2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Sample Efficient Dynamics Learning for Symmetrical Legged Robots: Leveraging Physics Invariance and Geometric Symmetries

Jee-eun Lee[1] and Jaemin Lee[3] and Tirthankar Bandyopadhyay[2] and Luis Sentis[1]

***Abstract*— Model generalization of the underlying dynamics is critical for achieving data efficiency when learning for robot control. This paper proposes a novel approach for learning dynamics leveraging the symmetry in the underlying robotic system, which allows for robust extrapolation from fewer samples. Existing frameworks that represent all data in vector space fail to consider the structured information of the robot, such as leg symmetry, rotational symmetry, and physics invariance. As a result, these schemes require vast amounts of training data to learn the system's redundant elements because they are learned independently. Instead, we propose considering the geometric prior by representing the system in symmetrical object groups and designing neural network architecture to assess invariance and equivariance between the objects. Finally, we demonstrate the effectiveness of our approach by comparing the generalization to unseen data of the proposed model and the existing models. We also implement a controller of a climbing robot based on learned inverse dynamics models. The results show that our method generates accurate control inputs that help the robot reach the desired state while requiring less training data than existing methods.**

***Index Terms*— Model Learning for Control; Representation Learning; Group-equivalent Neural Networks**

I. Introduction

In need of robots to traverse extreme terrains, various legged systems have been developed recently. For controlling these robots, model-based approaches have been extensively used [1], [2], [3], [4]. However, such strategies often fail due to complex and unpredictable effects generated by environmental interactions, thus they often require specific strategies [5]. Also, model-based control highly depends on the model's accuracy, requiring demanding extra work [6]. Alternatively, model-free approaches have shown remarkable success in this field recently. Using the latest deep reinforcement learning techniques, robots can learn complex tasks [7], [8], and locomotion [9], [10], [11].

However, the lack of physical plausibility in learned models limits their applicability to the vicinity of the training data. Hence, the efficacy of the model depends heavily on the sample complexity. To mitigate this limitation, exhibiting

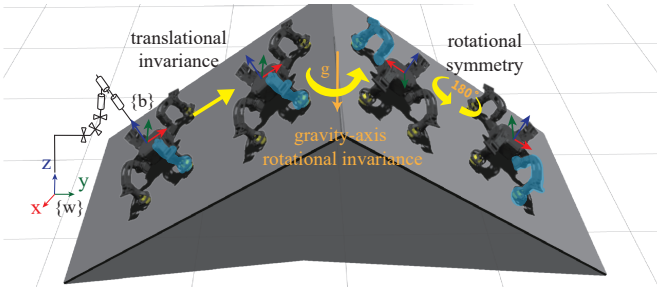

Fig. 1: The figure shows examples of physical invariance found in a floating-based robot where the legs are symmetrical. Robots in different configurations in the figure above have the same physical properties. e.g., the torque required at each joint to track the given motion is the same.

combinatorial generalization is also suggested as a key aspect for modern AI to resemble human intelligence, which can generalize beyond one's experience [12]. There are several network architectures proposed to incorporate prior knowledge. One example is utilizing a differential equation form to encode a physics prior in the Euler-Lagrange equation [13], [14]. By constraining the model to satisfy a differential equation, it extrapolates more accurately to unseen samples. A mixture of experts (MOE) can learn complex dynamics models leveraging model plausibility from a system equation (white-box) while increasing the accuracy with a deep neural network (black-box) [15]. Lastly, by encouraging the symmetric motion, it is shown that the learning could be more efficient and faster [16], [17].

Every floating base robot has translational and gravity-axis rotational invariance in dynamics as shown in Fig. 1. Specifically, multi-legged robots with rotational symmetry e.g. Magneto [18] have a set of configurations under the same physics. However, current schemes that represent the states and actions of the robot as a stacked vector hardly capture these properties, meaning that they only learn these symmetric properties through data, resulting in sample inefficiency and a lack of generalization capability. In this paper, we suggest a network topology that works on structured data –a graph or a set– reflecting the robot symmetry.

One popular neural network for structured data learning is graph neural networks (GNNs) [19],[20] that is known for its ability to consider the relations between objects by using graph structure for learning. In robotics, there are some previous works that use GNNs to learn dynamics [21], [22], [23]. However, most of these researches focus on the model transfer capability of GNNs over various types of robots. Reference [21] emphasizes that the model can predict the

[1]Jee-eun Lee is with Human Centered Robotics Laboratory, Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, USA, jelee@utexas.edu

[3]Jaemin Lee is with the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA, USA, jaemin87@caltech.edu

[1]Luis Sentis is with Faculty of Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, USA, lsentis@austin.utexas.edu

[2]Tirthankar Bandyopadhyay is with Robotics and Autonomous Systems Group, Data61, CSIRO, QLD 4069, Australia tirtha.bandy@csiro.au

dynamics of a new robot(i.e., where its data has not been used for training) by learning the physical relations between links and joints. Notably, GNNs can even be applied to zero-shot transfer learning for modular robots [22], [23].

However, asking models to learn all types of robot physics is unrealistic. It would require tremendous training data; it is not applicable to different sensors and actuators; and sensors produced by different manufacturers can have different performances. Instead, we focus on reducing the sample complexity for system identification using physical priors. In this paper, we introduce new data representations for symmetrical-legged robots and how GNNs and other group-equivariant neural networks can improve sample efficiency based on the permutation equivariance in robot dynamics.

A. Contribution

This paper proposes sample efficient dynamics learning for symmetrical-legged robots. By using grouped-by-leg structured data, we can define group invariance and equivariance followed by geometric symmetries of a robot based on the data representation. Then, by training a model based on the networks designed to preserve the group equivariance, we can learn the dynamics that incorporate physics priors, providing sample efficiency and generalization capability.

II. Preliminaries

Using *a priori* representational and computational assumptions to achieve generalization is actively studied in the deep learning. This can be realized by representing prior knowledge as equivariance or invariance under group actions. Below we introduce the relevant concepts and prior works.

A. Invariance, Equivariance, and Symmetries

In geometry, we say that an object has symmetry if the object has an invariance under the group action. In group theory, group action, invariance, and equivariance can be defined as follows:

Definition 1 (Group Action) A group G is said to act on a set X if there is a map $\psi : G \times X \rightarrow X$ called the Group Action such that $\psi(e, x) = x$ for all $x \in X$, and $\psi(g, \psi(h, x)) = \psi(gh, x)$ for all $g, h \in G, x \in X$. For compactness we follow the practice of denoting a group action by a '$\circ$' e.g. $g \circ x$.

Definition 2 (Invariance) Let G be a group which acts on the sets X. We say that the function $f : X \rightarrow Y$ is G-invariant if $f(g \circ x) = f(x)$ for all $x \in X, g \in G$.

Definition 3 (Equivariance) Let G be a group which acts on the sets X and Y. We say that the function $f : X \rightarrow Y$ is G-equivariant if $f(g \circ x) = g \circ (f(x))$ for all $x \in X, g \in G$.

B. Equivariance and Invariance in Neural Networks

Many works in deep learning leverage equivariance and invariance for constructing the neural network architecture. Data augmentation is a straightforward way to learn group-equivariant representation [24]; however, it is computationally inefficient due to the increased data volume to train the model.

Convolutional Neural Networks One well-known example to consider equivariance and invariance is convolutional neural networks (CNNs) [25], [26], which utilize convolutional layer for preserving object identity over the translation group.

Group-Equivariant Neural Networks A group convolutional layer is proposed to consider other symmetries like rotation and reflection [27],[28]. For more general structures, deep symmetry networks [29] capture a wide variety of invariances defined over arbitrary symmetry groups using kernel-based interpolation and symmetric space pooling. For a set structure like point clouds, simple permutation-equivariant layers [30], [31] are shown to be beneficial.

Graph Neural Networks Graph neural networks [19], [32] is a type of neural networks directly operating on graph structures where a system of objects and relations is represented through nodes and edges. GNNs can capture all permutation invariance, and equivariance [33] in nodes and edges. Graph Networks (GNs), proposed by [12], provides a general form of graph structure for learning, $G = (\mathbf{u}, \{\mathbf{n}_i\}, \{\mathbf{e}_j, s_j, r_j\})$ that includes a graph embedding vector called global features $\mathbf{u}$, a set of node features $\{\mathbf{n}_i\}_{i=1,..,N_n}$ and a set of edges $\{\mathbf{e}_j, s_j, r_j\}_{j=1,..,N_e}$, where $\mathbf{e}_j$ is a vector representing edge features and s_j, r_j are the indices of the sender and receiver nodes ranging from 1 to N_n. This data structure allows GNNs to capture the permutation equivariance of nodes and edges. Then graph networks are defined as a "graph2graph" module, which takes an input graph and returns an output graph with the updated features.

III. Structured Representation for Symmetrical Legged Robot Dynamics

This section describes the equivariance and invariance in inverse dynamics of symmetrical-legged robots. Then, we suggest data representation that can capture those properties in neural networks.

A. Dynamics of Floating-base Legged Robots

The dynamics of legged robots are commonly described as a function of the generalized coordinates $\mathbf{q} = [\mathbf{q}_b^\top, \mathbf{q}_j^\top]^\top \in \mathbb{R}^{n_q}$, where $\mathbf{q}_b \in \mathbb{R}^6, \mathbf{q}_j \in \mathbb{R}^{n_j}$ represent the configuration of the floating base and joints, respectively. As such, the rigid-body dynamics of a floating base robot can be obtained using Euler-Lagrange formalism yielding:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}_a^\top \boldsymbol{\tau}_a + \mathbf{J}_c(\mathbf{q})^\top \mathcal{F}_c \tag{1}$$

where $\mathbf{M}(\mathbf{q})$, $\mathbf{b}(\mathbf{q}, \dot{\mathbf{q}})$ represent the mass/inertia matrix and the Coriolis/centrifugal force plus the gravitational force. $\mathbf{S}_a \in \mathbb{R}^{n_a \times n_q}$ denotes the selection matrix corresponding to the index set of actuated joints, which maps $\boldsymbol{\tau}_a$ (the actuated joint torques) into the generalized forces and, $\mathbf{J}_c(\mathbf{q})$ denotes the stacked contact Jacobian matrix that maps contact wrench vector $\mathcal{F}_c$ into the generalized forces. Finally, from the dynamics equation, we can derive the forward model f_{fwd} and inverse model f_{inv} as:

$$\ddot{\mathbf{q}} = f_{\text{fwd}}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}), \quad \boldsymbol{\tau} = f_{\text{inv}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) \tag{2}$$

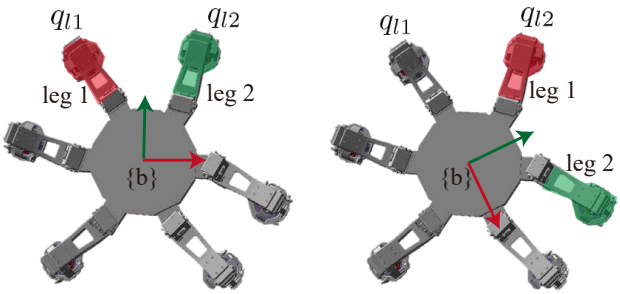

Fig. 2: Rotational equivariance in a symmetric hexapod. If we rotate the hexapod by $\frac{\pi}{3}$ and shift the joint values assigned to legs circularly, we can make a set of robot configurations that looks exactly the same. Then the robots have the same physics applied to each leg, seemingly at the same position.

B. Physics Invariance in Floating-base Robot

Let G_θ be the gravity-axis rotation group and T_p be the translation group acting on the base configuration space $\mathbf{q}_b = [x, y, z, rz, ry, rx]^\top$ as follows:

$$G_\theta \circ \mathbf{q}_b = [x, y, z, rz', ry', rx']^\top,$$
$$T_p \circ \mathbf{q}_b = [x + p_x, y + p_y, z + p_z, rz, ry, rx]^\top$$

where, rz', ry', rx' represent the EulerZYX angle of base configuration that is rotated by θ along the gravity axis. Similarly, we can represent $\dot{\mathbf{q}}_b, \ddot{\mathbf{q}}_b$ when the rotation around the gravity axis and translation groups are applied to the robot. Then, the gravity-axis rotational invariance and translational invariance can be represented as:

$$\begin{aligned}\boldsymbol{\tau} &= f_{\text{inv}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = f_{\text{inv}}(\mathbf{q}_b, \dot{\mathbf{q}}_b, \ddot{\mathbf{q}}_b, \mathbf{q}_j, \dot{\mathbf{q}}_j, \ddot{\mathbf{q}}_j) \\ &= f_{\text{inv}}(g \circ (\mathbf{q}_b, \dot{\mathbf{q}}_b, \ddot{\mathbf{q}}_b), \mathbf{q}_j, \dot{\mathbf{q}}_j, \ddot{\mathbf{q}}_j), \text{ for all } g \in \{G_\theta, T_p\}\end{aligned} \tag{3}$$

C. Rotational Equivariance in Symmetric Legged Robots

Let us assume a symmetric legged robot with n legs, which has n-th order rotational symmetry. Fig. 2 shows an example of $n = 6$. Each leg of the robot has n_l degrees of freedom and is labeled from 1 to n in a cyclic order. Then, we can split the joint configuration by leg groups: $\mathbf{q}_j = (\mathbf{q}_{l1}, \ldots, \mathbf{q}_{ln})$ where $\mathbf{q}_{li} \in \mathbb{R}^{n_l}$ represents i-th leg joint configuration. Similarly, the actuated joint torques can be divided into: $\boldsymbol{\tau}_a = (\boldsymbol{\tau}_{l1}, \ldots, \boldsymbol{\tau}_{ln})$. Now we define the leg circular shift for the joint configuration and torque as P_θ, which shifts the joint values(e.g. $\mathbf{q}_{li}, \dot{\mathbf{q}}_{li}, \ddot{\mathbf{q}}_{li}, \boldsymbol{\tau}_{li}$) assigned to each leg to the neighbouring leg as described in Fig. 2:

$$P_\theta \circ \mathbf{q}_j = P_\theta \circ (\mathbf{q}_{l1}, \mathbf{q}_{l2}, \ldots, \mathbf{q}_{ln}) = (\mathbf{q}_{l2}, \mathbf{q}_{l3}, \ldots, \mathbf{q}_{l1})$$
$$P_\theta \circ \boldsymbol{\tau}_a = P_\theta \circ (\boldsymbol{\tau}_{l1}, \boldsymbol{\tau}_{l2}, \ldots, \boldsymbol{\tau}_{ln}) = (\boldsymbol{\tau}_{l2}, \boldsymbol{\tau}_{l3}, \ldots, \boldsymbol{\tau}_{l1})$$

and by denoting $(P_\theta \circ \mathbf{q}_j, P_\theta \circ \dot{\mathbf{q}}_j, P_\theta \circ \ddot{\mathbf{q}}_j) = P_\theta \circ (\mathbf{q}_j, \dot{\mathbf{q}}_j, \ddot{\mathbf{q}}_j)$ and the rotation of the base configuration as $R_\theta \circ (\mathbf{q}_b, \dot{\mathbf{q}}_b, \ddot{\mathbf{q}}_b)$, it is clear that inverse dynamics of a robot with circular symmetry satisfies:

$$\begin{aligned}P_\theta \circ \boldsymbol{\tau} &= P_\theta \circ f_{\text{inv}}(\mathbf{q}_b, \dot{\mathbf{q}}_b, \ddot{\mathbf{q}}_b, \mathbf{q}_j, \dot{\mathbf{q}}_j, \ddot{\mathbf{q}}_j) \\ &= f_{\text{inv}}(R_\theta \circ (\mathbf{q}_b, \dot{\mathbf{q}}_b, \ddot{\mathbf{q}}_b), P_\theta \circ (\mathbf{q}_j, \dot{\mathbf{q}}_j, \ddot{\mathbf{q}}_j))\end{aligned} \tag{4}$$

Unfortunately, though it seems there exists a kind of equivariance in symmetric legged robot dynamics, Eqn. (4) shows that the current data representation cannot satisfy the definition of G-equivariance exactly, which is $f(g \circ x) = g \circ (f(x))$, due to $\mathbf{q}_b$ in the input space. We introduce a new representation model for the symmetrical-legged robot dynamics system to address this issue in the next section.

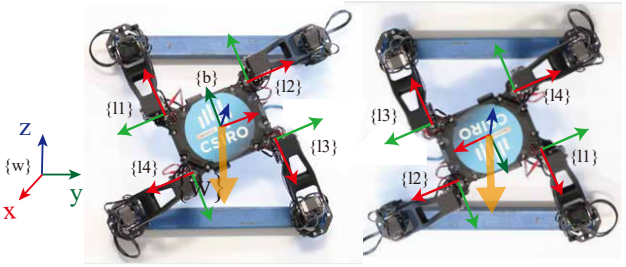

Fig. 3: Base link state representation in each leg frame. Given that the leg frames are also in symmetry for the symmetric configuration, the quantities expressed from the leg frames are invariant to the rotation of base link, allowing us to define equivariance over the rotation group.

D. Robot Representation Model

The main idea of this representation is first, to configure the domain $(\mathbf{q}_b, \mathbf{q}_j)$ and codomain $(\boldsymbol{\tau}_a)$ of the model as a set of leg configurations $(\{X_{li}\}_{i \in \text{leg}})$ and second, the fact that, for the given twist, its body-frame representation does not depend on the choice of the fixed frame [34]. For this purpose, we propose to express the information regarding base configuration from the leg coordinates and put the transformed information into each leg-group. To utilize the invariance and equivariance properties of dynamics addressed in section III-B and III-C without loss of information, we chose linear acceleration, angular velocity and gravity vector $(\boldsymbol{\omega}_b, \mathbf{a}_b, \mathbf{g}_b)$ to represent base configuration information. Then, we have transformed values:

$$\boldsymbol{\omega}_i = R_{ib}\boldsymbol{\omega}_b, \quad \mathbf{a}_i = R_{ib}\mathbf{a}_b, \quad \mathbf{g}_i = R_{ib}\mathbf{g}_b \tag{5}$$

where $\boldsymbol{\omega}_i, \boldsymbol{\omega}_b, \mathbf{a}_i$ and $\mathbf{a}_b$ represent angular velocities and linear accelerations of the base link expressed in i-th leg frame $\{li\}$ and base frame $\{b\}$, respectively. $\mathbf{g}_i, \mathbf{g}_b$ represent gravity vector expressed from the frame $\{li\}, \{b\}$, and R_{ib} represents rotation matrix that maps a vector from $\{b\}$ to $\{li\}$. We locate the leg frame $\{li\}$ at the first joint of the i-th leg at zero configuration as described in Fig. 3, which is attached to the base frame.

Instead of using (x, y, z, rz, ry, rx) to represent base configurations, by adding contact information, we can derive these information from the joint states of each leg for floating-base robots. Also we can benefit from taking values of IMU sensors in real robots for the above choice of data. In addition to $(\boldsymbol{\omega}_i, \mathbf{a}_i, \mathbf{g}_i)$, we added $\mathbf{p}_i$, the position vector from the i-th leg frame to the base link expressed in $\{li\}$ to identify the leg, and contact/adhesion Booleans b_c, b_m. Then, we can represent the robot states via leg-group data structure:

$$\begin{aligned}&(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) \triangleq \mathbf{x} = (\mathbf{x}_{l1}, \mathbf{x}_{l2}, \ldots, \mathbf{x}_{ln}) \\ &\text{where } \mathbf{x}_{li} = [\mathbf{q}_l^\top, \dot{\mathbf{q}}_l^\top, \ddot{\mathbf{q}}_l^\top, \boldsymbol{\omega}^\top, \mathbf{a}^\top, \mathbf{g}^\top, \mathbf{p}^\top, b_c, b_m]_i^\top \\ &\text{or } \mathbf{x}_{li} = [\mathbf{q}_l^\top, \dot{\mathbf{q}}_l^\top, \mathbf{q}_{l(d)}^\top, \dot{\mathbf{q}}_{l(d)}^\top, \boldsymbol{\omega}^\top, \mathbf{a}^\top, \mathbf{g}^\top, \mathbf{p}^\top, b_c, b_m]_i^\top\end{aligned} \tag{6}$$

where $\mathbf{q}_{li}, \dot{\mathbf{q}}_{li}, \ddot{\mathbf{q}}_{li}$ represent the joint states vectors of the i-th leg. Note that the next step desired states $\mathbf{q}_{l(d)}, \dot{\mathbf{q}}_{l(d)}$ can be used instead of $\ddot{\mathbf{q}}_{ji}$ in the discrete time system. Then we can rewrite the inverse dynamics as:

$$\begin{aligned}\boldsymbol{\tau} &= (\boldsymbol{\tau}_{l1}, \ldots, \boldsymbol{\tau}_{ln}) \\ &= f_{\text{inv}}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = f_{\text{inv}}(\mathbf{x}_{l1}, \mathbf{x}_{l2}, \ldots, \mathbf{x}_{ln}) = f_{\text{inv}}(\mathbf{x}) \end{aligned} \quad (7)$$

Now we show that this data representation satisfies invariance and equivariance properties discussed in section III-B and III-C.

Gravity-axis rotational invariance (G_θ) By representing all the base configuration quantities $(\mathbf{g}, \boldsymbol{\omega}, \mathbf{a}, \mathbf{p})_i$ from the frame attached to the base link, the data representation itself is invariant to G_θ. For example, $\mathbf{g}_i = R_{ib}\mathbf{g}_b$ is invariant to rotations about the gravity axis. It's derivation is as follows:

$$\mathbf{g}_b = R_{bw}\mathbf{g}_w = R_{bw}e^{[\hat{\mathbf{g}}_w]\theta}\mathbf{g}_w = R_{b'w}\mathbf{g}_w = \mathbf{g}_{b'} \quad (8)$$

where $\mathbf{g}_w = [0, 0, -9.81]^\top$ is the gravity vector expressed in the world frame, and $e^{[\hat{\mathbf{g}}_w]\theta} \in SO(3)$ represents the rotation about the axis $\hat{\mathbf{g}}_w$ by an angle θ. Suppose that $\{b'\}$ has the same origin as $\{b\}$ but it's rotated by θ about the gravity vector, we get $R_{b'w} = R_{bw}e^{[\hat{\mathbf{g}}_w]\theta}$. Finally, as vectors that are parallel to the rotation axis are not changed by the rotation, we get $e^{[\hat{\mathbf{g}}_w]\theta}\mathbf{g}_w = \mathbf{g}_w$, and thus $\mathbf{g}_i = R_{ib}\mathbf{g}_b$ is invariant to the gravity-axis rotation group with fixed R_{ib}.

Translational invariance (T_θ) Instead of taking current base link position and velocity as an input, we decided to let the model derive the linear states of the base link from the joint states of legs where its foot is in contact. As a result, our data representation can be invariant to T_θ, .

Rotational symmetry equivariance (P_θ) In the proposed representation, if the robots are in symmetry configuration, leg frames locations and orientations are also in symmetry. For example, as shown in Fig. 3, frame $\{l2\}$ of the left robot and frame $\{l4\}$ of the right are located in the same Configuration. Therefore the base link motion such as linear acceleration, angular velocity and gravity vector expressed from frame $\{l2\}$ of the left and frame $\{l4\}$ of the right will have the same values. Then, we have

$$\begin{aligned} P_\theta \circ \boldsymbol{\tau} &= f_{\text{inv}}(P_\theta \circ x) \\ \Leftrightarrow (\boldsymbol{\tau}_{l2}, \boldsymbol{\tau}_{l3}, \ldots, \boldsymbol{\tau}_{l1}) &= f_{\text{inv}}(x_{l2}, x_{l3}, \ldots, x_{l1}) \end{aligned}$$

Note that if we define the rotational symmetry equivariance (P_θ) to shift the values assigned to k-neighbouring leg instead of the right next leg, we can also describe a robot like Magneto, which has 4 symmetric legs but has 2 order of rotational symmetry, without loss of generality.

E. Symmetrical Legged Robot Inverse Dynamics Model

In this paper, we suggest two structured learning to consider symmetries in robot dynamics using GNNs and group-equivariant neural networks.

Symmetrical Dynamics GNNs (SD-GNNs) As explained in Eqn. (6), the domain of the inverse dynamics can be represented in a leg group. In a Graph Networks (GNs), data structure for learning is describes as $G = (\mathbf{u}, \{\mathbf{n}_i\}, \{\mathbf{e}_j, s_j, r_j\})$. As shown in Fig. 4, by setting the base link and the contact links as nodes and legs as edges, we can represent the domain ($\mathbf{x}$) and codomain (τ) of the inverse dynamics of legged robots as an input graph (G_{in}) and output graph (G_{out}). We then learn the mapping from states to actions using GNNs. The proposed SD-GNN is constructed based on the *encode-process-decode* architecture introduced in [12], and overall network scheme is described in Fig. 4.

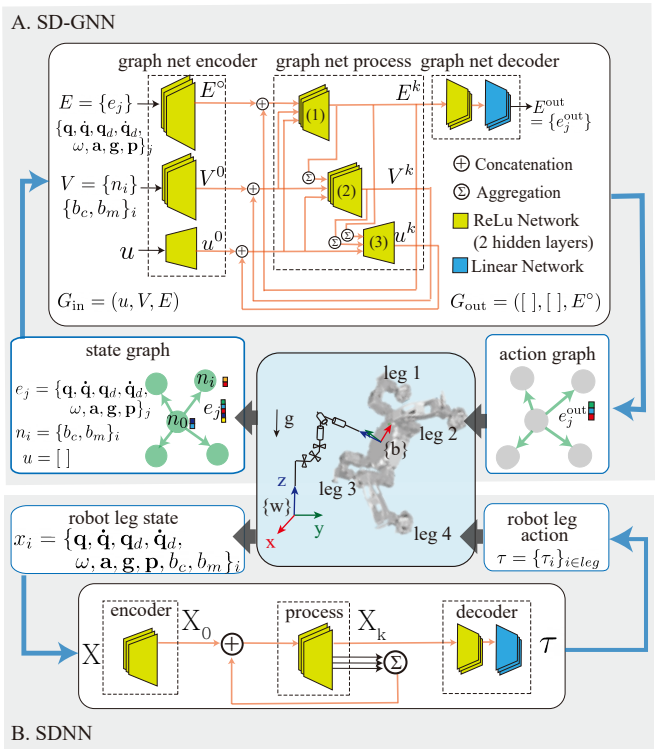

Fig. 4: A. Scheme of the inverse dynamics model developed based on SD-GNNs and B. SDNNs working on Magneto.

Symmetrical Dynamics Neural Networks (SDNNs) Though GNs provide the general form to represent robots in symmetry, there are unnecessary operations that slow the algorithm and lower the training accuracy. In this paper, we propose symmetrical dynamics neural networks (SDNNs) working on a set structure based on the permutation-equivariant layer similar to [30], [35] as described in Fig. 4. For the aggregation function, we used a summation operator, which is invariant to the permutation. By leveraging the repeated processing step with concatenating aggregated latent space of all legs, it can predict the desired output while preserving the equivariance to the rotational shift.

IV. Evaluation on a Simulator

In this section, to demonstrate the extrapolation and applicability of the proposed model, we compare the performance of the proposed SD-GNNs and SDNNs with a 3-hidden-layer FFNNs and typical GNNs described in [21] that does not consider the geometric symmetries in its data representation. We primarily focus on analyzing its generalization given a validation dataset and the model's reliability when it is used as a locomotion controller.

A. Data Generation

We have generated a set of trajectories of climbing robots in a DART simulation environment [36] using a whole body controller (WBC) [1], [2] for training and validation of the model. Climbing motions can be created by sequencing step motions where the reference swing foot trajectory is a Hermite spline derived from swing period, swing height and p_{goal}. For this experiment, we set the swing period to 0.5 seconds and sample the swing heights at around 0.05m, and goal positions at around 0.1m away from the initial foot position in the forward, backward, left, and right directions. The magnetic adhesion is deactivated and reactivated upon the initiation and completion of a swing phase.

B. Training Procedure

Each network was implemented on multi-layer perceptrons (MLPs) and ReLu activation. We used the Adam optimizer with 10^{-3} to 10^{-4} learning rates and a mini-batch size was set to 64. We trained each model until 70% of validation error does not exceed the training error more than five times in a row. The latent sizes and the number of hidden layers and processing steps of each network are set to have similar neurons for each network as follows:

	latent size	hidden layers num	processing steps num
FFNNs	256 units	3	-
Typical GNNs	global: 16 units edge: 64 units node: 16 units	2	5
SD-GNNs	global: 16 units edge: 64 units node: 16 units	2	2
SDNNs	leg: 128 units	2	2

TABLE I: Parameters for each neural network

C. Generalization Error to Unseen States

One way to show generalization ability of neural networks is to compare the validation error with respect to the training data size [14]. To demonstrate that the proposed SD-GNNs and SDNNs achieve better prediction to unseen data than other network topologies, we run the following experiments for hexa-magneto, which has 6 symmetrical legs. First, we uniformly sample input state and target action tuples to form training and validation datasets from climbing trajectories generated with different slopes and different initial base orientation as shown in Fig. 5. We then train FFNNs, typical GNNs [21], SD-GNNs, SDNNs to learn the inverse dynamics model base on the same training dataset and measure the model output error(torque) over the same test data set, which has not been used for training. Fig. 5 A and B show the generalization errors over the scenarios of gravitational invariance and rotational equivariance respectively. We can see the proposed SD-GNNs, SDNNs method greatly outperform FFNNs or typical GNNs. Fig. 5 C illustrates the sample efficiency and generalization ability of each network by showing the generalization errors with respect to the sample ratio used for training over the whole dataset. We created several training sets by picking scenarios randomly from the

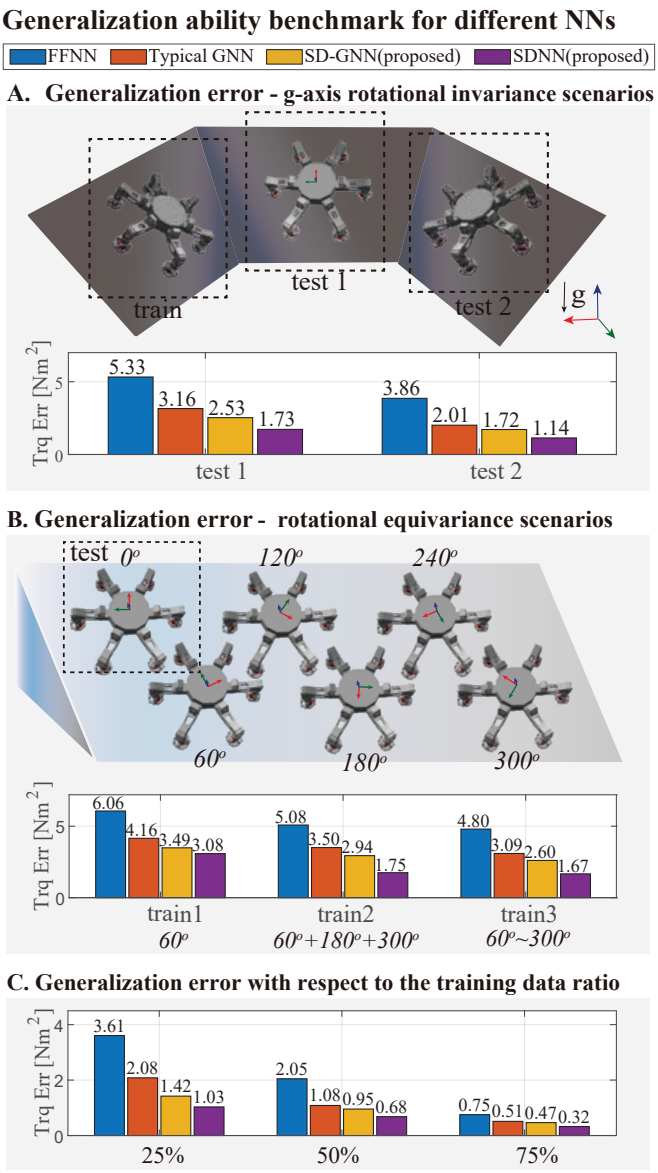

Fig. 5: subFig.A depicts robot climbing scenarios in various climbing slopes with the same inclination. subFig.B shows the performance of each networks to predict unseen data in rotational symmetry, and subFig.C shows sample efficiency.

whole dataset to train each model and measured the error of the model over the whole data. Fig. 5. C. shows that the proposed methods have lower prediction error with less training data, meaning that they yield better generalization ability and are sample efficient.

D. Generalization Performance on Various Robots

We investigate the generalization capability of FFNN and the proposed SDNN for three different legged robots, Magneto, Hexa-Magneto, and Nona-Magneto, which have 4, 6, and 9 legs, respectively. As shown in Fig. 6, SDNN consistently outperforms FFNN in predicting unseen data with fewer training data. While it may seem intuitive that SDNN would outperform FFNN more significantly, as the number of symmetrical legs in a robot increases, actual

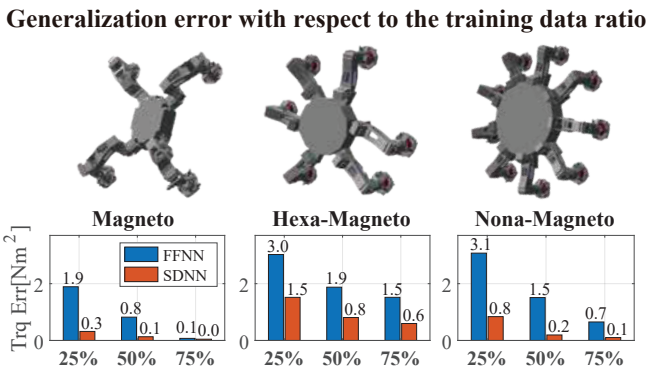

Fig. 6: Generalization capability comparison between FFNN and SDNN on robots with different number of legs.

findings show that the relative performance of SDNN over FFNN is more pronounced in Magneto than in Hexa or Nona-Magneto. This is somewhat unexpected, remaining the need for further research to verify the relation between the performance and complexity of the model.

E. Inverse Dynamics Control Performance

This section describes the implementation of two neural network inverse dynamics controllers, SDGNN-IDM(inverse dynamics model) and FFNN-IDM, as controllers to track a reference trajectory on Magneto. The controllers take the current and desired states $\mathbf{q}_d$ and $\dot{\mathbf{q}}_d$ of the robot as input states – see Fig. 4 – and directly apply $\boldsymbol{\tau}$ obtained from the model to the robot as control inputs. Even though we train both models until they converge to have a similarly small training loss, there were huge differences when they were used as a controller. As shown in Fig. 7 (a) and 7 (c), the FFNN-IDM based controller fails to track the trajectory within 50ms, while the SDGNN-IDM-based controller tracks the trajectory very well even after 10s. The FFNN controller seems to fail in tracking due to its lack of generalization ability to handle the errors accumulated over time during the simulation. A PD augmented FFNN-IDM based controller shown in Fig. 7 (b) supports this claim. It shows that PD gains, which are set small enough not to dominate the tracking performance, can improve the tracking performance by preventing accumulated errors from getting too large. Thus we can conclude that the FFNN-IDM based controller failed due to its lack of robustness. On the other hand, SDGNN-IDM works surprisingly well without any additional help of a traditional feedback controller (Fig. 7 (c)).

V. Discussion and Conclusion

In this paper, we introduce sample efficient robot dynamics learning exploiting the symmetries and physical invariance of the underlying robot. Structured data representation allows us to map the identical dynamics resulting from symmetric legs. Group-equivariant neural network architectures provide the means to incorporate identical dynamical structures in multi-legged robots. As such, it brings substantial efficiency benefits for dynamics learning and generalization capability.

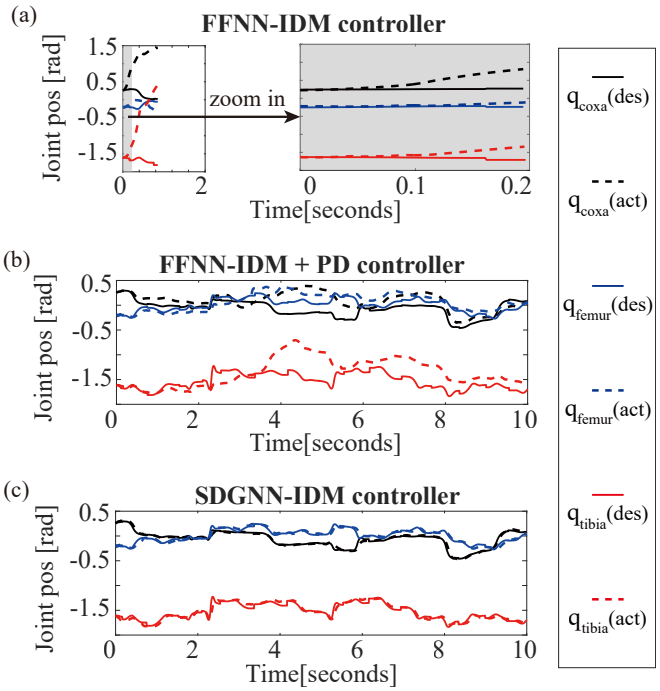

Fig. 7: Joint positions(coxa, femur, tibia joints) of one of Magneto's legs in a simulation are shown for various controllers. $\mathbf{q}$(des) and $\mathbf{q}$(act) represent the reference trajectory and the actual joint position obtained in the simulation when the control inputs derived from each model are applied.

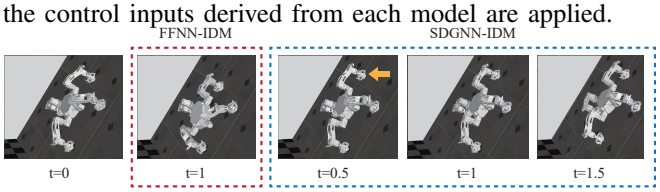

Fig. 8: Snapshots of Magneto climbing in simulation. Yellow arrows in the figures indicate the swing foot.

We have empirically validated the above claims from the following perspectives (i) the proposed SD-GNNs and SDNNs can make more accurate predictions on unseen data, and (ii) SD-GNNs-based inverse dynamics controllers are robust and accurate for trajectory tracking.

A future avenue of research will verify the applicability of the proposed symmetrical dynamics learning for robots with different geometric shapes. Though the proposed model only can capture the radial symmetry in robots, bi-symmetrical robots are also typical such as humanoids and quadrupeds in the legged robots community. We can also evaluate the model's robustness to the unseen environment by examining its performance under the noise and unknown environment properties such as friction, external force, and uneven terrain. We are also interested in knowing how the generalization of our proposed method can help learn robust control policy for data intensive tasks in reinforcement learning.

Acknowledgment

This work has been partially supported by the Office of Naval Research, ONR Grant N000141912311, and by DATA61, CSIRO.

REFERENCES

[1] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 04, pp. 505–518, 2005.

[2] D. Kim, S. J. Jorgensen, J. Lee, J. Ahn, J. Luo, and L. Sentis, "Dynamic locomotion for passive-ankle biped robots and humanoids using whole-body locomotion control," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 936–956, 2020.

[3] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous robots*, vol. 35, no. 2, pp. 161–176, 2013.

[4] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 295–302.

[5] J.-e. Lee, T. Bandyopadhyay, and L. Sentis, "Adaptive robot climbing with magnetic feet in unknown slippery structure," *Frontiers in Robotics and AI*, p. 205, 2022.

[6] J.-e. Lee and J. Park, "Kinematic parameter calibration for humanoid robot using relative pose measurement in walking motion," in *2019 16th International Conference on Ubiquitous Robots (UR)*. IEEE, 2019, pp. 712–717.

[7] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[8] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.

[9] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

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

[11] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," *arXiv preprint arXiv:2103.14295*, 2021.

[12] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," *arXiv*, pp. 1–40, 2018.

[13] M. Lutter, C. Ritter, and J. Peters, "Deep lagrangian networks: Using physics as model prior for deep learning," *arXiv preprint arXiv:1907.04490*, 2019.

[14] J. K. Gupta, K. Menda, Z. Manchester, and M. Kochenderfer, "Structured mechanical models for robot learning and control," in *Learning for Dynamics and Control*. PMLR, 2020, pp. 328–337.

[15] J. Ahn and L. Sentis, "Nested mixture of experts: Cooperative and competitive learning of hybrid dynamical system," in *Learning for Dynamics and Control*. PMLR, 2021, pp. 779–790.

[16] W. Yu, G. Turk, and C. K. Liu, "Learning symmetric and low-energy locomotion," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, pp. 1–12, 2018.

[17] F. Abdolhosseini, H. Y. Ling, Z. Xie, X. B. Peng, and M. Van de Panne, "On learning symmetric locomotion," in *Proceedings of the 12th ACM SIGGRAPH Conference on Motion, Interaction and Games*, 2019, pp. 1–10.

[18] T. Bandyopadhyay, R. Steindl, F. Talbot, N. Kottege, R. Dungavell, B. Wood, J. Barker, K. Hoehn, and A. Elfes, "Magneto: A versatile multi-limbed inspection robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2253–2260.

[19] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE transactions on neural networks*, vol. 20, no. 1, pp. 61–80, 2008.

[20] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[21] A. Sanchez-Gonzalez, N. Heess, J. T. Springenberg, J. Merel, M. Riedmiller, R. Hadsell, and P. Battaglia, "Graph networks as learnable physics engines for inference and control," *35th International Conference on Machine Learning, ICML 2018*, vol. 10, pp. 7097–7117, 2018.

[22] T. Wang, R. Liao, J. Ba, and S. Fidler, "Nervenet: Learning structured policy with graph neural networks," in *International conference on learning representations*, 2018.

[23] J. Whitman, M. Travers, and H. Choset, "Learning modular robot control policies," *arXiv preprint arXiv:2105.10049*, 2021.

[24] G.-J. Qi, L. Zhang, F. Lin, and X. Wang, "Learning generalized transformation equivariant representations via autoencoding transformations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[25] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012.

[27] Y. Lipman, O. Sorkine, D. Levin, and D. Cohen-Or, "Linear rotation-invariant coordinates for meshes," *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3, pp. 479–487, 2005.

[28] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical cnns," *arXiv preprint arXiv:1801.10130*, 2018.

[29] R. Gens and P. M. Domingos, "Deep symmetry networks," *Advances in neural information processing systems*, vol. 27, 2014.

[30] S. Ravanbakhsh, J. Schneider, and B. Poczos, "Deep learning with sets and point clouds," *arXiv preprint arXiv:1611.04500*, 2016.

[31] Y. Tang and D. Ha, "The sensory neuron as a transformer: Permutation-invariant neural networks for reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 22 574–22 587, 2021.

[32] J. Zhou, G. Cui, Z. Zhang, C. Yang, Z. Liu, L. Wang, C. Li, and M. Sun, "Graph Neural Networks: A Review of Methods and Applications," *arXiv*, pp. 1–22, 2018.

[33] H. Maron, H. Ben-Hamu, N. Shamir, and Y. Lipman, "Invariant and equivariant graph networks," *arXiv preprint arXiv:1812.09902*, 2018.

[34] K. M. Lynch and F. C. Park, *Modern robotics*. Cambridge University Press, 2017.

[35] T. Cohen and M. Welling, "Group equivariant convolutional networks," in *International conference on machine learning*. PMLR, 2016, pp. 2990–2999.

[36] J. Lee, M. X. Grey, S. Ha, T. Kunz, S. Jain, Y. Ye, S. S. Srinivasa, M. Stilman, and C. K. Liu, "Dart: Dynamic animation and robotics toolkit," *Journal of Open Source Software*, vol. 3, no. 22, p. 500, 2018.