RINA: Rapid Introspective Neural Adaptation for Out-of-Distribution Payload Configurations on Quadruped Robots

Oscar Youngquist and Hao Zhang

***Abstract*— Adaptive locomotion is a fundamental capability for quadruped robots, particularly in real-world scenarios when they must transport novel or out-of-distribution (O.O.D.) payloads across diverse terrains. Previous learning-based methods often tightly couple a locomotion controller's learned parameters with the adaptation process, which requires extensive pre-training or slow online updates when encountering O.O.D. payloads. To enable adaptation of quadruped locomotion to O.O.D. payloads, we propose the novel Rapid Introspective Neural Adaptation (RINA) method that rapidly compensates for differences between expected and actual joint torques caused by O.O.D. payloads. RINA introduces an adaptive residual dynamics representation that decouples the learning model's parameters from those used for adaptation. A new neural operator network is introduced to learn a set of basis functions as the learning model, which are combined using linear coefficients to predict residual dynamics. Then, these residual dynamics are used to adjust the locomotion controller's output, compensating for additional torques induced by the O.O.D. payload. During execution, the mixing coefficients can be rapidly and introspectively adapted on-the-go to generate joint torque compensations for O.O.D. payloads, while keeping the learned basis functions unchanged. Experimental results have demonstrated that our RINA approach well addresses on-the-go O.O.D. payload adaptation on varied natural terrains without collecting and retraining on additional data and outperforms baseline methods.**

More details of this work are provided on the project website: `https://hcrlab.gitlab.io/project/rina`.

I. Introduction

Quadruped robots have seen a dramatic growth in capabilities [1]–[8]. This rapid development has been motivated by the performance potential of legged systems in unstructured environments. Prominent applications include autonomous inspection [9]–[12], disaster response [13], [14], and search and rescue [15], [16]. In these scenarios, legged robots may be tasked with navigating various terrains while transporting payloads previously unseen during design or training, such as in the disaster response scenario in Fig. 1. Due to the additional torques induced on the quadruped's torso, the robot must rapidly adapt its locomotion controller in real time to satisfy the needs of the out-of-distribution (O.O.D.) payload configurations and terrain variations.

Due to the importance of payload adaptive legged locomotion, various approaches have been investigated. Non-learning approaches leverage control theory and knowledge of the physical dynamics of the system [17]–[20]. However, these approaches typically respond slowly to payload changes and require extensive engineering effort. Recently, approaches that aim to learn adaptive legged locomotion controllers have gained increasing interest [5], [6], [21]–[29]. However, current state-of-the-art methods suffer from tightly coupling the underlying learned controller with the adaptation process. This requires the collection of new data on-the-go to update the learning model's parameters or extensive pretraining.

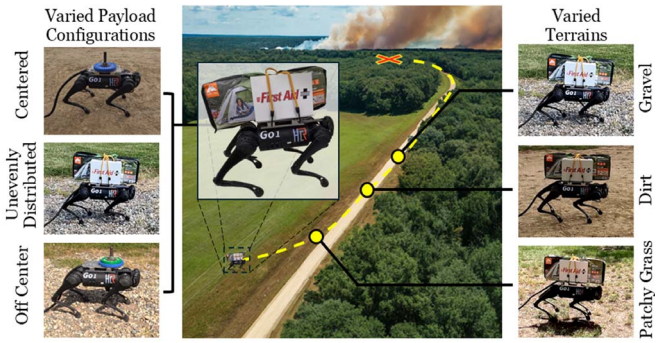


Fig. 1. A motivating scenario for payload adaptive quadruped locomotion. In a disaster response scenario payload adaptive locomotion enables a quadruped robot be to transport a variety unfamiliar payload configurations across varied natural terrains unseen during training.

To address the problem of adapting quadruped locomotion controllers to O.O.D. payload configurations, we propose a novel *Rapid Introspective Neural Adaptation* (RINA) method. RINA enables quadruped robots to rapidly compensate for differences between actual and expected joint torques induced by payloads unaccounted for by a parameterized locomotion controller. We design an adaptive residual dynamics representation that decouples the trained neural network parameters from the parameters employed for adaptation. A new neural operator network learns a set of basis functions that are mixed using linear coefficients to predict the residual dynamics. The predicted residual dynamics are then used to compensate the output of a locomotion controller for the torques induced by a O.O.D. payload. During execution, coefficients can be rapidly and introspectively adapted on-the-go in order to generate joint torque compensation values for the novel payloads while leaving the learned network parameters fixed. Introspective adaptation refers to the ability for a robot to monitor and assess its actual performance compared to its expected performance and update it's control output to minimize observed differences [30], [31]. Our approach enables quadruped robots to rapidly adapt to O.O.D. payload configurations without extensive pretraining or needing to fine tune the existing model with a new dataset on-the-go.

*This work was partially supported by NSF CAREER Award IIS-2308492, DARPA Young Faculty Award (YFA) D21AP10114-00, and NSF award IIS-2404386.

Oscar Youngquist and Hao Zhang are with Human-Centered Robotics Lab, University of Massachusetts Amherst, Amherst, MA 01002, USA. Email: {oyoungquist, hao.zhang}@umass.edu

The main contribution of this work is the introduction of our novel learning-based RINA method to address O.O.D. payload adaptation. Two specific novelties include:

- We enable the ability to rapidly adapt legged locomotion controllers to payloads unseen during design or training through introspectively updating joint torque command values, which allows quadruped robots to observe and adapt their joint torque output to compensate controllers for O.O.D. payloads on-the-go.
- We introduce a novel neural operator network to learn a decoupled representation of the joint torque residual dynamics for adaptation, eliminating the need to update the learned network parameters on-the-go.

II. Related Work

Robot adaptation to payloads is a critical problem that may involve various robot morphologys and payload characteristics [19], [30], [32]–[37]. Approaches to this problem can be broadly split into non-learning and learning methods.

A. Non-Learning Methods for Payload Adaptation

Non-learning approaches to payload adaptive quadruped locomotion typically model the physical dynamics of the robotic system and exploit the tools of control theory, e.g., based on online payload identification [17] and force-based quadratic programming [18]. Techniques based on Control Lyapunov Functions (CLFs) [19] guarantee system stability while achieving control objectives by representing a system's energy and ensuring it decreases over time towards a desired state. Recently, the method proposed in [20] integrates L1 adaptive control techniques [38], [39], consisting of a fast adaptation law and a low-pass filter to decouple estimation and control, into a force-based model predictive control (MPC) method. However, these approaches typically suffer from increasingly complex control architectures with high computational costs and require extensive expertise during development.

B. Learning Approaches for Payload Adaptation

The learning methods for adaptable quadruped locomotion can be coarsely sorted into two categories. The first involves Reinforcement Learning (RL) training with privileged state-information [6], [21]–[24], [40]. The adaptive ability of these approaches comes from large amounts of training data across a wide range of conditions. However their performance is limited to conditions encountered during training [41]. The second category updates RL controllers with new data for O.O.D. conditions in-situ (i.e. learning on-the-go) [25]–[29]. However, this process is too slow to adapt rapidly to sudden changes, like payload-induced torques.

Another class of methods combines learning with classical adaptive control. These methods generate compensation values to adjust control inputs or outputs to handle disturbances. Neural-Fly [42] implements a learned set of basis function and an adaptive control algorithm to compensate UAV thrust in windy conditions. RL is applied to generate offset values in an MPC method to create adaptive behaviors for quadruped robots [35]. However, this method's adaptive payload capabilities are relatively limited to the configurations it was exposed to during training. Moreover, these learning approaches do not directly consider the governing differential equations that mathematically model the quadruped robot's dynamical system.

C. Neural Operator Networks

Deep Operator Networks (DeepONets) are a neural network framework designed to learn mathematical operators that map between infinite-dimensional function spaces. This allows them to efficiently approximate complex relationships in dynamical systems governed by differential equations (DEs) [43]–[45]. Once trained, DeepONets can generalize well across different inputs and conditions, allowing them to adapt to a variety of problem settings without the need for retraining. DeepONets have been applied to modeling multiple DEs representing various dynamical systems [46]–[52]. However, the insights on neural operator networks have yet to be investigated and applied to robot control problems.

III. Approach

Notation: Matrices are denoted as boldface uppercase letters (e.g. $\mathbf{M} = \{M_{i,j}\} \in \mathbb{R}^{n \times m}$ is an $n \times m$ matrix whose element in the i-th row and j-th column is $M_{i,j}$), vectors are denoted as boldface lowercase letters (e.g., $\mathbf{v} \in \mathbb{R}^d$ is a d-dimensional vector whose i-th element is v_i), and scalars are represented by non-boldface characters (e.g., s, α).

A. Problem Formulation of Payload Compensation

As a quadruped robot navigates in an environment while carrying a payload, it collects proprioceptive observations from its onboard sensors, including joint encoders capturing the leg joint positions $\mathbf{q}$, velocities $\dot{\mathbf{q}}$, and accelerations $\ddot{\mathbf{q}}$, as well as the 3D orientation Θ, angular velocity $\dot{\Theta}$, and linear acceleration $\ddot{\mathbf{p}}$ of the torso from an IMU. Additionally, a state estimator based on Kalman filters is used to provide global torso position $\mathbf{p}$ and velocity $\dot{\mathbf{p}}$ based on the kinematics and $\ddot{\mathbf{p}}$. Then, the proprioceptive data obtained at each time step is concatenated into a vector $\mathbf{s} = [\dot{\mathbf{p}}, \Theta, \dot{\Theta}, \mathbf{q}, \dot{\mathbf{q}}]$ as the input to the controller.

We define the expected locomotion behavior as the torso's expected linear and angular velocities $\mathbf{b}^{ex} = [\dot{\mathbf{p}}^{ex}, \dot{\Theta}^{ex}]^\top$ (e.g., provided by a planner or through user input). Then a parameterized locomotion controller $\Phi(\cdot)$ uses $\mathbf{s}$ and $\mathbf{b}^{ex}$ to generate expected torque commands $\mathbf{a}^{ex}$ for the controlled leg joints by $\Phi : (\mathbf{s}, \mathbf{b}^{ex}) \mapsto \mathbf{a}^{ex}$. During execution, we can calculate the actual joint torques $\mathbf{a}^{act}$ by plugging sensor measurements into the known robot dynamics. However, when the robot carries a payload, $\mathbf{a}^{act}$ will deviate from the expected command, i.e., $\mathbf{a}^{act} - \mathbf{a}^{ex} \neq \mathbf{0}$. In order to model such joint torque differences, we represent $\mathbf{a}^{act}$ as a combination of the expected torques and the torques induced by the unmodeled payload: $\mathbf{a}^{act} = \mathbf{a}^{ex} + p(\mathbf{s}, c)$. The function $p : (\mathbf{s}, c) \mapsto \mathbf{a}^p$ maps the robot state $\mathbf{s}$ and hidden state c, representing the payload configuration, to the payload induced joint torques $\mathbf{a}^p$. Without knowing $p(\mathbf{s}, c)$ or c,

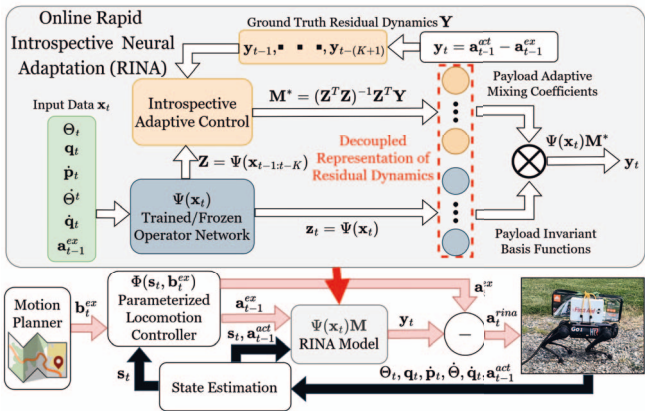


Fig. 2. Overview of RINA for on-the-go O.O.D. payload adaptation. Using a new decoupled representation based on neural operator basis functions, RINA rapidly compensates for the joint torque residual dynamics induced by O.O.D. payload configurations through efficiently updating mixing coefficients for the basis functions.

we can estimate $p(\cdot)$ through $p(\mathbf{s}, c) \approx \mathbf{a}^{act} - \mathbf{a}^{ex}$, which models $p(\mathbf{s}, c)$ as the residual dynamics between the actual and expected joint torques. Treating $p(\mathbf{s}, c)$ as joint torque offsets to compensate for the impact of the payload on the robot dynamics, we can generate the payload adaptive torque commands by $\Phi(\mathbf{s}, \mathbf{b}^{ex}) - p(\mathbf{s}, c) = \mathbf{a}^{adp}$.

However, during execution the robot does not have direct access to $\mathbf{a}^{act}$ because the $\mathbf{a}^{ex}$ produced by $\Phi(\mathbf{s}, \mathbf{b}^{ex})$ have not been applied to the robot's current state $\mathbf{s}$. As such, the robot must predict the residual dynamics $p(\mathbf{s}, c)$ based on $\mathbf{s}$. To address this critical challenge, we learn an adaptable residual dynamics representation, $\Omega : (\mathbf{s}_t, \mathbf{a}^{ex}_{t-1}) \mapsto p(\mathbf{s}_t, c)$, which uses the current state $\mathbf{s}_t$ and the previous torque command $\mathbf{a}^{ex}_{t-1}$ to estimate the residual dynamics $p(\mathbf{s}_t, c)$.

Letting $\mathbf{x}_t = [\mathbf{s}_t, \mathbf{a}^{ex}_{t-1}]$, and denoting the ground truth residual dynamics as $\mathbf{y}_t = p(\mathbf{s}_t, c)$, the problem of learning $\Omega(\mathbf{x}_t)$ to provide torque offsets can be formulated as:

$$\underset{\Omega}{\operatorname{argmin}} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathbf{D}} \|\mathbf{y}_i - \Omega(\mathbf{x}_i)\|^2 \tag{1}$$

where $\mathbf{D} = \{\mathbf{D}_1, \ldots, \mathbf{D}_C\}$ denotes a dataset collected from C payload conditions, and $\mathbf{D}_c = [(\mathbf{x}_0, \mathbf{y}_0), \ldots, (\mathbf{x}_{N_c}, \mathbf{y}_{N_c})]$ includes N_c data samples collected in the c-th condition. Eq. (1) aims to learn an adaptable representation $\Omega(\cdot)$ through minimizing the residual dynamics prediction error of $\Omega(\cdot)$ over $\mathbf{D}$. Once $\Omega(\cdot)$ is trained, it is used to generate payload adaptive joint torques: $\Phi(\mathbf{s}_t, \mathbf{b}^{ex}_t) - \Omega(\mathbf{x}_t) = \mathbf{a}^{adp}_t$. This problem formulation enables legged robots to rapidly adapt to familiar payloads by compensating for differences between $\mathbf{a}^{ex}$ and $\mathbf{a}^{act}$ resulting from payloads.

The learning problem in Eq. (1) is formulated as a function regression, which uses samples from some input distribution to train $\Omega(\cdot)$ to predict a continuous variable for some output distribution. However, this formulation has three limitations. First, the evolution of complex dynamical systems (e.g., quadruped robots) are typically mathematically modeled by mathematical operators composed of ordinary differential equations (ODEs), which cannot be adequately addressed by function regression [43], [46]. Second, training $\Omega(\cdot)$ requires a substantial amount of data that covers a wide range of payload conditions [6], [22], [23]. Otherwise, the model may overfit to the training data, limiting its ability to generalize to O.O.D. system states and payload conditions [46], [48], [51], [52]. Third, to directly adapt $\Omega(\cdot)$ on-the-go, the robot must collect data to update $\Omega(\cdot)$'s parameters during execution [26], [27], [29]. Collecting new data on-the-go is usually infeasible or dangerous in the field, and the online update of $\Omega(\cdot)$ may not converge on time for rapid adaptation.

B. RINA for Learning O.O.D. Payload Adaptation

We propose a novel RINA method for rapid O.O.D. payload adaptation while addressing the aforementioned limitations. An overview of RINA is illustrated in Fig. 2. To enable on-the-go adaptation without needing to collect new data for online training, we propose an adaptive decoupled operator representation that is computed as a linear combination of a learned set of basis functions and mixing coefficients. A set of basis functions $\mathbf{z}$ are defined as a linearly independent subset of a function space that can be linearly combined to represent every function within the space [53], [54] (i.e. $\mathbf{z}$ spans the function space). Different from the direct prediction in Eq. (1), we define $\Psi : \mathbf{x} \mapsto \mathbf{z}$ to learn basis functions $\mathbf{z}$ spanning all payload configurations.

We introduce mixing coefficients $\mathbf{M}$ for individual payload conditions to linearly combine the set of basis functions $\Psi(\cdot)$. Combining the payload invariant $\Psi(\mathbf{x})$ with the payload specific $\mathbf{M}$ results in the $\mathbf{y}$ representation $\Psi(\mathbf{x})\mathbf{M} = \mathbf{y}$. In this representation, $\mathbf{M}$ projects the function space $\Psi(\cdot)$, summarizing the robot's current payload agnostic dynamics, to the function space of residual joint torques caused by a payload $p(\mathbf{s}, c)$. Accordingly, $\mathbf{M}$ can be interpreted mathematically as an operator. For complex dynamical systems like quadruped robots, the governing equations of motion take the form of ODEs, defining a mathematical operator describing the evolution of the system through time [39], [43]–[46]. Through leveraging the operator $\mathbf{M}$ in the residual dynamics model, RINA aligns it's mathematical representation with that of the dynamics governing the system. Moreover, learning an operator representation via basis functions enables generalization to novel payload conditions through learning a set of functions that span the output space. This allows for the estimation of $\mathbf{y}$ for novel payloads by mixing $\Psi(\cdot)$. Therefore, RINA separates learning a representation of the function space $\Psi(\cdot)$ from adapting it to produce specific functions through updating the operator $\mathbf{M}$. This separation enables $\Psi(\mathbf{x})\mathbf{M}$ to adapt to novel payload configurations by updating $\mathbf{M}$, without needing to retrain $\Psi(\cdot)$ with new data.

To predict $\mathbf{y}$ using $\Psi(\mathbf{x})\mathbf{M}$, $\mathbf{M}$ must be updated for any encountered payload configuration. RINA updates $\mathbf{M}$ through assessing the performance of $\Psi(\mathbf{x})\mathbf{M}$ at predicting the ground truth $\mathbf{y}$ that results from a particular payload. Then, the update of $\mathbf{M}$ is mathematically computed by:

$$\mathbf{M}^* = \underset{\mathbf{M}}{\operatorname{argmin}} \sum_{(\mathbf{x}_k, \mathbf{y}_k) \in \mathbf{B}^A} \|\mathbf{y}_k - \Psi(\mathbf{x}_k)\mathbf{M}\|^2 \tag{2}$$

where $\mathbf{B}^A$ is a dataset of size K used to adapt $\mathbf{M}$ to individual

Algorithm 1: RINA Training

Input: $\mathbf{D}$
Output: Trained models $\Psi(\cdot), \zeta(\cdot)$

1 Initialize $\Psi(\cdot), \zeta(\cdot)$
2 **while** ***not converged*** **do**
3 Sample $\mathbf{B}^A$ and $\mathbf{B}^T$ from $\mathbf{D}$
4 $\mathbf{M}^* \leftarrow \text{argmin}_{\mathbf{M}} \sum_{(\mathbf{x}_k, \mathbf{y}_k) \in \mathbf{B}^A} \|\mathbf{y}_k - \Psi(\mathbf{x}_k)\mathbf{M}\|^2$
5 **if** $\|\mathbf{M}^*\| > \gamma$ **then** $\mathbf{M}^* \leftarrow \gamma \frac{\mathbf{M}^*}{\|\mathbf{M}^*\|}$
6 Update $\Psi(\cdot)$ with: $\sum_{(\mathbf{x}_i, \mathbf{y}_i) \in B^T} \|\mathbf{y}_i - \Psi(\mathbf{x}_i)\mathbf{M}^*\|^2 - \lambda \mathcal{L}\left(\zeta(\Psi(\mathbf{x}_i))\right)$
7 **if** *rand*() $\leq \delta$ **then**
8 Update $\zeta(\cdot)$ with $\sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathbf{B}^T} \mathcal{L}\left(\zeta(\Psi(\mathbf{x}_i))\right)$
9 **return** $\Psi(\cdot), \zeta(\cdot)$

payload conditions. Eq. (2) introspectively updates $\mathbf{M}$ through observing and minimizing the $\mathbf{y}$ prediction error of $\Psi(\mathbf{x})\mathbf{M}$ on $\mathbf{B}^A$.

However, $\Psi(\cdot)$ may overfit to the payload specific distributions in $\mathbf{x}$ for each condition in $\mathbf{D}$. As a result, $\Psi(\cdot)$ becomes unique to each specific condition, while $\mathbf{M}$ remains fixed, limiting the model's ability to adapt to O.O.D. payload configurations without retraining. To prevent overfitting, we use a discriminator network $\zeta(\Psi(\mathbf{x}))$ trained with the cross-entropy loss $\mathcal{L}(\cdot)$ to predict the true payload condition from $\Psi(\mathbf{x})$. The loss $\mathcal{L}(\zeta(\Psi(\mathbf{x})))$ acts as regularization for $\Psi(\cdot)$, ensuring that it does encode payload specific information, forcing it to be represented in $\mathbf{M}$.

Putting all these components together results in our final loss function for RINA:

$$\max_{\zeta} \min_{\Psi, \mathbf{M}} \sum_{(\mathbf{x}_i, \mathbf{y}_i) \in \mathbf{D}} \|\mathbf{y}_i - \Psi(\mathbf{x}_i)\mathbf{M}\|^2 - \lambda \mathcal{L}\left(\zeta(\Psi(\mathbf{x}_i))\right) \quad (3)$$

where $\lambda > 0$ controls the degree of discriminator regularization. Eq. (3) learns a set of basis functions that adaptively represent the residual dynamics through having $\Psi(\cdot)$ and $\zeta(\cdot)$ compete in a zero-sum game during training: In the outer maximization, $\zeta(\cdot)$ attempts to maximize the performance of predicting the payload configuration from $\Psi(\mathbf{x})$. In the inner minimization, $\Psi(\cdot)$ seeks to learn a set of the basis functions that minimizes the error of predicting $\mathbf{y}$ while fooling the discriminator into making incorrect predictions.

We present our training procedure in Algorithm 1. Given a dataset collected from distinct payload conditions, each epoch randomly samples disjoint training $\mathbf{B}^T$ and adaptation $\mathbf{B}^A$ batches from a single payload condition $\mathbf{D}_c \in \mathbf{D}$ on Line 3. $\mathbf{B}^A$ is then used on a frozen $\Psi(\cdot)$ to update $\mathbf{M}^*$ based on Eq. (4) on Line 4. To avoid ambiguity in $\mathbf{M}^*$, it is normalized via $\mathbf{M}^* = \frac{\mathbf{M}^*}{\|\mathbf{M}^*\|} \cdot \gamma$ if $\|\mathbf{M}^*\| > \gamma$ for $\gamma > 0$ on Line 5. After approximating $\mathbf{M}^*$ the parameters of $\Psi(\cdot)$ are updated using Eq. (3) with $\mathbf{B}^T$ on Line 6. To improve training stability and generalization to O.O.D. conditions, the Lipschitz property of the $\Psi(\cdot)$ is regulated through layer-wise spectral normalization [42], [55], [56]. This is performed by calculating the L2 norm on the parameters of each layer in $\Psi(\cdot)$ and normalizing if the norm exceeds a threshold $\kappa > 0$. Finally, $\zeta(\cdot)$ is updated on Line 8.

Algorithm 2: RINA On-the-Go Execution

Input: $\Psi(\cdot), \mathbf{a}^{act}, \mathbf{s}, \mathbf{b}^{ex}$
Output: $\mathbf{a}^{rina}, \mathbf{a}^{ex}$

1 Initialize $t \leftarrow 0$, $\mathbf{Z} \leftarrow \emptyset$, $\mathbf{Y} \leftarrow \emptyset$, $\mathbf{M} \leftarrow \emptyset$
2 **while** ***running*** **do**
3 **if** $t > 0$ **then**
4 $\mathbf{Y} \leftarrow \mathbf{Y} \cap (\mathbf{a}^{act}_{t-1} - \mathbf{a}^{ex}_{t-1})$
5 $\mathbf{Z} \leftarrow \mathbf{Z} \cap \mathbf{z}_{t-1}$
6 **if** $t \bmod K = 0$ **then**
7 $\mathbf{M}^* \leftarrow (\mathbf{Z}^\top \mathbf{Z})^{-1} \mathbf{Z}^\top \mathbf{Y}$
8 **if** $\|\mathbf{M}^*\| > \gamma$ **then** $\mathbf{M}^* \leftarrow \gamma \frac{\mathbf{M}^*}{\|\mathbf{M}^*\|}$
9 $\mathbf{M} \leftarrow \beta \mathbf{M}^* + (1 - \beta)\mathbf{M}$
10 $\mathbf{Z} \leftarrow \emptyset$, $\mathbf{Y} \leftarrow \emptyset$
11 $\mathbf{a}^{ex}_t \leftarrow \Phi(\mathbf{s}_t, \mathbf{b}^{ex}_t)$
12 $\mathbf{z}_t \leftarrow \Psi(\mathbf{s}_t, \mathbf{a}^{ex}_{t-1})$
13 **if** $\mathbf{M} \neq \emptyset$ **then**
14 $\mathbf{a}^{rina}_t \leftarrow \mathbf{a}^{ex}_t - \mathbf{z}_t \mathbf{M}$
15 **return** $\mathbf{a}^{rina}_t$
16 **else**
17 **return** $\mathbf{a}^{ex}_t$
18 $t \leftarrow t + 1$, $\mathbf{z}_{t-1} \leftarrow \mathbf{z}_t$, $\mathbf{a}^{ex}_{t-1} \leftarrow \mathbf{a}^{ex}_t$

C. RINA for Rapid On-the-Go Adaptation

To ensure $\mathbf{M}$ can be rapidly updated to adapt to O.O.D. payload configurations, we define $\mathbf{Z} = [\Psi(\mathbf{x}_{t-1}), \ldots, \Psi(\mathbf{x}_{t-(K+1)})]^\top$ as a set of the predicted basis functions for K time steps, and $\mathbf{Y} = [\mathbf{y}_{t-1}, \ldots, \mathbf{y}_{t-(K+1)}]^\top$ as the set of K corresponding ground truth residual dynamics. Then, Eq. (2) can be efficiently solved through the closed form equation:

$$\mathbf{M}^* = (\mathbf{Z}^\top \mathbf{Z})^{-1} \mathbf{Z}^\top \mathbf{Y} \quad (4)$$

RINA then calculates payload adaptive joint torques using the decoupled operator formulation by:

$$\Phi(\mathbf{s}_t, \mathbf{b}^{ex}_t) - \Psi(\mathbf{x}_t)\mathbf{M}^* = \mathbf{a}^{rina}_t \quad (5)$$

The first term generates the expected torques $\mathbf{a}^{ex}$ using the locomotion controller $\Phi(\cdot)$. The second term produces the predicted $\mathbf{y}$ used as joint torque offset values to compensate for unknown or O.O.D. payloads. Unlike the function regression formulation in Eq. (1), the decoupled operator formulation in Eq. (5) enables the robot to monitor the performance of $\Psi(\cdot)\mathbf{M}$ at generating payload compensation values, and updates $\mathbf{M}$ using Eq. (4) to improve its ability to adapt.

Our on-the-go adaptation procedure is presented in Algorithm 2. At each time step, the robot state $\mathbf{s}_t$, computed actual joint torques from the previous time step $\mathbf{a}^{act}_{t-1}$, and expected torso locomotion behaviors $\mathbf{b}^{ex}_t$ are provided to the algorithm by the robots state estimator and locomotion planner. On Lines 3-5, the matrices of predicted basis functions $\mathbf{Z}$ and residual dynamics $\mathbf{Y}$ are updated. On Lines 6-8, using a short history of K time steps to consider the impact of momentum, $\mathbf{M}^*$ is introspectively adapted based on Eq. (4) and normalized to avoid ambiguity across payload conditions. To minimize jerkiness and oscillations in the adaptive response, we use a regularized update technique to encourage smooth $\mathbf{M}$ updates, as seen on Line 9, where β serves as a regularization weight

used to control the magnitude of the $\mathbf{M}$ update. On Line 10, $\mathbf{Z}$ and $\mathbf{Y}$ are reset. The expected joint torques $\mathbf{a}_t^{ex}$ and basis functions $\mathbf{z}_t$ are generated on Lines 11 and 12 respectively. After the first $\mathbf{M}$ update, $\mathbf{a}_t^{rina}$ is calculated and sent to joint motors on Lines 13-15. Otherwise, on Line 17, $\mathbf{a}^{ex}$ is sent to the motors. Lastly, we increment time and store $\mathbf{a}_t^{ex}$ and $\Psi(\mathbf{x}_t)$ for use in the next time step on Line 18.

RINA achieves the capability of rapid on-the-go adaptation for two reasons. First, RINA models the operator $\mathbf{M}$ that maps the basis functions summarizing the robot's payload agnostic dynamics $\Psi(\mathbf{x})$ to payload induced residual joint torques $p(\mathbf{s}, c)$. Through mixing $\Psi(\mathbf{x})$ with $\mathbf{M}$, RINA learns a decoupled $p(\mathbf{s}, c)$ operator representation $\Psi(\mathbf{x})\mathbf{M}$. This decoupled representation allows RINA to adapt to O.O.D. payload configurations by only updating $\mathbf{M}$ while leaving the trained parameters of $\Psi(\cdot)$ fixed. This bypasses the requirements for on-the-go data collection and online training. Second, RINA can rapidly update $\mathbf{M}$ through the closed form solution based on matrix manipulation in Eq. (4), which further improves efficiency for adaptation to O.O.D. payload configurations during execution.

IV. Experiments

A. Experimental Settings

We experimentally validate RINA through physical robot experiments using the Unitree Go1 quadruped robot [57]. $\Phi(\cdot)$ is implemented using the Whole Body Impulse Control (WBIC) method [4] due to it's high update frequency (0.5 kHz) and robust whole body formulation. $\Psi(\cdot)$ is implemented as an MLP with three hidden layers predicting a set of 16 basis functions and $\mathbf{M}$ is updated using a history of $K = 5$ time steps at a rate of 100 Hz.

RINA is trained using a small dataset gathered from 12 minutes of real-world robot operation, featuring three payloads: 0, 5, and 10 pounds (lbs), in addition to 3.31 lbs of mounting hardware. Payloads were kept within the Go1 platform's recommended capacity and positioned near the robot's center of mass (COM). Data collection occurred indoors on flat concrete terrain, with a human operator guiding the robot through motions such as rotations, forward, backward, and lateral movements. Training and execution were conducted on an 8-core i9 machine with 64 GB of RAM. Further details are provided in the supplementary material on the project website.

We compare RINA's adaptation to O.O.D. payload configurations with two baseline methods: (1) a non-adaptive method that doesn't adjust $\Phi(\cdot)$ (**WBIC**) [4], and (2) a feedforward MLP neural network with the same architecture as RINA but predicting joint torque residuals directly as in Eq. (1) (**MLP**). Both adaptive methods use the WBIC controller for $\Phi(\cdot)$. We evaluate performance using the Root Mean Squared Error (RMSE) between the expected $\mathbf{b}^{ex}$ and actual $\mathbf{b}^{act}$ locomotion behaviors, quantifying each method's ability to handle disturbances from O.O.D. payloads. We also report RINA and MLP's percentage improvement over WBIC.

TABLE I

Quantitative results of RINA and comparisons with baselines on indoor concrete terrains.

Payload	**Linear-Velo. Error** (m/s)			**Angular-Velo. Error** (rad/s)		
	WBIC	MLP	RINA	WBIC	MLP	RINA
0 C	0.091	0.108	**0.081**	0.398	0.418	**0.397**
5 C	0.108	0.133	**0.104**	0.416	0.512	**0.407**
10 C	0.124	0.135	**0.110**	0.464	0.495	**0.457**
5 F	0.125	0.132	**0.091**	0.466	0.472	**0.410**
7.5 F	0.131	0.126	**0.102**	0.463	0.467	**0.429**
5 L	0.117	0.130	**0.097**	0.427	0.473	**0.413**
7.5 L	0.117	0.133	**0.105**	0.424	0.470	**0.408**
5 FL	0.118	0.138	**0.108**	0.477	0.487	**0.431**
7.5 FL	0.133	0.125	**0.104**	0.480	0.450	**0.397**
Avg.	0.119	0.130	**0.101**	0.446	0.473	**0.416**
% Im.	-	-9.24%	**15.13%**	-	-6.05%	**6.73%**

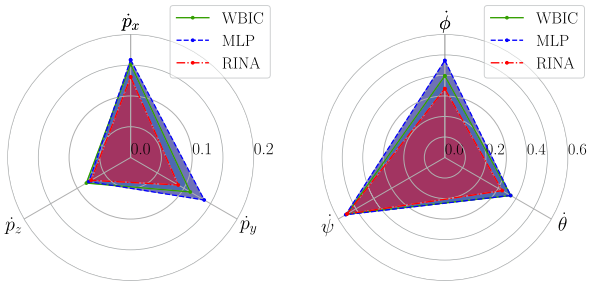
B. Results on Adaptation to O.O.D. Payload Configurations

To assess adaptability to O.O.D. payload configurations, we tested off-center payloads of 5 lbs and 7.5 lbs, positioned 3 inches in front of the COM (**F**), 1.5 inches to the left of the COM (**L**), and 3.5 inches in front and 1 inch to the left of the COM (**FL**). Off-center payloads of 10 lbs were excluded as all methods failed under this condition. Performance was also evaluated with centered payloads (**C**) from the training data. Tests were conducted on flat concrete by a human operator, averaging 31,646 time steps (63 seconds) per condition.

The indoor experiment results are quantitatively shown in Table I, with a summary of the overall results in Fig. 3. On average, RINA performs 18.72% better for the linear velocity tracking and 9.39% better for the angular velocity tracking compared to both baseline methods. In contrast, MLP reduces performance compared to the non-adaptive WBIC baseline, with decreases of 9.24% and 6.05% for linear and angular velocity tracking respectively. This is because the MLP lacks RINA's introspective adaptation ability, which adjusts output based on residual dynamics compensation performance.

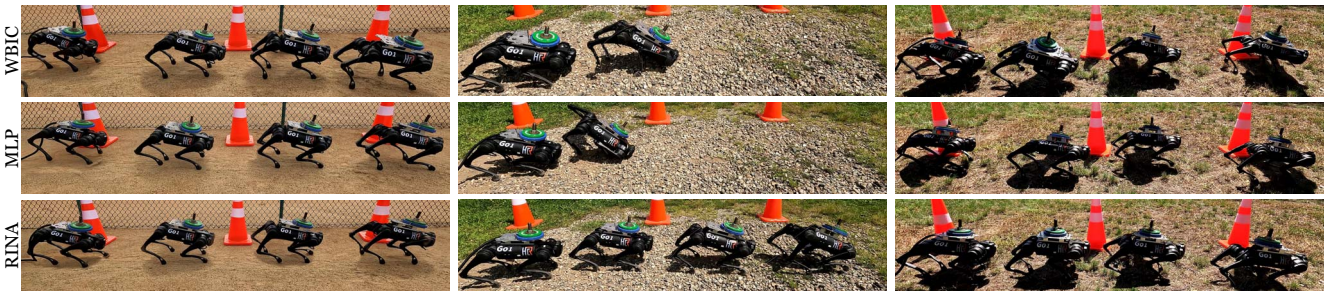
C. Results for O.O.D. Payload Adaptation On Varied Terrains

We evaluated RINA's performance on unseen terrains by comparing it to the baselines on three outdoor terrains: grass, dirt, and gravel. The same two off-center payloads from the indoor experiments were used, but with fewer mounting positions: centered, front, and left. The robot moved straight



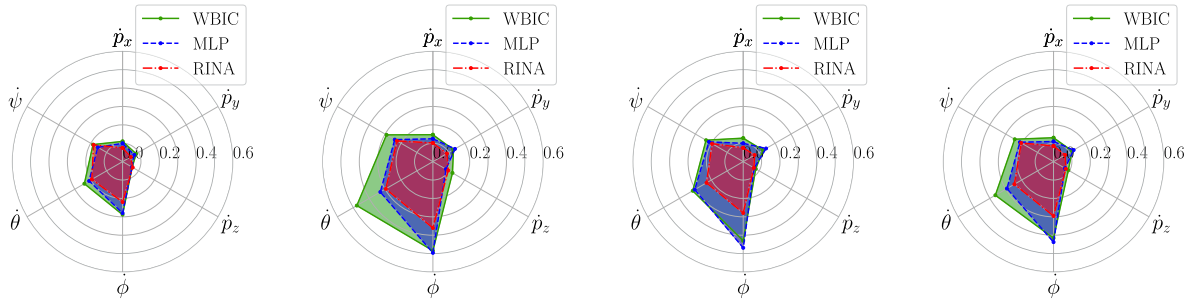
(a) RMSE on Linear Velocity (b) RMSE on Angular Velocity

Fig. 3. Radar plots depict the average RMSE of the robot's torso locomotion behaviors from indoor experiments, including (a) linear velocity error (x, y, z in the global frame) and (b) angular velocity error (roll, pitch, yaw).



(a) Dirt Terrain (b) Gravel Terrain (c) Grass Terrain

Fig. 4. Qualitative results on O.O.D. payload adaptation on unseen outdoor terrains.



(a) RMSE on Dirt Terrain (b) RMSE on Gravel Terrain (c) RMSE on Grass Terrain (d) RMSE on Tested Terrains

Fig. 5. Quantitative results based on radar plots to present the average RMSE of the torso locomotion behaviors across outdoor terrain variations.

TABLE II

QUANTITATIVE RESULTS AVERAGED ACROSS THREE OUTDOOR TERRAINS: DIRT, GRAVEL, AND GRASS. † INDICATES A METHOD EXPERIENCED A FAILURE DURING DATA COLLECTION AND ‡ INDICATES FAILURES PREVENTED DATA COLLECTION ON THE GRAVEL TERRAIN.

Payload	**Linear-Velo. Error** (m/s)			**Angular-Velo. Error** (rad/s)		
	WBIC	MLP	RINA	WBIC	MLP	RINA
5 C	0.106	0.095	**0.070**	0.272	0.283	**0.220**
7.5 C	0.122	0.106	**0.079**	0.403	0.363	**0.251**
5 F	0.109†	0.106†	**0.074**	0.347†	0.276†	**0.222**
7.5 F	0.107‡	0.099†	**0.076**	0.333‡	0.326†	**0.251**
5 L	0.106	0.099	**0.076**	0.334	0.326	**0.251**
7.5 L	0.112‡	0.118	**0.096**	0.316‡	0.381	**0.305**
Avg.	0.114‡	0.105†	**0.079**	0.349‡	0.327†	**0.251**
% Im.	-	7.89%	**30.70%**	-	6.30%	**28.08%**

across the terrains without backward and lateral movements, or rotations. A human operator collected evaluation data, averaging 8,777 time steps (17.5 seconds) per experiment.

Qualitative results for each method transporting O.O.D. payloads across outdoor terrains are shown in Fig. 4. The dirt terrain was most similar to the flat concrete used for training, and all approaches performed similarly with RINA slightly outperforming the baselines (Fig. 5). On the grass terrain, the robot's feet would get caught in grass clumps, causing the robot to lurch to the side leading to large roll velocity errors. RINA quickly compensated for these errors through introspective adaptation, while the baseline methods responded more slowly. The largest errors occurred on gravel, where both WBIC and MLP faced multiple failures and larger angular velocity errors. RINA successfully transported the payloads across the gravel terrain without failures.

The expected vs. actual torso tracking performance for the outdoor experiments is shown in Table II, with average results per terrain in Fig. 5. RINA outperformed both baselines, with an average improvement of 27.73% for linear velocity tracking and 25.66% for angular velocity tracking. The improved tracking performance of the adaptive methods in outdoor experiments compared to indoor tests is mainly due to the absence of lateral movements or rotations. These movements where found to cause significant roll, pitch, and lateral velocity tracking errors in the indoor experiments.

V. CONCLUSION

In this paper, we have proposed RINA as a novel learning-based residual dynamics compensation method for adapting quadruped locomotion controllers to O.O.D. payload configurations. RINA introduces an adaptive residual dynamics representation that separates the learning model's parameters from those used for adaptation. A neural operator network learning model is introduced to learn a set of basis functions, which are then mixed using linear coefficients to predict the residual dynamics. During execution, the mixing coefficients are introspectively adapted on-the-go in order to rapidly produce joint torque compensations for O.O.D. payloads, while keeping the learned basis functions fixed. Our experiments demonstrate that RINA enables rapid, on-the-go adaptation to O.O.D. payloads across multiple terrains and outperforms baseline methods. Future work will focus on incorporating the information of both joint positions and velocities in order to enhance RINA's ability to adapt to payloads on 3D terrains.

ACKNOWLEDGMENTS

The aerial terrain overview centered in Fig. 1 was AI generated using Google Gemini.

REFERENCES

[1] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[2] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the mit cheetah 3 through convex model-predictive control," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[3] C. D. Bellicoso, F. Jenelten, C. Gehring, and M. Hutter, "Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2261–2268, 2018.

[4] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.

[5] 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, p. eaau5872, 2019.

[6] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," *Robotics: Science and Systems*, 2021.

[7] M. Seo, R. Gupta, Y. Zhu, A. Skoutnev, L. Sentis, and Y. Zhu, "Learning to walk by steering: Perceptive quadrupedal locomotion in dynamic environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[8] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "Anymal parkour: Learning agile navigation for quadrupedal robots," *Science Robotics*, vol. 9, no. 88, p. eadi7566, 2024.

[9] C. Gehring, P. Fankhauser, L. Isler, R. Diethelm, S. Bachmann, M. Potz, L. Gerstenberg, and M. Hutter, "Anymal in the field: Solving industrial inspection of an offshore hvdc platform with a quadrupedal robot," in *Field and Service Robotics*. Springer, 2021, p. 247.

[10] S. Halder, K. Afsari, J. Serdakowski, S. DeVito, M. Ensafi, and W. Thabet, "Real-time and remote construction progress monitoring with a quadruped robot using augmented reality," *Buildings*, vol. 12, no. 11, p. 2027, 2022.

[11] K. Afsari, S. Halder, R. King, W. Thabet, J. Serdakowski, S. DeVito, M. Ensafi, and J. Lopez, "Identification of indicators for effectiveness evaluation of four-legged robots in automated construction progress monitoring," in *Construction Research Congress*, 2022.

[12] S. Halder, K. Afsari, E. Chiou, R. Patrick, and K. A. Hamed, "Construction inspection & monitoring with quadruped robots in future human-robot teaming: A preliminary study," *Journal of Building Engineering*, vol. 65, p. 105814, 2023.

[13] R. Siegwart, M. Hutter, P. Oettershagen, M. Burri, I. Gilitschenski, E. Galceran, and J. Nieto, "Legged and flying robots for disaster response," in *World Engineering Conference and Convention (WECC)*, 2015.

[14] T. Klamt, D. Rodriguez, M. Schwarz, C. Lenz, D. Pavlichenko, D. Droeschel, and S. Behnke, "Supervised autonomous locomotion and manipulation for disaster response with a centaur-like robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[15] C. D. Bellicoso, M. Bjelonic, L. Wellhausen, K. Holtmann, F. Günther, M. Tranzatto, P. Fankhauser, and M. Hutter, "Advances in real-world applications for legged robots," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1311–1326, 2018.

[16] J. Delmerico, S. Mintchev, A. Giusti, B. Gromov, K. Melo, T. Horvat, C. Cadena, M. Hutter, A. Ijspeert, D. Floreano *et al.*, "The current state and future outlook of rescue robotics," *Journal of Field Robotics*, vol. 36, no. 7, pp. 1171–1191, 2019.

[17] G. Tournois, M. Focchi, A. Del Prete, R. Orsolino, D. G. Caldwell, and C. Semini, "Online payload identification for quadruped robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[18] M. Sombolestan, Y. Chen, and Q. Nguyen, "Adaptive force-based control for legged robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[19] M. V. Minniti, R. Grandia, F. Farshidian, and M. Hutter, "Adaptive clf-mpc with application to quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 565–572, 2021.

[20] M. Sombolestan and Q. Nguyen, "Adaptive-force-based control of dynamic legged locomotion over uneven terrain," *IEEE Transactions on Robotics*, vol. 40, pp. 2462–2477, 2024.

[21] 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*, 2020.

[22] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[23] Z. Zhuang, Z. Fu, J. Wang, C. G. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao, "Robot parkour learning," in *Conference on Robot Learning*, 2023.

[24] X. Cheng, K. Shi, A. Agarwal, and D. Pathak, "Extreme parkour with legged robots," *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[25] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani, "Data efficient reinforcement learning for legged robots," in *Conference on Robot Learning*, 2020.

[26] X. Song, Y. Yang, K. Choromanski, K. Caluwaerts, W. Gao, C. Finn, and J. Tan, "Rapidly adaptable legged robots via evolutionary meta-learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[27] L. Smith, J. C. Kew, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Legged robots that keep on learning: Fine-tuning locomotion policies in the real world," in *IEEE/RSJ International Conference on Robotics and Automation (ICRA)*, 2022.

[28] L. Smith, J. C. Kew, T. Li, L. Luu, X. B. Peng, S. Ha, J. Tan, and S. Levine, "Learning and adapting agile locomotion skills by transferring experience," *arXiv preprint arXiv:2304.09834*, 2023.

[29] L. Smith, Y. Cao, and S. Levine, "Grow your limits: Continuous improvement with real-world rl for robotic locomotion," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[30] S. Siva, M. Wigness, J. Rogers, and H. Zhang, "Enhancing consistent ground maneuverability by robot adaptation to complex off-road terrains," in *Conference on Robot Learning*, 2021.

[31] X. Zhou, H. Wu, J. Rojas, Z. Xu, and S. Li, *Nonparametric Bayesian Learning for Collaborative Robot Multimodal Introspection*. Springer Nature, 2020.

[32] S. Belkhale, R. Li, G. Kahn, R. McAllister, R. Calandra, and S. Levine, "Model-based meta-reinforcement learning for flight with suspended payloads," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1471–1478, 2021.

[33] X. Zhao, Y. You, A. Laurenzi, N. Kashiri, and N. Tsagarakis, "Locomotion adaptation in heavy payload transportation tasks with the quadruped robot centauro," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[34] G. Valsecchi, N. Rudin, L. Nachtigall, K. Mayer, F. Tischhauser, and M. Hutter, "Barry: A high-payload and agile quadruped robot," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 6939–6946, 2023.

[35] Y. Chen and Q. Nguyen, "Learning agile locomotion and adaptive behaviors via rl-augmented mpc," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[36] S. Lyu, X. Lang, H. Zhao, H. Zhang, P. Ding, and D. Wang, "Rl2ac: Reinforcement learning-based rapid online adaptive control for legged robot robust locomotion," *Robotics: Science and Systems (RSS)*, 2024.

[37] B. Pandit, A. Gupta, M. S. Gadde, A. Johnson, A. K. Shrestha, H. Duan, J. Dao, and A. Fern, "Learning decentralized multi-biped control for payload transport," *8th Annual Conference on Robot Learning*, 2024.

[38] C. Cao and N. Hovakimyan, "Design and analysis of a novel l1 adaptive control architecture with guaranteed transient performance," *IEEE Transactions on Automatic Control*, vol. 53, no. 2, pp. 586–591, 2008.

[39] ——, "Stability margins of l1 adaptive controller: Part ii," in *IEEE American Control Conference*, 2007.

[40] D. Chen, B. Zhou, V. Koltun, and P. Krähenbühl, "Learning by cheating," in *Conference on Robot Learning*, 2020.

[41] Y. Song, S. Kim, and D. Scaramuzza, "Learning quadruped locomotion using differentiable simulation," *arXiv preprint arXiv:2403.14864*, 2024.

[42] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural-fly enables rapid learning for agile flight in strong winds," *Science Robotics*, vol. 7, no. 66, 2022.

[43] T. Chen and H. Chen, "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems," *IEEE transactions on neural networks*, vol. 6, no. 4, pp. 911–917, 1995.

[44] I. Antoniou and G. Lumer, *Generalized functions, Operator Theory, and Dynamical Systems*. CRC Press, 1998, vol. 399.

[45] S. L. Brunton, M. Budisic, E. Kaiser, and J. N. Kutz, "Modern koopman theory for dynamical systems," *SIAM Review*, vol. 64, no. 2, pp. 229–340, 2022.

[46] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via deeponet based on the universal approximation theorem of operators," *Nature machine intelligence*, vol. 3, no. 3, pp. 218–229, 2021.

[47] S. Wang, H. Wang, and P. Perdikaris, "Improved architectures and training algorithms for deep operator networks," *Journal of Scientific Computing*, vol. 92, no. 2, p. 35, 2022.

[48] Z. Zhang, L. Wing Tat, and H. Schaeffer, "Belnet: Basis enhanced learning, a mesh-free neural operator," *Proceedings of the Royal Society A*, vol. 479, no. 2276, p. 20230043, 2023.

[49] S. Goswami, A. Bora, Y. Yu, and G. E. Karniadakis, "Physics-informed deep neural operator networks," in *Machine Learning in Modeling and Simulation: Methods and Applications*. Springer, 2023, pp. 219–254.

[50] S. Lee and Y. Shin, "On the training and generalization of deep operator networks," *SIAM Journal on Scientific Computing*, vol. 46, no. 4, pp. C273–C296, 2024.

[51] G. Lin, C. Moya, and Z. Zhang, "Learning the dynamical response of nonlinear non-autonomous dynamical systems with deep operator neural networks," *Engineering Applications of Artificial Intelligence*, vol. 125, p. 106689, 2023.

[52] Q. Cao, S. Goswami, T. Tripura, S. Chakraborty, and G. E. Karniadakis, "Deep neural operators can predict the real-time response of floating offshore structures under irregular waves," *Computers & Structures*, vol. 291, p. 107228, 2024.

[53] D. Billheimer, "Functional data analysis, edited by jo ramsay and bw silverman," 2007.

[54] J.-L. Wang, J.-M. Chiou, and H.-G. Müller, "Functional data analysis," *Annual Review of Statistics and its Application*, vol. 3, no. 1, pp. 257–295, 2016.

[55] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, "Spectrally-normalized margin bounds for neural networks," *Advances in Neural Information Processing Systems*, 2017.

[56] G. Shi, W. Hönig, Y. Yue, and S.-J. Chung, "Neural-swarm: Decentralized close-proximity multirotor control using learned interactions," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[57] [Online]. Available: https://www.unitree.com/go1