2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Floating-Base Deep Lagrangian Networks

Lucas Schulze[1,6], Juliano Decico Negri[2], Victor Barasuol[3], Vivian Suzano Medeiros[2], Marcelo Becker[2], Jan Peters[1,4,5,6], Oleg Arenz[1,6]

***Abstract*— Grey-box methods for system identification combine deep learning with physics-informed constraints, capturing complex dependencies while improving out-of-distribution generalization. Despite the growing importance of floating-base systems such as humanoids and quadrupeds, current grey-box models ignore their specific physical constraints. For instance, the inertia matrix is not only positive definite but also exhibits branch-induced sparsity and input independence. Moreover, the 6×6 composite spatial inertia of the floating base inherits properties of single-rigid-body inertia matrices. As we show, this includes the triangle inequality on the eigenvalues of the composite rotational inertia. To address the lack of physical consistency in deep learning models of floating-base systems, we introduce a parameterization of inertia matrices that satisfies all these constraints. Inspired by Deep Lagrangian Networks (DeLaN), we train neural networks to predict physically plausible inertia matrices that minimize inverse dynamics error under Lagrangian mechanics. For evaluation, we collected and released a dataset on multiple quadrupeds and humanoids. In these experiments, our Floating-Base Deep Lagrangian Networks (FeLaN) achieve better overall performance on both simulated and real robots, while providing greater physical interpretability.**

I. INTRODUCTION

Accurate dynamic models are essential for simulation, control, and state estimation in robotics. Classical system identification (SysID) constructs a white-box model based on physical principles and estimates its parameters from data. This approach requires prior knowledge of the system, but generalizes well as long as the model assumptions hold. In contrast, black-box methods approximate the dynamics directly from data without any prior knowledge. Although flexible, they often generalize poorly outside the training domain and lack interpretability. Grey-box methods combine data-driven models with physical structure. For example, Hamiltonian Neural Networks (HNN) [1] and Deep Lagrangian Networks (DeLaN) [2] embed Hamiltonian and Lagrangian mechanics into deep learning architectures, respectively. By enforcing physical properties such as conservation laws, these formulations improve interpretability, generalization, and sample efficiency, and have demonstrated success on fixed-base [3] and single rigid-body [4], [5] robots. Moreover, learned function approximators can yield physically consistent models even with partially or fully unobserved states [6], and can be combined with latent representations [7].

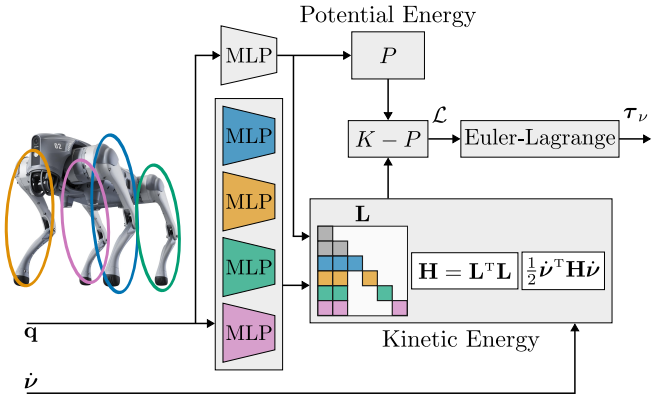


Fig. 1: Floating-Base Deep Lagrangian Networks (FeLaN) for Unitree Go2 quadruped.

However, naively applying these methods to floating-base systems, such as quadrupeds and humanoids, neglects several physical constraints which may limit the generalization of the learned model. These robots have multiple kinematic chains, leading to a sparse inertia matrix with blocks dependent on specific joint inputs. The composite spatial inertia matrix [8], which maps floating-base accelerations to the total wrench applied at the base, must respect a triangle inequality [9]. Even for DeLaN, which predicts the Cholesky decomposition of the inertia matrix, imposing these additional constraints is challenging: sparsity and input-independencies are not preserved in the Cholesky matrix [10] and enforcing the triangle inequality is non-trivial. Hence, grey-box models for floating-base robots remain limited to simplified dynamics [11], [12].

In this work, we introduce a novel parameterization of the inertia matrix of floating-base systems based on the reordered Cholesky factorization [10]. Our parameterization ensures positive definiteness; branch-induced sparsity and input-independencies; the triangle inequality of the composite spatial inertia; and the stationarity of the system total mass. Based on DeLaN, we propose Floating-Base Deep Lagrangian Networks (FeLaN), a novel grey-box method for identifying the dynamics of floating-base systems, depicted in Fig. 1. We validate our method in several simulation environments and on multiple real quadrupeds and

This work was funded by the German Research Foundation (DFG) - Project number PE 2315/18-1, and by the German Federal Ministry of Research, Technology and Space (BMFTR) - Project number 01IS23057B, and as part of the Robotics Institute Germany (RIG). This project has been supported by a hardware donation by NVIDIA through the Academic Grant Program.
[1]Department of Computer Science, Technical University of Darmstadt, Germany. [2]Mobile Robotics Group, São Carlos School of Engineering, University of São Paulo (EESC-USP), Brazil. [3]Dynamic Legged Systems Lab, Istituto Italiano di Tecnologia (IIT). [4]Hessian.AI. [5]German Research Center for AI (DFKI), Research Department: Systems AI for Robot Learning. [6]Robotics Institute Germany (RIG).
Corresponding author: lucas.schulze@robot-learning.de

humanoids: Unitree Go2 [13], Boston Dynamics Spot [14], Pal Robotics Talos [15], and HyQReal2, the updated version of HyQReal [16].

To summarize, our main contributions are as follows:

- We extend the full physical consistency condition of single rigid body to the composite spatial inertia.
- We present a novel parametrization of the inertia matrix for floating-base systems that preserves branch-induced sparsity and ensures full physical consistency of the composite spatial inertia.
- Inspired by DeLaN, we introduce a deep learning architecture for physically consistent system identification of floating-base robots, combining Lagrangian mechanics with our proposed parametrization.
- We perform an extensive comparison of our approach against multiple baselines on both simulated and real robot data.

The remainder of this paper is organized as follows. Section II reviews related work on system identification of floating-base systems. Section III briefly reviews Lagrangian mechanics and DeLaN. We then present our proposed parametrization of the whole-body inertia matrix in Sec. IV and introduce the deep learning architecture built on this parametrization in Sec. V. Section VI describes the compared baselines and the evaluated experiments. Finally, Section VII summarizes our findings and discusses future work.

II. Related Work

Classical SysID techniques for rigid-body systems usually estimate the kinematic parameters (e.g., link lengths, joint axes) separately from the dynamic parameters (e.g., mass, center of mass, inertia), either by prior estimation or by assuming they are given [17]. This separation is motivated by the linearity of the equations of motion in the inertial parameters, enabling estimation via linear least squares under persistent excitation [18]. An alternative approach is to employ differentiable simulators to directly identify all system parameters through stochastic gradient descent [19], [20]. For floating-base systems, prior work has explored different sensing modalities, including joint torques [21], joint torques with force sensors [22], and force plates [23].

Conversely, black-box methods employ nonparametric models to learn either the full system dynamics or the residual with respect to a nominal model. For floating-base systems, used methods include Gaussian process regression [24], [25]; and neural networks [26], [27]. However, these methods usually model only the forward or inverse dynamics, not holding physical properties.

Integrating physics into deep learning demonstrated a useful inductive bias, especially in small data regimes [28]. For floating-base robots, [11] used DeLaN to estimate a simplified formulation of a quadruped's inertia matrix for model-based locomotion. However, only positive definiteness of the inertia is enforced, a necessary but not sufficient condition for full physical consistency, as we prove in Sec. IV-A.

As the generalized coordinates of a floating-base system include the base pose in $\mathrm{SE}(3)$, one could decide to represent the robot's configuration by the pose of each body in $\mathrm{SE}(3)$ instead of joint coordinates. For example, [29], [30], [31] follow this approach, where the inertia matrix simplifies to a block-diagonal concatenation of each body's spatial inertia. While this formulation can ensure full physical consistency, it introduces redundant coordinates and requires knowledge of the system kinematics, as the body poses are unobserved states. In addition, the reported applications were restricted to systems with a few number of degrees of freedom, e.g., planar pendulums, quadrotors.

Noether's theorem connects symmetries and conservation laws, motivating their use as an inductive bias [28]. For instance, [32] and [33] add symmetry constraints for ground reaction force estimation, demonstrating significant improvements over black-box baselines. However, these methods depend on prior knowledge of the robot's symmetry groups, which is not always available or generalizable to any robot.

Floating-base systems may have multiple kinematic branches only indirectly coupled through the base, yielding partial functional decoupling. To exploit this property, [34] learns an inverse dynamics function for each leg, including the base position expressed in the leg frame, which requires kinematic information. In fact, the partial functional decoupling is truly described by the branch-induced sparsity of the inertia matrix [8], which was first leveraged by [10] for efficient inertia matrix factorization.

Unlike structural constraints, Sparse Identification of Nonlinear Dynamics (SINDy) [35] promotes sparsity through optimization. [12] combines linear autoencoders with SINDy to learn a reduced-order quadruped model. While this enforces sparsity, it fails to capture branch-induced sparsity and to guarantee the composite spatial inertia properties.

III. Preliminaries

A. Notation and Definitions

The $\mathrm{n} \times \mathrm{n}$ identity matrix is denoted as $\mathbf{1}_{\mathrm{n}}$. $\mathbf{S}(\mathbf{a})$ and $\mathbf{S}_{\mathbf{a}}$ denote the skew-symmetric operator over the vector $\mathbf{a}$. $\mathbf{A} \succeq 0$ and $\mathbf{A} \succ 0$ represents that the symmetric matrix $\mathbf{A}$ is positive semidefinite and definite, respectively. The superscript $^{\mathbf{W}}$ denotes a variable in the world frame, whereas all other variables are expressed in the base frame. The dependence of matrices and vectors on configuration variables is written explicitly at first mention.

B. Lagrangian Mechanics

Lagrangian mechanics formulates rigid-body dynamics in terms of the Lagrangian

$$\mathcal{L}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}}) = K(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}}) - P(\boldsymbol{\nu}) = \frac{1}{2}\dot{\boldsymbol{\nu}}^{\mathrm{T}}\mathbf{H}(\boldsymbol{\nu})\dot{\boldsymbol{\nu}} - P(\boldsymbol{\nu}), \quad (1)$$

where $\boldsymbol{\nu}$ and $\dot{\boldsymbol{\nu}}$ are generalized positions and velocities, K and P are the kinetic and potential energies, and $\mathbf{H}(\boldsymbol{\nu})$ is the positive-definite inertia matrix.

The equations of motion can be derived from the Euler-Lagrange equation with non-conservative forces $\boldsymbol{\tau}_\nu$

$$\frac{\partial^2 \mathcal{L}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}})}{\partial^2 \dot{\boldsymbol{\nu}}}\ddot{\boldsymbol{\nu}} + \frac{\partial \mathcal{L}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}})}{\partial \boldsymbol{\nu} \partial \dot{\boldsymbol{\nu}}}\dot{\boldsymbol{\nu}} - \frac{\partial \mathcal{L}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}})}{\partial \boldsymbol{\nu}} = \boldsymbol{\tau}_\nu, \quad (2)$$

where $\boldsymbol{\tau}_\nu$ includes all external forces acting on the system, e.g., actuated joint torques, contact forces. Equation (2) can be written in the following compact form

$$\mathbf{H}(\boldsymbol{\nu})\dot{\boldsymbol{\nu}} + \mathbf{c}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}}) + \mathbf{g}(\boldsymbol{\nu}) = \boldsymbol{\tau}_\nu, \tag{3}$$

where $\mathbf{c}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}})$ and $\mathbf{g}(\boldsymbol{\nu})$ are the generalized Coriolis and gravitational forces, respectively.

C. Deep Lagrangian Networks

Based on (1), DeLaN [36] approximates the joint-space inertia matrix $\mathbf{H}$ and P using two networks to estimate the equations of motion (3) of a fixed-base robot. To guarantee physical consistency, i.e., $\mathbf{H} \succ 0$, DeLaN parameterizes the inertia matrix in terms of its Cholesky decomposition, i.e.

$$\mathbf{H}(\mathbf{q}) = \mathbf{C}(\mathbf{q})\mathbf{C}(\mathbf{q})^{\mathrm{T}}, \tag{4}$$

where $\mathbf{C}(\mathbf{q})$ is a lower triangular matrix with positive diagonal elements.

D. Fully Physically-Consistent Spatial Inertia

The inertia of a rigid body is defined by its mass $\mathrm{m}_b \in \mathbb{R}$, the position of the center of mass in the body frame $\mathbf{r}_\mathrm{c} \in \mathbb{R}^3$, the first mass moment $\mathbf{h}_b = \mathrm{m}_b \mathbf{r}_\mathrm{c}$, and the rotational inertia $\mathbf{I}_b \in \mathbb{R}^{3\times3}$ defined at the body frame. These parameters define the body spatial inertia [8]

$$\mathbf{M}_b = \begin{bmatrix} \mathrm{m}_b \mathbf{1}_3 & \mathbf{S}(\mathbf{h}_b)^{\mathrm{T}} \\ \mathbf{S}(\mathbf{h}_b) & \mathbf{I}_b \end{bmatrix}. \tag{5}$$

As $\mathbf{I}_b$ results from the principal moment of inertia and the positivity of the mass density function, [9] demonstrated that the eigenvalues of $\mathbf{I}_b$, i.e., I_x, I_y, I_z, are not only positive, but also satisfies the triangle inequality

$$\mathrm{I}_x + \mathrm{I}_y \geq \mathrm{I}_z,\ \mathrm{I}_y + \mathrm{I}_z \geq \mathrm{I}_x,\ \mathrm{I}_x + \mathrm{I}_z \geq \mathrm{I}_y, \tag{6}$$

which is equivalent to

$$\frac{1}{2}\mathrm{Tr}(\mathbf{I}_b) \geq \lambda_{\max}(\mathbf{I}_b) \tag{7}$$

where $\lambda_{\max}(\mathbf{I}_b)$ is the largest eigenvalue of $\mathbf{I}_b$. Hence, full physical consistency is only defined for $\mathrm{m}_b > 0$ and (6) fulfilled. From (7), [37] observed that the triangle inequality is equivalent to requiring a positive semidefinite covariance $\boldsymbol{\Sigma}_\mathrm{b}$ of the body's mass distribution, which are related through

$$\mathbf{I}_b = \mathrm{Tr}(\boldsymbol{\Sigma}_\mathrm{b})\mathbf{1}_3 - \boldsymbol{\Sigma}_\mathrm{b}. \tag{8}$$

IV. Physically Consistent Inertia Matrix Parametrization

In this section, we investigate the floating-base inertia matrix $\mathbf{H}$ and extend full physical consistency to the composite spatial inertia. We then propose an unconstrained parametrization of $\mathbf{H}$ that addresses all identified constraints.

A. Inertia Matrix of a Floating-Base system

For floating-base systems, the generalized coordinates account for both base pose $\mathbf{q}_\mathbf{B} \in \mathrm{SE}(3)$, and joint position $\mathbf{q} \in \mathbb{R}^{\mathrm{n_q}}$, leading to $\boldsymbol{\nu} = [\mathbf{q}_\mathbf{B}; \mathbf{q}]$. The top-left 6×6 block matrix of $\mathbf{H}$ corresponds to the composite spatial inertia $\mathbf{H}_\mathbf{B}(\mathbf{q})$ of the whole system [8]. As in (5), $\mathbf{H}_\mathbf{B}$ is defined in the base frame from the total mass m, total first mass moment $\mathbf{h}(\mathbf{q})$, and composite rotational inertia $\mathbf{I}_\mathbf{B}(\mathbf{q})$ as

$$\mathbf{H}_\mathbf{B}(\mathbf{q}) = \begin{bmatrix} \mathrm{m}\mathbf{1}_3 & \mathbf{S}(\mathbf{h}(\mathbf{q}))^{\mathrm{T}} \\ \mathbf{S}(\mathbf{h}(\mathbf{q})) & \mathbf{I}_\mathbf{B}(\mathbf{q}) \end{bmatrix}. \tag{9}$$

Unlike the spatial inertia of a single rigid body, both $\mathbf{h}(\mathbf{q})$ and $\mathbf{I}_\mathbf{B}(\mathbf{q})$ depend on the joint configuration $\mathbf{q}$ due to the transformation from body to base frame, whereas m remains constant and results in an isotropic linear inertia. Since $\mathbf{H}_\mathbf{B}$ has the same structure as the spatial inertia of a single rigid body (5), $\mathbf{H}_\mathbf{B} \succ 0$ is a necessary but not sufficient condition for full physical consistency, as established by Lemma 1.

Lemma 1: A composite inertia matrix is fully physically consistent only if $\mathrm{m} > 0$ and $\mathbf{I}_\mathbf{B}(\mathbf{q})$ satisfies the triangle inequality.

Proof: Through the parallel axis theorem, the rotational inertia of the ith body with respect to the base frame is

$$\mathbf{I}_i(\mathbf{q}) = \mathbf{R}_i(\mathbf{q})\mathbf{I}_{bi}\mathbf{R}_i(\mathbf{q})^{\mathrm{T}} + \mathrm{m}_i\mathbf{S}(\mathbf{r}_i(\mathbf{q}))\mathbf{S}(\mathbf{r}_i(\mathbf{q}))^{\mathrm{T}}, \tag{10}$$

where $\mathbf{R}_i(\mathbf{q}) \in \mathrm{SO}(3)$ is the rotation from body to base frame, $\mathbf{r}_i$ is the body position w.r.t. the base. The parallel axis theorem preserves the triangle inequality, as it corresponds to a change of reference frame [9]. From (7), the trace of the composite rotational inertia satisfies a minimum bound:

$$\mathrm{Tr}(\mathbf{I}_\mathbf{B}(\mathbf{q})) = \sum_{i=1}^{\mathrm{n_b}} \mathrm{Tr}(\mathbf{I}_i(\mathbf{q})) \geq \sum_{i=1}^{\mathrm{n_b}} 2\lambda_{\max}(\mathbf{I}_i(\mathbf{q})). \tag{11}$$

Yet, $\mathbf{I}_i \succ 0$ implies in $\sum_{i=1}^{\mathrm{n_b}} \lambda_{\max}(\mathbf{I}_i(\mathbf{q})) \geq \lambda_{\max}(\mathbf{I}_\mathbf{B}(\mathbf{q}))$, that combined with (11), yields

$$\frac{1}{2}\mathrm{Tr}(\mathbf{I}_\mathbf{B}(\mathbf{q})) \geq \sum_{i=1}^{\mathrm{n_b}} \lambda_{\max}(\mathbf{I}_{bi}(\mathbf{q})) \geq \lambda_{\max}(\mathbf{I}_\mathbf{B}(\mathbf{q})), \tag{12}$$

proving the triangle inequality. ∎

Note that by expressing $\mathbf{H}_\mathbf{B}$ in the base frame, $\mathbf{H}$ depends only on $\mathbf{q}$ rather than on $\boldsymbol{\nu}$, yielding translational and rotational invariance with respect to the base coordinates.

Due to the branch-induced sparsity, the joint inertia $\mathbf{H}_k$ of the kth kinematic branch is a function only of the kth branch joints $\mathbf{q}_k$, along with the linear and rotational inertial coupling with the robot's base.

B. Branch-Induced Sparsity

For fixed-base robots with a single kinematic branch, a positive-definite inertia matrix can be estimated from the Cholesky factor (4) [2]. However, as established by Lemma 1, $\mathbf{H} \succ 0$ is a necessary but not a sufficient condition for a physically consistent $\mathbf{H}$ of a floating-base system.

The main limitation is that the standard Cholesky factorization (4) typically results in a dense factor $\mathbf{C}(\mathbf{q})$ [10], not

holding the branched-induced sparsity and not necessarily the structure of $\mathbf{H_B}(\mathbf{q})$. To exploit the branch-induced sparsity and efficiently factorize a given $\mathbf{H}$, [10] proposes to use a reordered Cholesky factorization

$$\mathbf{H}(\mathbf{q}) = \mathbf{L}(\mathbf{q})^{\mathrm{T}}\mathbf{L}(\mathbf{q}), \tag{13}$$

where $\mathbf{L}(\mathbf{q})$ is a lower triangular matrix as $\mathbf{C}(\mathbf{q})$.

The reordered factorization in (13) is equivalent to a standard Cholesky factorization on a permutation of the original matrix. It preserves the same sparsity pattern from $\mathbf{H}$ in $\mathbf{L}$, while having the same numerical properties, e.g., positive definiteness. The only additional information required is the bodies' connectivity graph, i.e., the kinematic chains.

For a robot with $\mathrm{n_k}$ kinematic branches, note that $\mathbf{L}$ has the following structure:

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_{\mathrm{L}} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{L}_{\mathrm{RL}} & \mathbf{L}_{\mathrm{R}} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{L}_{1\mathrm{L}} & \mathbf{L}_{1\mathrm{R}} & \mathbf{L}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{L}_{2\mathrm{L}} & \mathbf{L}_{2\mathrm{R}} & \mathbf{0} & \mathbf{L}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \ddots \\ \mathbf{L}_{\mathrm{n_k L}} & \mathbf{L}_{\mathrm{n_k R}} & \mathbf{0} & \mathbf{0} & \dots & \mathbf{L}_{\mathrm{n_k}} \end{bmatrix} \tag{14}$$

where $\mathbf{L}_{\mathrm{L}}(\mathbf{q})$, $\mathbf{L}_{\mathrm{RL}}(\mathbf{q})$, $\mathbf{L}_{\mathrm{R}}(\mathbf{q})$ are a function of all joints $\mathbf{q}$; $\mathbf{L}_{k\mathrm{L}}(\mathbf{q}_k)$, $\mathbf{L}_{k\mathrm{R}}(\mathbf{q}_k)$, $\mathbf{L}_k(\mathbf{q}_k)$ depend only on $\mathbf{q}_k$ of the kth kinematic branch.

Therefore, any matrix $\mathbf{L}$ with the sparsity and functional characteristics in (14) implies the same properties in $\mathbf{H}$ using (13). Unless otherwise stated, $\mathbf{L}$ will hereafter denote the one defined in (13).

Based on the structure in (14), we parametrize its matrix blocks to fulfill all physical constraints of $\mathbf{H_B}(\mathbf{q})$. The following subsections detail how these constraints are ensured.

C. Positive Definiteness

As the reordered Cholesky (13) is equivalent to a factorization on a permuted matrix, $\mathbf{L}$ with positive diagonal implies $\mathbf{H} \succ 0$. More precisely, this can be achieved by ensuring all the diagonal blocks (e.g., $\mathbf{L}_{\mathrm{L}}$, $\mathbf{L}_{\mathrm{R}}$, $\mathbf{L}_1$, $\dots$, $\mathbf{L}_{\mathrm{n_k}}$) to have positive diagonal elements.

D. Triangle Inequality

The relation (8) between a semidefinite covariance matrix and the rotational inertia is defined in any arbitrary frame [37]. Therefore, we expand its concept to the composite spatial inertia using Lemma 1, relating a configuration dependent covariance matrix $\mathbf{\Sigma}_{\mathrm{R}}(\mathbf{q}) \succ 0$ to a fully physically consistent $\mathbf{I_B}(\mathbf{q})$ by

$$\mathbf{I_B} = \mathrm{Tr}(\mathbf{\Sigma}_{\mathrm{R}})\mathbf{1}_3 - \mathbf{\Sigma}_{\mathrm{R}}, \tag{15}$$

where $\mathbf{\Sigma}_{\mathrm{R}}(\mathbf{q}) = \mathbf{L}_{\Sigma}(\mathbf{q})^{\mathrm{T}}\mathbf{L}_{\Sigma}(\mathbf{q})$ with $\mathbf{L}_{\Sigma}(\mathbf{q}) \succ 0$.

To apply this parameterization to $\mathbf{I_B}(\mathbf{q})$, we expand (13) with (14) to obtain

$$\mathbf{L}_{\mathrm{R}}^{\mathrm{T}}\mathbf{L}_{\mathrm{R}} + \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I_B}, \tag{16}$$

where $\mathbf{W}(\mathbf{q}) = [\mathbf{L}_{1\mathrm{R}}(\mathbf{q}_1); \dots; \mathbf{L}_{\mathrm{n_k R}}(\mathbf{q}_{\mathrm{n_k}})]$. Hence, we can express $\mathbf{L}_{\mathrm{R}}$ in terms of $\mathbf{\Sigma}_{\mathrm{R}}$ and $\mathbf{W}(\mathbf{q})$ as

$$\mathbf{L}_{\mathrm{R}}^{\mathrm{T}}\mathbf{L}_{\mathrm{R}} = \mathbf{I_B} - \mathbf{W}^{\mathrm{T}}\mathbf{W} \tag{17}$$

$$= \mathrm{Tr}(\mathbf{\Sigma}_{\mathrm{R}})\mathbf{1}_3 - \mathbf{\Sigma}_{\mathrm{R}} - \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{D}, \tag{18}$$

$$\mathbf{L}_{\mathrm{R}} = \mathrm{reorderedChol}(\mathbf{D}) \text{ with } \mathbf{D} \succ 0, \tag{19}$$

where reorderedChol is the factorization proposed by [10].

E. First mass moment

Considering the linear and angular coupling due to the first mass moment given by $\mathbf{S}(\mathbf{h}(\mathbf{q}))$ in (9), expanding (13) defines the following constraint

$$\mathbf{L}_{\mathrm{RL}}^{\mathrm{T}}\mathbf{L}_{\mathrm{R}} + \mathbf{K}^{\mathrm{T}}\mathbf{W} = \mathbf{S}_{\mathbf{h}}^{\mathrm{T}}, \tag{20}$$

where $\mathbf{K}(\mathbf{q}) = [\mathbf{L}_{1\mathrm{L}}(\mathbf{q}_1); \dots; \mathbf{L}_{\mathrm{n_k L}}(\mathbf{q}_{\mathrm{n_k}})]$. Therefore, the constraint (20) can be fulfilled by expressing $\mathbf{L}_{\mathrm{RL}}(\mathbf{q})$ in terms of $\mathbf{L}_{\mathrm{R}}$, $\mathbf{K}$, $\mathbf{W}$, and the skew-symmetric matrix $\mathbf{S_h}$,

$$\mathbf{L}_{\mathrm{RL}} = ((\mathbf{S}_{\mathbf{h}}^{\mathrm{T}} - \mathbf{K}^{\mathrm{T}}\mathbf{W})\mathbf{L}_{\mathrm{R}}^{-1})^{\mathrm{T}}. \tag{21}$$

F. Mass

Similarly, the following constraint is defined for the linear inertia of the composite spatial inertia

$$\mathbf{L}_{\mathrm{L}}^{\mathrm{T}}\mathbf{L}_{\mathrm{L}} + \mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathrm{m}\mathbf{1}_3, \tag{22}$$

where $\mathbf{U} = [\mathbf{L}_{\mathrm{RL}}; \mathbf{K}]$. Thus, we express $\mathbf{L}_{\mathrm{L}}$ in terms of the total mass $\mathrm{m} > 0$ and $\mathbf{U}$ as

$$\mathbf{L}_{\mathrm{L}}^{\mathrm{T}}\mathbf{L}_{\mathrm{L}} = \mathrm{m}\mathbf{1}_3 - \mathbf{U}^{\mathrm{T}}\mathbf{U} = \mathbf{T}, \tag{23}$$

$$\mathbf{L}_{\mathrm{L}} = \mathrm{reorderedChol}(\mathbf{T}) \text{ with } \mathbf{T} \succ 0. \tag{24}$$

G. Branched Kinematic Chains

Branched kinematic chains, i.e., chains with sub-chains branching from one of their bodies, introduce additional branched-induced sparsity. This type of branching is commonly observed in humanoids, where the arms and head branch from the upper torso, and in quadrupeds with spine joints. Specifically, the factor $\mathbf{L}_{kj}$ associated to the sub-chain j of a chain k depends only on $\mathbf{q}_{kj}$. Consequently, the branch-induced sparsity can be fully exploited through appropriate computation of the elements of $\mathbf{L}$, while the remainder of the parametrization and its associated physical consistency properties are preserved.

H. Parametrization Overview

Given m, $\mathbf{h}$, $\mathbf{L}_{\Sigma}$ and $\mathbf{L}_{k\mathrm{L}}, \mathbf{L}_{k\mathrm{R}}, \mathbf{L}_k$ for each k kinematic branch, we compute an $\mathbf{L}(\mathbf{q})$ which leads to a physically consistent $\mathbf{H}$ through (20), (22), and (24). By directly parameterizing the composite spatial inertia and the joint-space inertia, we require fewer parameters than estimating all 16 kinodynamic parameters for each robot body. Table I shows the number of parameters estimated for each evaluated robot.

V. Floating-Base Deep Lagrangian Networks

In this section, we present how we combine deep learning with the parametrization proposed in Sec. IV.

TABLE I: Number of parameters per robot.

	Go2 / Spot / HyQReal2 ($\mathrm{n_q} = 12$)	Spot with Arm ($\mathrm{n_q} = 19$)	Talos ($\mathrm{n_q} = 22$)
$\mathbf{H}$	324	625	784
Standard $\mathbf{C}$	171	325	406
Reordered $\mathbf{L}$	117	187	210
Proposed	106	176	199
Body Parameters	208	320	368

A. Inertia Matrix

Similar to DeLaN [36], we estimate the elements of $\mathbf{L}(\mathbf{q})$ with deep neural networks. Positive definiteness is satisfied by enforcing positive diagonal elements, achieved by applying an activation function, e.g., ReLU, softplus, combined with an offset ϵ_L.

To enforce input-independencies for each kinematic branch, we define $\mathrm{n_k}$ networks $\mathbf{NN}_k$ to estimate $\mathbf{L}_{k\mathrm{L}}$, $\mathbf{L}_{k\mathrm{R}}$, and $\mathbf{L}_k$ only from the joints $\mathbf{q}_k$ of the kth branch. Meanwhile, $\mathbf{L}_\mathrm{R}$ and $\mathbf{L}_\Sigma$ are estimated by a single network $\mathbf{NN}_\mathrm{R}$ that takes all joints $\mathbf{q}$ as input, while m is computed by θ_m^2, where $\theta_\mathrm{m} \in \mathbb{R}$ is an unconstrained trainable scalar.

Note that (17) requires $\mathbf{D} \succ 0$, which is not necessarily true only from the network's outputs. Therefore, to ensure a proper factorization, we define $\hat{\mathbf{D}}$ using a shifted term β as

$$\beta = \epsilon_\mathrm{D} + \mathrm{softplus}(-\mu_\mathbf{D}), \tag{25}$$

$$\hat{\mathbf{D}} = \mathbf{D} + \beta \mathbf{1}_3, \tag{26}$$

where $\mu_\mathbf{D}$ is the smallest eigenvalue of $\mathbf{D}$, and ϵ_D is a small offset. Analogously, we ensure $\mathbf{T} \succ 0$ in (24) by

$$\hat{\mathrm{m}} = \mathrm{softplus}(\mathrm{m} - \max(\boldsymbol{\lambda}_\mathbf{U})) + \epsilon_\mathrm{m} + \max(\boldsymbol{\lambda}_\mathbf{U}), \tag{27}$$

$$\mathbf{T} = \hat{\mathrm{m}} \mathbf{1}_3 - \mathbf{U}^\mathrm{T}\mathbf{U} \tag{28}$$

where $\boldsymbol{\lambda}_\mathbf{U}$ are the eigenvalues of the product $\mathbf{U}^\mathrm{T}\mathbf{U}$ and ϵ_m is a small offset. Both $\mathbf{U}^\mathrm{T}\mathbf{U}$ and $\mathbf{W}^\mathrm{T}\mathbf{W}$ can be interpreted as the contribution of all the kinematic branches to the composite spatial inertia. Therefore, ϵ_m and ϵ_D can be interpreted as minimal bounds regarding the contribution of the robot's base to the spatial inertia. Furthermore, $\hat{\mathrm{m}}$ and $\hat{\mathbf{D}}$ do not affect any of the physical consistency constraints. Finally, Algorithm 1 summarizes the inertia matrix computation from the networks' outputs and (21), (23), (27), (28).

B. Potential Energy

Consider the potential energy of a system with $\mathrm{n_b}$ bodies,

$$P(\mathbf{q}) = \sum_{i=1}^{\mathrm{n_b}} \mathrm{m}_i \mathbf{g}^\mathrm{T} \mathbf{r}_i^\mathbf{W} = \sum_{i=1}^{\mathrm{n_b}} \mathrm{m}_i \mathbf{g}^\mathrm{T} (\mathbf{r}_\mathrm{U}^\mathbf{W} + \mathbf{R}\mathbf{r}_i(\mathbf{q})) \tag{29}$$

$$= \mathbf{g}^\mathrm{T} \left(\sum_{i=1}^{\mathrm{n_b}} \mathrm{m}_i \mathbf{r}_\mathrm{U}^\mathbf{W} + \mathbf{R} \sum_{i=1}^{\mathrm{n_b}} \mathrm{m}_i \mathbf{r}_i(\mathbf{q}) \right), \tag{30}$$

where $\mathbf{g} \in \mathbb{R}^3$ is the gravity vector, $\mathbf{r}_\mathrm{U}^\mathbf{W} \in \mathbb{R}^3$ is the robot's base position in the world frame, $\mathbf{R}$ is the rotation matrix from base to world frame. Note that the first sum yields the total mass m, and the last one the first mass moment $\mathbf{h}$

$$P(\mathbf{q}) = \mathbf{g}^\mathrm{T} (\mathrm{m}\mathbf{r}_\mathrm{U}^\mathbf{W} + \mathbf{R}\mathbf{h}(\mathbf{q})). \tag{31}$$

Algorithm 1 Inertia Matrix Computation

Require: joint position $\mathbf{q}$, parameters $(\theta_\mathrm{m}, \boldsymbol{\phi}_\mathrm{R}, \ldots, \boldsymbol{\phi}_{\mathrm{n_k}})$

$\mathbf{h}$, $\mathbf{L}_\Sigma = \mathbf{NN}_\mathrm{R}(\mathbf{q}, \boldsymbol{\phi}_\mathrm{R})$, $\mathrm{m} = \theta_\mathrm{m}^2$
for every $k \in \{1, \ldots, \mathrm{n_k}\}$ **do**
  $\mathbf{L}_{k\mathrm{L}}, \mathbf{L}_{k\mathrm{R}}, \mathbf{L}_k = \mathbf{NN}_k(\mathbf{q}_k, \boldsymbol{\phi}_k)$
end for
$\boldsymbol{\Sigma}_\mathrm{R} = \mathbf{L}_\Sigma^\mathrm{T} \mathbf{L}_\Sigma$
$\mathbf{K} = [\mathbf{L}_{1\mathrm{L}}; \ldots, \mathbf{L}_{\mathrm{n_k}\mathrm{L}}]$; $\mathbf{W} = [\mathbf{L}_{1\mathrm{R}}; \ldots; \mathbf{L}_{\mathrm{n_k}\mathrm{R}}]$
$\mathbf{D} = \mathrm{Tr}(\boldsymbol{\Sigma}_\mathrm{R}) \mathbf{1}_3 - \boldsymbol{\Sigma}_\mathrm{R} - \mathbf{W}^\mathrm{T}\mathbf{W}$
$\beta = \epsilon_\mathrm{D} + \mathrm{softplus}(-\mu_\mathbf{D})$
$\hat{\mathbf{D}} = \mathbf{D} + \beta \mathbf{1}_3$
$\mathbf{L}_\mathrm{R} = \mathrm{reorderedChol}(\hat{\mathbf{D}})$
$\mathbf{L}_\mathrm{RL} = ((\mathbf{S_h} - \mathbf{K}^\mathrm{T}\mathbf{W}) \mathbf{L}_\mathrm{R}^{-1})^\mathrm{T}$
$\mathbf{U} = [\mathbf{L}_\mathrm{RL}; \mathbf{K}]$
$\hat{\mathrm{m}} = \mathrm{softplus}(\mathrm{m} - \max(\boldsymbol{\lambda}_\mathbf{U})) + \epsilon_\mathrm{m} + \max(\boldsymbol{\lambda}_\mathbf{U})$
$\mathbf{T} = \hat{\mathrm{m}} \mathbf{1}_3 - \mathbf{U}^\mathrm{T}\mathbf{U}$
$\mathbf{L}_\mathrm{L} = \mathrm{reorderedChol}(\mathbf{T})$
Assemble $\mathbf{L}$ in (14) with $\mathbf{L}_\mathrm{L}, \mathbf{U}, \mathbf{L}_\mathrm{R}, \mathbf{W}, \mathbf{L}_1, \ldots, \mathbf{L}_{\mathrm{n_k}}$
$\mathbf{H} = \mathbf{L}^\mathrm{T}\mathbf{L}$
return $\hat{\mathrm{m}}$, $\mathbf{h}$, $\mathbf{H}$

Therefore, we compute the potential energy directly from the estimated terms of $\mathbf{H}$, without another network.

C. Generalized Coordinates

For floating-base systems, the orientation is usually described by Euler angles $\boldsymbol{\Theta}$ or quaternions, while the angular velocities are used for the generalized velocities. Consequently, the generalized velocities are not the time derivative of the generalized position. To enable automatic differentiation, we convert the angular velocities to Euler angles time derivatives with $\dot{\boldsymbol{\Theta}} = \mathbf{W}_\eta(\boldsymbol{\Theta})\boldsymbol{\omega}$. This transformation also requires converting the angular wrenches to generalized torques. As the potential energy is defined in the world frame, $\mathbf{H}$ in (13) also has to be transformed from base to world frame. Therefore, we apply the following transformations to the total external torque and inertia matrix:

$$\boldsymbol{\tau}_\nu = \begin{bmatrix} \mathbf{1}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_\eta(\boldsymbol{\Theta})^\mathrm{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1}_{\mathrm{n_q}} \end{bmatrix} \boldsymbol{\tau}, \quad \mathbf{H}^\mathbf{W} = \mathbf{T}_\mathbf{H}^\mathrm{T} \mathbf{H} \mathbf{T}_\mathbf{H}, \tag{32}$$

with

$$\mathbf{T}_\mathbf{H} = \begin{bmatrix} \mathbf{R}(\boldsymbol{\Theta})^\mathrm{T} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{W}_\eta(\boldsymbol{\Theta})^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{1}_{\mathrm{n_q}} \end{bmatrix}. \tag{33}$$

D. Loss Function

The Lagrangian (1) cannot be learned directly with standard supervised learning, as the system's energy is not observed. Following [36], the energies are learned indirectly by minimizing the error from the estimated torque $f^{-1}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}}, \ddot{\boldsymbol{\nu}}, \boldsymbol{\phi})$ given by Algorithm 2, by optimizing

$$\boldsymbol{\phi}^* = \arg\min_{\boldsymbol{\phi}} \left\| \boldsymbol{\tau}_\nu - f^{-1}(\boldsymbol{\nu}, \dot{\boldsymbol{\nu}}, \ddot{\boldsymbol{\nu}}, \boldsymbol{\phi}) \right\|_{\mathbf{W}_\tau}^2 \tag{34}$$

$$+ w_\mathbf{U} \textstyle\sum_{i=1}^{3} \mathrm{softplus}(\boldsymbol{\lambda}_{\mathbf{U}i} - \mathrm{m}) + w_\mathbf{D} \mathrm{softplus}(-\mu_\mathbf{D})^2$$

TABLE II: Inverse dynamics NMSE over 10 seeds for simulated Go2 and Talos.

Method	Go2		Talos		rNMSE
	Train	Test	Train	Test	
MLP	7.5e-2 $\pm$ 1.1e-2	1.1e-1 $\pm$ 3.1e-2	1.9e0 $\pm$ 1.8e-1	3.9e0 $\pm$ 4.9e-1	9.9e-1
DeLaN	2.2e-3 $\pm$ 1.4e-3	4.2e-3 $\pm$ 6.3e-3	1.7e-1 $\pm$ 1.3e-2	2.8e-1 $\pm$ 4.7e-2	4.4e-2
DeLaN-PP	4.0e-3 $\pm$ 1.9e-3	6.9e-3 $\pm$ 6.9e-3	8.6e-1 $\pm$ 1.4e0	1.1e0 $\pm$ 1.5e0	1.7e-1
FeLaN-BS	1.2e-3 $\pm$ 5.5e-4	1.9e-3 $\pm$ 2.8e-3	5.1e-2 $\pm$ 1.0e-2	1.2e-1 $\pm$ 1.5e-1	1.3e-2
FeLaN	1.4e-3 $\pm$ 5.5e-4	1.7e-3 $\pm$ 1.2e-3	**3.3e-2 $\pm$ 1.1e-2**	**6.7e-2 $\pm$ 3.6e-2**	**4.9e-3**
DNEA NS	3.2e-2 $\pm$ 5.7e-2	2.9e-2 $\pm$ 5.0e-2	3.8e-1 $\pm$ 6.2e-1	4.7e-1 $\pm$ 6.7e-1	1.8e-1
DNEA PD	1.7e-3 $\pm$ 4.7e-3	1.4e-3 $\pm$ 3.5e-3	3.6e-1 $\pm$ 7.0e-1	4.3e-1 $\pm$ 7.6e-1	5.0e-2
DNEA Tri	9.5e-4 $\pm$ 1.6e-3	8.0e-4 $\pm$ 1.1e-3	5.5e-1 $\pm$ 9.8e-1	6.8e-1 $\pm$ 1.1e0	8.0e-2
DNEA Cov	**6.8e-4 $\pm$ 4.9e-5**	6.4e-4 $\pm$ 2.6e-4	2.5e-1 $\pm$ 6.8e-1	2.8e-1 $\pm$ 7.1e-1	2.8e-2
DNEA Log	6.9e-4 $\pm$ 1.3e-4	**6.2e-4 $\pm$ 1.1e-4**	3.0e-1 $\pm$ 8.2e-1	3.1e-1 $\pm$ 7.6e-1	3.2e-2

where $\mathbf{W}_{\boldsymbol{\tau}}$ is the diagonal covariance of $\boldsymbol{\tau}_\nu$ in the training set; $w_{\mathbf{U}}$ and $w_{\mathbf{D}}$ weight the auxiliaries terms from (25) and (26). Note that physical consistency is achieved as a hard constraint due to the structured parametrization, the extra terms only encourage m and $\mathbf{D}$ to be positive and positive-definite, respectively, which empirically accelerated training during our experiments. We also apply the input layer $\mathbf{T}_{\mathbf{q}}(\mathbf{q}) = [\cos(\mathbf{q}); \sin(\mathbf{q})]$ to each network, as in [36].

Algorithm 2 Inverse Dynamics

Require: $\boldsymbol{\nu}$, $\dot{\boldsymbol{\nu}}$, $\ddot{\boldsymbol{\nu}}$, $\boldsymbol{\phi}$
1: $[\mathbf{r}_{\mathrm{U}}^{\mathbf{W}}; \boldsymbol{\Theta}; \mathbf{q}] = \boldsymbol{\nu}$
2: $\hat{\mathrm{m}}$, $\mathbf{h}(\mathbf{q})$, $\mathbf{H}(\mathbf{q})$ = Algorithm 1$(\mathbf{q}, \boldsymbol{\phi})$
3: P = (31) with $\hat{\mathrm{m}}$, $\mathbf{h}(\mathbf{q})$
4: $\mathbf{H}^{\mathbf{W}} = \mathbf{T}_{\mathbf{H}}^{\mathrm{T}} \mathbf{H} \mathbf{T}_{\mathbf{H}}$
5: $\mathcal{L} = \frac{1}{2} \dot{\boldsymbol{\nu}}^{\mathrm{T}} \mathbf{H}^{\mathbf{W}} \dot{\boldsymbol{\nu}} - P$
6: **return** $\boldsymbol{\tau}_\nu$ = (2) with $\mathcal{L}$

VI. Experiments

We evaluate our proposed architecture for system identification across floating-base robots of different sizes and degrees of freedom on simulation and real-world data.

A. Baselines

To evaluate the trade-off of physics assumptions and estimation accuracy, we compare our method against different baselines for estimating the inverse dynamics model. We employ a multilayer perceptron (MLP) with inputs $\boldsymbol{\Theta}$, $\mathbf{q}$, $\dot{\boldsymbol{\nu}}$, $\ddot{\boldsymbol{\nu}}$. Additionally, we use DeLaN [36] with one network using $\mathbf{q}$ as input to estimate $\mathbf{C}$ in (32), and a second network that uses the entire $\boldsymbol{\nu}$ to estimate $P(\boldsymbol{\nu})$. To evaluate the proposed potential energy parametrization in (31), we implement a modified version of DeLaN, namely DeLaN-PP, which uses a single network for $\mathbf{H}(\mathbf{q})$ and computes P from its estimated terms using (31). We also consider a variant of our proposed network FeLaN, named FeLaN-BS, that incorporates only branch-induced sparsity and positive definiteness. In this variant, $\mathbf{L}_{\mathrm{L}}, \mathbf{L}_{\mathrm{RL}}$, and $\mathbf{L}_{\mathrm{R}}$ are estimated by a single network. We add the input layer $\mathbf{T}_{\mathbf{q}}$ to the inputs of all models, except for MLP and the linear coordinates in DeLaN. We use 2 hidden layers of 32 neurons for MLP, and 2 layers of 16 neurons for the other networks.

Regarding white-box methods, we use Mujoco XLA (MJX) [38] as a differentiable recursive Newton-Euler (DNEA) function, parametrized by 16 kinodynamic parameters per body. In this case, physical consistency is connected to the inertial parameters. Therefore, we evaluate parameterization with different degrees of physical consistency:

- DNEA NS [20]: unconstrained inertial parameters
- DNEA PD [20]: positive definiteness via $\mathbf{I}_b = \mathbf{C}_b \mathbf{C}_b^{\mathrm{T}}$
- DNEA Tri [9]: (6) via second moments of mass
- DNEA Cov [37]: parametrize $\boldsymbol{\Sigma}_{\mathrm{b}} = \mathbf{C}_b \mathbf{C}_b^{\mathrm{T}}$
- DNEA Log [39]: log-Cholesky factorization of the pseudo-inertia matrix

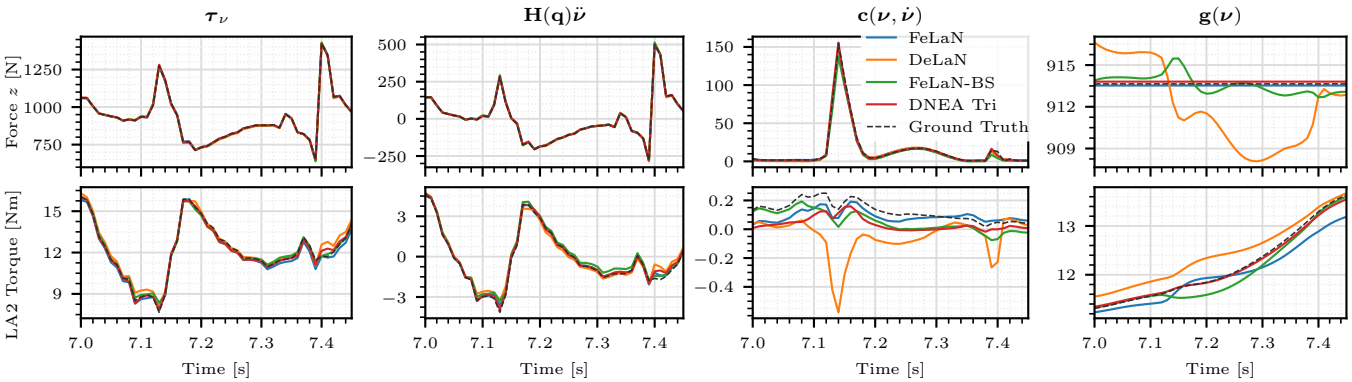
All methods are implemented in JAX [40] and trained with stochastic gradient optimization for 2k epochs on quadrupeds and 3k on Talos. We use two hyperparameter sets: one for Talos and one for all the quadrupeds. Results are averaged over 10 seeds. Code and datasets are available online[1].

B. Simulated Robots

For an initial evaluation on robots mainly governed by rigid-body dynamics, we collect data at 100 Hz for Go2 and Talos in MJX. For Talos, we consider only the first four arm joints, while the remaining joints are kept fixed. To ensure sufficient excitation and data diversity, we uniformly sample desired velocities and base heights and track them with the model predictive controller [41] on both robots. For Talos' arms, we provide sine-wave reference trajectories with uniformly sampled amplitudes and frequencies. Each dataset yields a total of 50k samples. As the ground-truth model is available, we compute $\boldsymbol{\tau}_\nu$ using (3). Table II reports the mean and standard deviation over seeds of the torque mean squared error, normalized by $\mathbf{W}_{\boldsymbol{\tau}}$, for all methods, denoted as NMSE. To compare performance across robots, we define the relative NMSE as $r\mathrm{NMSE}_i = (\mathrm{NMSE}_i - \mathrm{NMSE}_{\min})/\mathrm{NMSE}_{\max}$ for each method on the test set, also shown in Tab. II.

For Go2, the fully physically consistent DNEAs achieve the best performance, as their consistent parametrization fully captures the system's dynamics. All grey-box methods perform well, with FeLaN and FeLaN-BS outperforming DNEA NS, demonstrating the importance of physical consistency. For Talos, FeLaN performed best, followed by FeLaN-BS. Although the amplitude errors remain small, the slightly

[1] https://schulze18.github.io/felan_website



(a) Total Torque (b) Inertial Torque (c) Coriolis Torque (d) Gravitational Torque

Fig. 2: Components of the estimated force z and left arm second joint (LA2) torque for simulated Talos.

TABLE III: Inverse dynamics NMSE over 10 seeds for real robot data.

Method	HyQReal2		Spot		Spot with Arm		Talos		rNMSE	$\overline{r\text{NMSE}}$
	Train	Test	Train	Test	Train	Test	Train	Test		Sim + Real
MLP	$\mathbf{3.0 \pm 0.4}$	$\mathbf{4.5 \pm 0.4}$	$\mathbf{5.9 \pm 0.2}$	$\mathbf{4.2 \pm 0.3}$	$\mathbf{7.8 \pm 0.4}$	$\mathbf{9.3 \pm 0.5}$	28.0 ± 0.1	27.3 ± 0.1	0.202	0.464
DeLaN	92.5 ± 3.6	213.1 ± 104.2	6.9 ± 0.1	5.1 ± 0.1	17.2 ± 12.9	10.7 ± 2.2	$\mathbf{3.8 \pm 0.2}$	$\mathbf{5.2 \pm 2.0}$	0.307	0.219
DeLaN-PP	6.9 ± 1.8	8.2 ± 2.4	8.0 ± 0.2	5.9 ± 0.1	11.1 ± 0.3	9.6 ± 0.2	7.5 ± 0.3	7.3 ± 0.2	**0.086**	0.114
FeLaN-BS	7.6 ± 0.4	9.0 ± 0.6	8.4 ± 0.1	6.3 ± 0.1	10.5 ± 0.2	9.5 ± 0.3	8.1 ± 0.2	7.8 ± 0.2	0.103	0.073
FeLaN	6.6 ± 0.1	7.8 ± 0.2	8.1 ± 0.1	6.0 ± 0.1	10.5 ± 0.2	9.6 ± 0.2	7.5 ± 1.0	7.2 ± 0.5	0.090	**0.062**
DNEA NS	12.8 ± 0.2	14.5 ± 0.2	10.1 ± 0.1	7.6 ± 0.1	14.4 ± 0.2	10.9 ± 0.2	15.9 ± 0.5	14.8 ± 0.5	0.248	0.225
DNEA PD	12.9 ± 0.2	14.7 ± 0.3	10.1 ± 0.1	7.6 ± 0.1	14.4 ± 0.2	10.9 ± 0.2	17.1 ± 1.0	16.0 ± 1.0	0.260	0.190
DNEA Tri	13.0 ± 0.2	14.8 ± 0.2	10.2 ± 0.1	7.6 ± 0.1	14.5 ± 0.2	11.0 ± 0.2	17.5 ± 1.0	16.4 ± 0.8	0.266	0.204
DNEA Cov	13.0 ± 0.2	14.8 ± 0.2	10.1 ± 0.1	7.6 ± 0.1	14.5 ± 0.3	11.0 ± 0.3	17.5 ± 1.0	16.4 ± 0.8	0.266	0.186
DNEA Log	13.2 ± 0.1	15.0 ± 0.2	10.2 ± 0.1	7.6 ± 0.1	14.5 ± 0.2	11.0 ± 0.2	17.7 ± 1.0	16.5 ± 0.9	0.268	0.189

worse performance of DNEA on Talos may be due to contact changes, adding larger spikes in the data when compared to Go2, making the estimation more challenging. DeLaN still performed similarly compared to the other methods, while DeLaN-PP demonstrated sensitivity to different initialization. For both robots, the MLP performed the worst. Accordingly, FeLaN demonstrated the most consistent results, with the lowest rNMSE, followed by FeLaN-BS.

Figure 2 shows the components in (3) for the force z and one torque joint of Talos. All grey-box methods approximate the total value well, see Fig. 2a. FeLaN better predicts the individual terms, in particular the stationary gravity component of z in Fig. 2d, due to its proper mass parametrization.

C. Real Robots

To evaluate all methods on real-world data, we ran experiments on four robots: HyQReal2, Spot, Spot with Arm, and Talos. As a ground truth model is not available, we estimate $\boldsymbol{\tau}_\nu$, which for a legged robot with $\mathrm{n_c}$ contacts is given by

$$\boldsymbol{\tau}_\nu = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\tau}_\mathrm{q} \end{bmatrix} + \sum_{i=1}^{\mathrm{n_c}} \mathbf{J}_{\mathrm{c}\,i}^\mathrm{T} \mathbf{f}_{\mathrm{c}i} \tag{35}$$

where $\boldsymbol{\tau}_\mathrm{q} \in \mathbb{R}^{\mathrm{n_q}}$ are the joint torques; $\mathbf{J}_{\mathrm{c}i}$ and $\mathbf{f}_{\mathrm{c}i}$ are the ith contact Jacobian and force. All evaluated robots have torque sensors. For contact forces, Talos uses an ankle sensor, HyQReal2 relies on a force estimator based on its dynamical model, and Spot's forces were measured with external 6D force plates. Although the estimation of $\boldsymbol{\tau}_\nu$ requires kinematics information, all the methods still learn agnostically to any kinematic parameter. For HyQReal2 and Talos, trajectories similar to those in simulation were executed, producing 75k and 125k samples, respectively. To keep Spot on the force plates, small velocity commands were sent while trotting in place, resulting in 60k samples. For the arm, a position controller tracked sine-wave trajectories with uniformly sampled parameters, yielding 45k samples.

Table III reports the NMSE results, including the weighted average $\overline{r\text{NMSE}}$ over real and simulated robots. Unlike in simulation, real robot dynamics are affected by actuator and contact dynamics, noise, and other unmodeled effects. Consequently, DNEA methods struggle to accurately estimate the torques due to limited model complexity, being in general outperformed by most of the other methods. Following the simulation results on Talos, the performance gap of DNEAs is larger for the heavier robots. On HyQReal2 and Talos, 125 kg and 100 kg, respectively, DNEA errors are roughly three times higher than the best. Their larger mass implies larger contact forces and thereby larger torques than on Spot. This tends to amplify the influence of actuator and contact dynamics that the models do not capture, thereby limiting prediction accuracy. In contrast, DeLaN and MLP achieve the best robot-specific results. Because these methods impose weaker or no physical constraints, they are more flexible in learning input–output mappings that may not be physically consistent. However, they demonstrated sensitivity to initialization and hyperparameter choices, and do not perform

consistently across all robots.

Overall, DeLaN-PP, FeLaN-BS, and FeLaN remain competitive with the best robot-specific models while achieving the best rNMSE. This supports incorporating physics into deep learning by improving consistency across robots, reducing sensitivity to dataset-specific biases. In particular, it highlights the effectiveness of our potential energy parameterization. Considering $\overline{r\text{NMSE}}$, FeLaN achieves the best overall performance across simulation and real robots, followed by FeLaN-BS, underscoring the benefit of branch-induced sparsity and full physical consistency of the composite spatial inertia.

VII. Conclusion

In this work, we introduced a novel physically consistent parametrization of the floating-base inertia matrix, based on a reordered Cholesky factorization, which achieves branch-induced sparsity and ensures full physical consistency of the composite spatial inertia. Building on this parametrization, we proposed FeLaN, a grey-box method for physically consistent system identification of floating-base dynamics. FeLaN learns the whole-body dynamics model using only prior knowledge on the kinematic chains, without requiring any knowledge of dynamics or other kinematic parameters. We validated FeLaN on both simulated and real robot data, showing overall better performance than state-of-the-art methods while offering greater interpretability.

As future work, we plan to explore using FeLaN for locomotion and loco-manipulation tasks, either as a learned dynamical model or alongside adaptive control techniques.

References

[1] S. Greydanus *et al.*, "Hamiltonian neural networks," in *Advances in Neural Information Processing Systems*, 2019.

[2] M. Lutter *et al.*, "Deep lagrangian networks: Using physics as model prior for deep learning," in *International Conference on Learning Representations*, 2019.

[3] M. Lutter, K. Listmann, and J. Peters, "Deep lagrangian networks for end-to-end learning of energy-based control for under-actuated systems," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[4] T. Duong *et al.*, "Port-hamiltonian neural ode networks on lie groups for robot dynamics learning and control," *IEEE Transactions on Robotics*, 2024.

[5] A. Altawaitan *et al.*, "Hamiltonian dynamics learning from point cloud observations for nonholonomic mobile robot control," *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[6] P. Toth *et al.*, "Hamiltonian generative networks," in *International Conference on Learning Representations*, 2020.

[7] L. Schulze, J. Peters, and O. Arenz, "Context-aware deep lagrangian networks for model predictive control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

[8] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer US, 2008.

[9] S. Traversaro *et al.*, "Identification of fully physical consistent inertial parameters using optimization on manifolds," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016.

[10] R. Featherstone, "Efficient factorization of the joint-space inertia matrix for branched kinematic trees," *The International Journal of Robotics Research*, June 2005.

[11] P. Kotecha *et al.*, "Investigating lagrangian neural networks for infinite horizon planning in quadrupedal locomotion," 2025.

[12] G. Buriani *et al.*, "Symbolic learning of interpretable reduced-order models for jumping quadruped robots," 2025.

[13] Unitree Robotics. (2023) Go2 quadruped robot.

[14] Boston Dynamics. (2024) Spot: The agile mobile robot.

[15] O. Stasse *et al.*, "Talos: A new humanoid research platform targeted for industrial applications," in *IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 2017.

[16] C. Semini *et al.*, "Brief introduction to the quadruped robot hyqreal," in *International Conference on Robotics and Intelligent Machines (I-RIM)*. IRIM, 2019.

[17] T. Lee *et al.*, "Robot model identification and learning: A modern perspective," *Annual Review of Control, Robotics, and Autonomous Systems*, 2024.

[18] T. Lee, B. D. Lee, and F. C. Park, "Optimal excitation trajectories for mechanical systems identification," *Automatica*, 2021.

[19] M. Lutter *et al.*, "Differentiable physics models for real-world offline model-based reinforcement learning," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[20] G. Sutanto *et al.*, "Encoding physical constraints in differentiable newton-euler algorithm," in *Proc. 2nd Conference on Learning for Dynamics and Control*. PMLR, 2020.

[21] S. Khorshidi *et al.*, "Physically-consistent parameter identification of robots in contact," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[22] Y. Ogawa *et al.*, "Dynamic parameters identification of a humanoid robot using joint torque sensors and/or contact forces," in *IEEE-RAS International Conference on Humanoid Robots*, 2014.

[23] K. Ayusawa *et al.*, "Identifiability and identification of inertial parameters using the underactuated base-link dynamics for legged multibody systems," *The International Journal of Robotics Research*, 2014.

[24] R. Grandia *et al.*, "Contact invariant model learning for legged robot locomotion," *IEEE Robotics and Automation Letters*, 2018.

[25] E. Arcari *et al.*, "Bayesian multi-task learning mpc for robotic mobile manipulation," *IEEE Robotics and Automation Letters*, 2023.

[26] E. Heiden *et al.*, "Neuralsim: Augmenting differentiable simulators with neural networks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[27] A. Nagabandi *et al.*, "Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[28] J. Watson *et al.*, "Machine learning with physics knowledge for prediction: A survey," *Trans. on Machine Learning Research*, 2025.

[29] M. Finzi *et al.*, "Simplifying hamiltonian and lagrangian neural networks via explicit constraints," *Advances in neural information processing systems*, 2020.

[30] T. Duong and N. Atanasov, "Hamiltonian-based neural ODE networks on the SE(3) manifold for dynamics learning and control," in *Robotics: Science and Systems (RSS)*, 2021.

[31] V. Duruisseaux *et al.*, "Lie group forced variational integrator networks for learning and control of robot systems," in *Learning for Dynamics and Control Conference*, 2023.

[32] D. Ordoñez-Apraez *et al.*, "Morphological symmetries in robotics," *The International Journal of Robotics Research*, 2025.

[33] F. Xie *et al.*, "Morphological-symmetry-equivariant heterogeneous graph neural network for robotic dynamics learning," in *Proc. 7th Annual Learning for Dynamics & Control Conference*. PMLR, 2025.

[34] J.-E. Lee *et al.*, "Sample efficient dynamics learning for symmetrical legged robots: Leveraging physics invariance and geometric symmetries," 2022.

[35] S. L. Brunton *et al.*, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, 2016.

[36] M. Lutter and J. Peters, "Combining physics and deep learning to learn continuous-time dynamics models," *The International Journal of Robotics Research*, 2023.

[37] P. M. Wensing *et al.*, "Linear matrix inequalities for physically consistent inertial parameter identification: A statistical perspective on the mass distribution," *IEEE Robotics and Automation Letters*, 2018.

[38] E. Todorov *et al.*, "Mujoco: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.

[39] C. Rucker and P. M. Wensing, "Smooth parameterization of rigid-body inertia," *IEEE Robotics and Automation Letters*, 2022.

[40] J. Bradbury *et al.*, "JAX: composable transformations of Python+NumPy programs," 2018.

[41] L. Amatucci *et al.*, "Primal-dual ilqr for gpu-accelerated learning and control in legged robots," *IEEE Robotics and Automation Letters*, 2026.