of the A-RMA-priv, while RMA and A-RMA-static seem to do much worse. The additional phase 3 in A-RMA allows the base policy to learn smooth behaviours while accounting for the imperfect extrinsics estimator.

d) Slippery Ground: We use MATLAB Simulink to quantify the ability of locomotion controller to maintain gait stability on slippery ground. Fig. 4 shows that both HZD and Robust MLP fail to maintain robot balance when the ground friction coefficient is set to 0.2 while RMA and A-RMA succeed to control the robot to walk forwards and backwards. We do a line search on friction values and observe that the minimal ground friction for which HZD and Robust MLP can maintain stability is 0.3 while RMA and A-RMA can go down to 0.2. This is significant because bipedal robots

are inherently unstable and become very hard to control as the coefficient of friction approaches zero. Note that the RL-controllers were not trained for friction value of 0.2 (see Table I), demonstrating that RMA and A-RMA have better generalization at test time than Robust MLP. Furthermore, as shown in Fig. 4d, Cassie does not deviate a lot in the lateral direction (with command $\dot{q}_y^d = 0$) when it walks backwards using A-RMA, while there exists a large drift to the right for RMA in Fig. 4c. This showcases A-RMA’s superior tracking performance even on a slippery ground.

B. Real World Experiments

We deploy A-RMA on a Cassie bipedal robot in the real world, and test in four scenarios: 1) tracking variable commands on a nominal ground, 2) robot walking while towing a heavy load with cable, 3) slippery ground, and 4) rough terrain with variable softness. We use the same policy for all our experiments which was trained in MuJoCo simulation and is deployed without any finetuning or calibration. The results are demonstrated in Fig. 5. Videos are available at [48] and included in the video attachment.

As shown in Fig. 5a, 5b, Cassie is able to track varying commands including forwards, backwards, and sideways walking speed while maintaining gait stability and low ground impacts.

In the load carrying task, Cassie controlled by A-RMA is able to tow the load (around 40 kg with wheels) while maintaining forward walking speed with no significant drift to other directions, as shown in Fig. 5c. During the tests, the

robot is able to stay robust to the pulling force from the cable which fluctuates depending on the speed of the payload and the tension in the string making it hard to model. Despite, slack/taut changes happening during the trials, the robot is able to maintain stability against such hybrid mode switches.

During the slippery ground test, we cover the ground by a plastic sheet with water in between in order to reduce the ground friction coefficient, as shown in Fig. 5d. When the robot steps onto the plastic sheet, there are significant slips between the robot feet and ground, *i.e.*, unexpected contact changes which makes controlling a life-sized robot with such low ground traction very challenging. However, the proposed policy is able to adapt to the changes in ground friction and contacts, and therefore able to maintain gait stability on such a slippery ground while tracking varying commands in both forward and lateral directions.

We also introduce challenging types of terrains and contacts in the tests presented in Fig. 5e, where we randomly place soft foams and wooden planks on the ground and let the robot walk onto that region. Such random changes of softness on the ground will change the contact type between the robot feet and ground: the contact region is on the sole when robot steps on a rigid plank while the contact can happen anywhere on the robot foot if it step into the soft foam, which making the contact very hard to model. Cassie, using A-RMA, is not only able to step on the rigid plank but also onto the soft foams without losing balance. Please note that the softness of the ground is not randomized during training as simulating a soft contact with high fidelity is still an open question. However, A-RMA is still able to generalize to the softness changes in the real world despite never having seen it during training.

VII. CONCLUSION

We study bipedal walking in complex, high-dimensional, life-sized Cassie robot. We propose A-RMA which extends the RMA algorithm by accounting for imperfect extrinsics estimation from the adaptation module. This is done by adding an additional base policy finetuning phase using model-free RL. This phase uses the imperfect extrinsics from adaptation module to allow the base policy to account for the unobservability in extrinsics. We have shown empirical gains in performance over model based as well as one of state-of-the-art RL controllers. We train A-RMA on bipedal Cassie robot in simulation and then use the same A-RMA controller to experiment with walking on foam, slippery surface, rough terrain and payload towing in the real world. A-RMA shows generalization to terrains beyond what is seen during training without additional real-world finetuning or calibration. One limitation of the current work is that the robot is blind and only uses proprioception. For future work, it would be interesting to integrate on-board visual cameras for walking.

ACKNOWLEDGEMENT

This work was supported by the DARPA Machine Common Sense program and National Science Foundation Grant CMMI-1944722.

REFERENCES

- [1] M. Vukobratovic and B. Borovac, “Zero-moment point—thirty five years of its life,” *International journal of humanoid robotics*, vol. 1, no. 01, pp. 157–173, 2004. 1, 2
- [2] S. Kuindersma, F. Permenter, and R. Tedrake, “An efficiently solvable quadratic program for stabilizing dynamic locomotion,” in *IEEE International Conference on Robotics and Automation*, 2014, pp. 2589–2594. 1, 2
- [3] J. Pratt, T. Koolen, T. De Boer, J. Rebuta, S. Cotton, J. Carff, M. Johnson, and P. Neuhaus, “Capturability-based analysis and control of legged locomotion, part 2: Application to m2v2, a lower-body humanoid,” *The international journal of robotics research*, vol. 31, no. 10, pp. 1117–1133, 2012. 1, 2
- [4] X. Xiong and A. D. Ames, “Coupling reduced order models via feedback control for 3d underactuated bipedal robotic walking,” in *IEEE-RAS International Conference on Humanoid Robots*, 2018. 1, 2
- [5] S. Feng, X. Xinjilefu, W. Huang, and C. G. Atkeson, “3d walking based on online optimization,” in *2013 13th IEEE-RAS International Conference on Humanoid Robots*, 2013, pp. 21–27. 1, 2
- [6] H. Dai, A. Valenzuela, and R. Tedrake, “Whole-body motion planning with centroidal dynamics and full kinematics,” in *IEEE-RAS International Conference on Humanoid Robots*, 2014, pp. 295–302. 1, 2
- [7] J. W. Grizzle, C. Chevallereau, A. Ames, and R. Sinnet, “3d bipedal robotic walking: Models, feedback control, and open problems,” in *IFAC Symposium on Nonlinear Control Systems*, 2010. 1, 2
- [8] N. A. Radford, P. Strawser, K. Hambuchen, J. S. Mehling, W. K. Verdeyen, A. S. Donnan, J. Holley, J. Sanchez, V. Nguyen, L. Bridgewater *et al.*, “Valkyrie: Nasa’s first bipedal humanoid robot,” *Journal of Field Robotics*, vol. 32, no. 3, pp. 397–419, 2015. 1
- [9] X. Da, O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle, “From 2d design of underactuated bipedal gaits to 3d implementation: Walking with speed tracking,” *IEEE Access*, vol. 4, pp. 3469–3478, 2016. 1, 2
- [10] Y. Gong and J. Grizzle, “Angular momentum about the contact point for control of bipedal locomotion: Validation in a lip-based controller,” *arXiv preprint arXiv:2008.10763*, 2020. 1, 2
- [11] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, “Feedback control of a cassie bipedal robot: Walking, standing, and riding a segway,” in *American Control Conference*, 2019, pp. 4559–4566. 1, 2, 5
- [12] Z. Li, C. Cummings, and K. Sreenath, “Animated cassie: A dynamic relatable robotic character,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. 1, 2, 4, 5
- [13] J. Reher and A. D. Ames, “Control lyapunov functions for compliant hybrid zero dynamic walking,” *arXiv preprint arXiv:2107.04241*, 2021. 1, 2
- [14] E. Dantec, R. Budhiraja, A. Roig, T. Lembono, G. Saurel, O. Stasse, P. Fernbach, S. Tonneau, S. Vijayakumar, S. Calinon *et al.*, “Whole body model predictive control with a memory of motion: Experiments on a torque-controlled talos,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8202–8208. 1
- [15] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning quadrupedal locomotion over challenging terrain,” *Science Robotics*, vol. 5, no. 47, 2020. 1, 2, 3
- [16] A. Kumar, Z. Fu, D. Pathak, and J. Malik, “RMA: Rapid Motor Adaptation for Legged Robots,” in *Robotics: Science and Systems*, 2021. 1, 2, 3, 5
- [17] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, “Reinforcement learning for robust parameterized locomotion control of bipedal robots,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 2811–2817. 1, 2, 3, 4, 5
- [18] J. Siekmann, S. Valluri, J. Dao, L. Bermillo, H. Duan, A. Fern, and J. Hurst, “Learning memory-based control for human-scale bipedal locomotion,” in *Robotics: Science and Systems*. 2020. 1, 2
- [19] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, “Domain randomization for transferring deep neural networks from simulation to the real world,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 23–30. 1, 2
- [20] Z. Li, J. Zeng, S. Chen, and K. Sreenath, “Vision-aided autonomous navigation of underactuated bipedal robots in height-constrained environments,” *arXiv preprint arXiv:2109.05714*, 2021. 2
- [21] N. Kohl and P. Stone, “Policy gradient reinforcement learning for fast quadrupedal locomotion,” in *IEEE International Conference on Robotics and Automation*, 2004., vol. 3. IEEE, 2004, pp. 2619–2624. 2

- [22] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019. 2
- [23] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Robotics: Science and Systems*, 07 2020. 2
- [24] A. Iscen, K. Caluwaerts, J. Tan, T. Zhang, E. Coumans, V. Sindhwani, and V. Vanhoucke, "Policies modulating trajectory generators," in *Conference on Robot Learning*. PMLR, 2018, pp. 916–926. 2
- [25] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning," in *Robotics: Science and Systems*, 2018. 2
- [26] G. A. Castillo, B. Weng, T. C. Stewart, W. Zhang, and A. Hereid, "Velocity regulation of 3d bipedal walking robots with uncertain dynamics through adaptive neural network controller," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7703–7709. 2
- [27] G. A. Castillo, B. Weng, W. Zhang, and A. Hereid, "Hybrid zero dynamics inspired feedback control policy design for 3d bipedal locomotion using reinforcement learning," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 8746–8752. 2
- [28] Z. Xie, G. Berseth, P. Clary, J. Hurst, and M. van de Panne, "Feedback control for cassie with deep reinforcement learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 1241–1246. 2
- [29] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. Panne, "Learning locomotion skills for cassie: Iterative design and sim-to-real," in *Conference on Robot Learning*, 2020, pp. 317–329. 2
- [30] 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 *International Conference on Robotics and Automation*, 2019, pp. 6023–6029. 2
- [31] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," in *Robotics: Science and Systems*, 2021. 2
- [32] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7309–7315. 2
- [33] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," *Conference on Robot Learning*, 2021. 2
- [34] L. Yang, Z. Li, J. Zeng, and K. Sreenath, "Bayesian optimization meets hybrid zero dynamics: Safe parameter learning for bipedal locomotion control," *arXiv preprint arXiv:2203.02570*, 2022. 2
- [35] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," in *Robotics: Science and Systems*, 2013. 2
- [36] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *IEEE international conference on robotics and automation*, 2018, pp. 1–8. 2
- [37] W. Yu, V. C. Kumar, G. Turk, and C. K. Liu, "Sim-to-real transfer for biped locomotion," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 3503–3510. 2
- [38] W. Yu, J. Tan, Y. Bai, E. Coumans, and S. Ha, "Learning fast adaptation with meta strategy optimization," *IEEE Robotics and Automation Letters*, 2020. 2
- [39] W. Yu, J. Tan, C. K. Liu, and G. Turk, "Preparing for the unknown: Learning a universal policy with online system identification," in *Robotics: Science and Systems*, 2017. 2
- [40] W. Yu, C. K. Liu, and G. Turk, "Policy transfer with strategy optimization," in *International Conference on Learning Representations*, 2018. 2
- [41] W. Zhou, L. Pinto, and A. Gupta, "Environment probing interaction policies," in *7th International Conference on Learning Representations, ICLR 2019*, 2019. 2
- [42] W. Yu, V. C. V. Kumar, G. Turk, and C. K. Liu, "Sim-to-real transfer for biped locomotion," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2019. 2
- [43] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017. 2
- [44] X. Song, Y. Yang, K. Choromanski, K. Caluwaerts, W. Gao, C. Finn, and J. Tan, "Rapidly adaptable legged robots via evolutionary meta-learning," in *International Conference on Intelligent Robots and Systems (IROS)*, 2020. 2
- [45] I. Clavera, A. Nagabandi, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn, "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," in *International Conference on Learning Representations*, 2019. 2
- [46] 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. 4
- [47] Agility Robotics, "cassie-mujoco-sim. (2018)." [Online]. Available: <https://github.com/osudr/cassie-mujoco-sim> 4
- [48] "Experiment video. (2022)." [Online]. Available: <https://youtu.be/HSdFX0qQqg> 6