

# Proprioceptive Image: An Image Representation of Proprioceptive Data from Quadruped Robots for Contact Estimation Learning

Gabriel Fischer Abati<sup>1,2\*</sup>, João Carlos Virgolino Soares<sup>1</sup>, Giulio Turrisi<sup>1</sup>, Victor Barasuol<sup>1</sup>, Claudio Semini<sup>1</sup>

**Abstract**—This paper presents a novel approach for representing proprioceptive time-series data from quadruped robots as structured two-dimensional images, enabling the use of convolutional neural networks for learning locomotion-related tasks. The proposed method encodes temporal dynamics from multiple proprioceptive signals, such as joint positions, IMU readings, and foot velocities, while preserving the robot’s morphological structure in the spatial arrangement of the image. This transformation captures inter-signal correlations and gait-dependent patterns, providing a richer feature space than direct time-series processing. We apply this concept in the problem of contact estimation, a key capability for stable and adaptive locomotion on diverse terrains. Experimental evaluations on both real-world datasets and simulated environments show that our image-based representation consistently enhances prediction accuracy and generalization over conventional sequence-based models, underscoring the potential of cross-modal encoding strategies for robotic state learning. Our method achieves superior performance on the contact dataset, improving contact state accuracy from 87.7% to 94.5% over the recently proposed MI-HGNN method, using a 15 times shorter window size.

## I. INTRODUCTION

Deep learning has been highly successful when sequential or high-dimensional signals are converted into visual representations that suit convolutional architectures. In speech recognition, raw audio is commonly transformed into spectrograms, i.e., 2D time–frequency images, enabling CNN-based models to achieve state-of-the-art results, from keyword spotting to end-to-end transcription [1], [2], [3]. Similarly, in cybersecurity, software files and their execution behavior have been mapped to grayscale or RGB images, where CNNs exploit visual patterns for malware detection [4]. These data-to-image mappings leverage the spatial feature learning strengths of image-based deep networks.

Motivated by these successes, we transform quadruped proprioceptive time-series signals into 2D images that encode both temporal dynamics and robot morphology. Signals are arranged according to the robot’s physical layout, with time incorporated as an additional spatial dimension, preserving inter-signal correlations and gait-dependent structure in a CNN-friendly format. This yields a richer and more spatially coherent feature space than learning directly from raw time

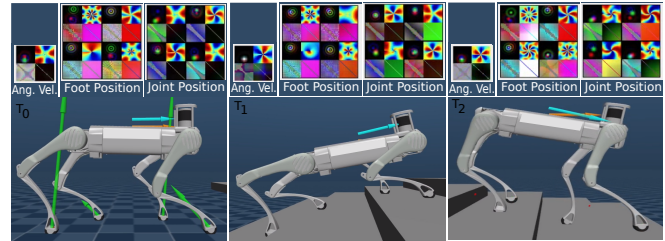


Fig. 1: Examples of proprioceptive images over three time steps of a quadruped robot walking on uneven terrain. Each image encodes angular velocity, joint position, and foot position, illustrating how the representation varies across environments and supports learning of tasks such as foot contact estimation.

series, and we demonstrate its effectiveness on foot contact estimation, a key component for stable and adaptive locomotion [5], [6].

Figure 1 shows three time steps of a quadruped traversing uneven terrain, along with the corresponding proprioceptive images derived from IMU base angular velocity, encoder joint positions, and estimated foot positions. The resulting patterns vary across environments while providing a compact representation that supports learning tasks such as contact estimation.

### A. Related Work - 1D time series to image conversion

To enable CNN-based classification on time-series data, Wang and Oates [7] proposed the Gramian Angular Field (GAF) and Markov Transition Field (MTF) encodings. GAF constructs a Gramian matrix [8] whose entries capture trigonometric relations between time indices, yielding a bijective mapping that preserves temporal correlations and supports approximate reconstruction.

In contrast, MTF represents temporal dynamics through transition statistics: the signal is discretized into quantile bins and a time-indexed transition matrix is formed [7]. Although MTF captures multi-span transition probabilities (including self-transitions along the diagonal), it is sensitive to the binning choice, which complicates its use with heterogeneous modalities. Moreover, both MTF and GAF exhibit rate invariance, producing identical images for increasing and decreasing sequences with the same rate.

Recurrence Plots (RP) [9] encode temporal structure by marking state recurrences, supporting the diagnosis of drift,

This work was supported by the European Union – NextGenerationEU, the PNRR MUR Project PE000013 “Future Artificial Intelligence Research (FAIR)”.

<sup>1</sup> Dynamic Legged Systems (DLS) lab, Istituto Italiano di Tecnologia (IIT), Genova, Italy.

<sup>2</sup> University of Genoa, Genoa, Italy

\* Corresponding author, gabriel.fischer@iit.it

periodicity, and chaotic dynamics, but they are less responsive to subtle variations.

Overall, converting 1D time series into 2D encodings (GAF, MTF, RP) enables spatial feature learning with CNNs and has shown strong performance across benchmarks [10], [11], [12]. Performance can be further improved via multimodal fusion (meta-features, attention, or concatenation), which captures complementary static, dynamic, and recurrence cues, with applications ranging from manufacturing defect detection to heartbeat classification [13], [14], [15]. As such, fused image encodings paired with tailored CNN/attention architectures remain a practical and effective strategy for time-series classification [16], [17].

In this work, we bring this paradigm to quadruped robotics by converting proprioceptive time-series signals into structured 2D representations informed by robot morphology, enabling CNN-based feature extraction for tasks such as foot contact estimation.

### B. Related Work - Contact Estimation

Contact estimation for quadruped robots is a fundamental perception problem, since contact measurements are not always available [6]. Yet it remains challenging due to sensor noise, terrain variability, and effects such as slippage.

A common baseline estimates contact by thresholding measured or estimated ground reaction forces (GRF) [18], but it often fails to generalize across robots and environments. To improve robustness, several works combine probabilistic modeling, sensor fusion, and system dynamics. Camurri et al. [5] estimate contact probabilistically using GRF computed from joint positions, velocities, and efforts. Menner and Berntorp [19], instead, cast locomotion as a switched system, where each mode corresponds to a different set of feet in contact. Maravgakis et al. [20] use Kernel Density Estimation (KDE) over IMU batches to infer the probability of stable 6-DoF contact, while the Dual  $\beta$ -Kalman Filter [21] explicitly addresses slippage and variable leg length.

Despite these advances, model-based and thresholding approaches remain noise-sensitive and typically require empirical tuning, limiting robustness on lightweight robots and changing terrains. Consequently, recent work has shifted toward data-driven techniques. Youm et al. [22] learn a contact-probability measurement model via a Neural Measurement Network to improve IEKF performance under slip, and Lee et al. [23] combine probabilistic torque/dynamics-based estimation with a neural compensator for residual errors. Beyond these hybrid strategies, fully learning-based methods have also emerged. For example, Lin et al. [24] propose Contact-CNN, a multi-modal network that relies only on proprioceptive signals (joint encoders, IMU, kinematics) to classify foot-ground contact across diverse gaits and terrains, removing the need for explicit contact sensing.

Ordóñez-Præez et al. [25] introduced Discrete Morphological Symmetries (DMSs), a group-theoretic symmetry groups of dynamical systems. Building on this, they proposed ECNN, a variant of Contact-CNN that applies sparse

equivariant convolutions and batched augmentation to enforce hard equivariance, making large architectures tractable and improving leg-level F1 and test loss. Also, MI-HGNN [26] introduces a morphology-informed heterogeneous graph neural network (GNN) that constructs a graph from a robot’s kinematic structure (nodes being the base/joints/feet, and edges representing links) to constrain learning for contact perception problems. Empirically, MI-HGNN attains substantially higher contact-detection accuracy (mean 87.7% vs 78.8% from ECNN [25]) on the Contact Data Set [24].

Although recent data-driven methods [24], [25], [26] have achieved promising results, they still rely on raw proprioceptive data streams as input. In contrast, our approach introduces a novel representation to create unique image patterns to describe quadruped robot sensing. To the best of our knowledge, this is the first method in robotics that converts proprioceptive signals into image representation.

### C. Contributions

The contributions of the paper can be summarized as follows:

- A novel representation of proprioceptive signals into 2D images, with the name Proprioceptive image (PI). The core objective of the PI representation is to encode both temporal dynamics and morphological characteristics of quadruped robots. This image representation can be explored in learned-based approaches to perform complex tasks in robotics.
- We propose a CNN architecture that is able to learn from PI information and perform contact estimation for quadruped robots.
- We show the results of the PI trained models in real-world datasets and in simulated environments, and compared with state-of-the-art methods. In the contact dataset from [24], our method improved contact state accuracy from 87.7% to 94.5% over the recently proposed MI-HGNN method [26], using a 15 times shorter window size.

## II. METHODOLOGY

In this section, we present the four key components of the PI formulation and explain how they are fused to construct the final PI representation, which is tailored to the robot’s morphology. Each component captures essential temporal and statistical characteristics of a single proprioceptive state and encodes them into a sub-image through well-defined mathematical transformations. Specifically, it is extracted and encoded into image form using the following elements from the signal data stream: the slope dynamics in Section II-B, spike patterns in Section II-C, modified GAF polar distance in Section II-D, and the global-aware local shift in Section II-E. Figure 2 shows how to generate the Proprioceptive Image from proprioceptive signal data.

### A. Preliminaries

**Input Values:** Let  $s = \{s_i\}_{i=1}^T$  be a discrete 1D time series sampled from a proprioceptive signal (estimated or

measured by a sensor). At each timestamp  $t$ , we define a sliding window of length  $w$  as  $x_t = [s_{t-w+1}, \dots, s_t] \in \mathbb{R}^w$ . The goal is to convert each key component into a  $w \times w$  image and concatenate them to form a  $2w \times 2w$  image.

**Global Bounds:** The PI's representation uses as reference the global bounds of the input signal. The global bounds are defined as the maximum and minimum values of the proprioceptive signals outputs  $\mathcal{G} = [s_{min}^{global}, s_{max}^{global}]$  (usually obtained from the sensor datasheet).  $\mu_{\mathcal{G}}$  is defined as the average of the global bounds  $\mu_{\mathcal{G}} = (s_{max}^{global} + s_{min}^{global})/2$ . The normalized sliding window sequence, computed with respect to the global bounds, is defined as  $x_t^{\mathcal{G}} = [s_{t-w+1}^{\mathcal{G}}, \dots, s_t^{\mathcal{G}}] \in \mathbb{R}^w$ , where each  $s_i^{\mathcal{G}} \in [-1, 1]$  represents the globally normalized signal value at time  $i$  computed as:

$$s_i^{\mathcal{G}} = \frac{(s_i - s_{max}^{global}) + (s_i - s_{min}^{global})}{(s_{max}^{global} - s_{min}^{global})} \quad (1)$$

**Local Range:** For each proprioceptive signal of interest, we compute a pair of normalized deviation measures  $\delta$ . These values quantify how far the current local operating range  $\mathcal{Y} = [s_{min}^{local}, s_{max}^{local}]$  of the signal is shifted relative to its global reference range. The local range is estimated from a sliding window by taking the lower and upper percentiles (10th and 90th) and expanding them by a small margin to ensure robustness to noise. Given the global average  $\mu_{\mathcal{G}}$  and global bounds  $\mathcal{G}$ , the normalized deviation  $\delta = [\delta_{min}, \delta_{max}]$  is defined as:

$$\delta_{min} = \frac{s_{min}^{local} - \mu_{\mathcal{G}}}{|\mu_{\mathcal{G}} - s_{min}^{global}|} \quad \delta_{max} = \frac{s_{max}^{local} - \mu_{\mathcal{G}}}{|\mu_{\mathcal{G}} - s_{max}^{global}|} \quad (2)$$

**Quadruped Robot Morphology:** The quadruped robot consists of a rigid trunk and four identical legs — left-front (LF), right-front (RF), left-hind (LH), and right-hind (RH). Each leg has three actuated joints: hip abduction/adduction (HAA), hip flexion/extension (HFE), and knee flexion/extension (KFE). Each joint provides proprioceptive feedback, including torque (or torque estimates from motor currents), angular position, and velocity. The trunk houses an inertial measurement unit (IMU) for measuring accelerations, angular velocities, and estimating body orientation.

**PI Concatenation:** The primary objective of each PI is to capture key dynamics of a data stream and encode them into a compact image representation. To also incorporate the robot's morphology, individual PIS are concatenated both channel-wise and spatially to form a final representation for a given proprioceptive signal type (e.g., torque, joint position, joint velocity). PIS are grouped into leg PIS and trunk PIS. Leg signals include per-leg proprioceptive signals such as joint positions, joint velocities, foot positions, foot velocities, and GRFs. Trunk signals, such as the data from IMUs, provide measurements of the trunk's motion in space (angular velocity and linear acceleration), which serve as global indicators of the robot's state. For example, the torque PI for a single leg  $PI^{Leg} \in \mathbb{R}^{2w \times 2w \times 3}$  is formed by concatenating channel-wise the PIS of its three joints (or three linear dimensions in the case of foot positions and foot

velocities):

$$PI_{Torque}^{Leg} = [(PI_{Torque}^{HAA}, PI_{Torque}^{HFE}, PI_{Torque}^{KFE})] \quad (3)$$

and the full-body torque PI is obtained by arranging the per-leg torque PIS spatially:

$$PI_{Torque} = \begin{bmatrix} PI_{Torque}^{LF} & PI_{Torque}^{RF} \\ PI_{Torque}^{LH} & PI_{Torque}^{RH} \end{bmatrix} \in \mathbb{R}^{4w \times 4w \times 3} \quad (4)$$

Trunk PIS are constructed by concatenating the components of each dimension channel-wise. The concatenation technique is used in all key components of the PI except the Global-Aware Local Shift.

## B. Slope Dynamics

Given the temporal window of scalar values  $x_t$ , we encode three dynamical descriptors — mean slope, jerk, and peak density — into a spatial Gaussian blob modulated by concentric ripples. More precisely, we fit a linear model  $x_i = \beta_t i + b$  over the window  $w$  to compute the slope  $\beta_t \in \mathbb{R}$  using least-squares and clipped to be  $[-1, 1]$ :

$$\hat{\beta}_t = \text{clip}\left(\frac{\beta_t}{\sigma(x) + \epsilon}, -1, 1\right) \quad (5)$$

where  $\sigma(\cdot)$  is the standard deviation of the window, and  $\epsilon$  ensures numerical stability. The normalized slope  $\hat{\beta}$  determines the horizontal displacement of the blob. The jerk is computed as the absolute value of the second time-derivative,

$$\mathcal{J} = \frac{1}{w-2} \sum_{t=0}^{w-3} |(x_{t+2} - x_{t+1}) - (x_{t+1} - x_t)| \quad (6)$$

and normalized as  $\hat{\mathcal{J}} = \text{clip}\left(\frac{\mathcal{J}}{\sigma(\hat{x}) + \epsilon}, 0, 1\right)$  by the velocity deviation  $\sigma(\hat{x})$ , which defines the vertical displacement of the blob. The peak density is defined as the ratio between the number of local maxima and the window size, scaled to  $[0, 1]$ :

$$\text{peak} = \text{clip}\left(\frac{\#peaks}{w} \cdot 5, 0, 1\right) \quad (7)$$

which controls the ripple frequency  $f_{ripple} = 2 + \lfloor 6 \cdot \text{peak} \rfloor$ . These normalized features are mapped to the center of a Gaussian blob, whose coordinates  $c_x$  and  $c_y$  are given by:

$$c_x = \lfloor \frac{\hat{\beta} + 1}{2}(w-1) \rfloor \quad c_y = \lfloor (1 - \hat{\mathcal{J}})(w-1) \rfloor \quad (8)$$

Let  $R_{i,j} = \sqrt{(i - c_x)^2 + (j - c_y)^2}$  be the radial distance from the blob center, the base Gaussian is computed as:

$$G_{i,j} = \exp\left(-\frac{R_{i,j}^2}{2\sigma_g^2}\right) \quad (9)$$

with Gaussian spread of  $\sigma_g = w/6.25$ . To introduce the ripple texture, we apply a cosine modulation:

$$\rho_{i,j} = 0.5 + 0.5 \cos\left(\frac{f_{ripple} R_{i,j}}{\sigma_g}\right) \quad (10)$$

and form the final pattern  $A = G \odot \rho$ , normalized to the interval  $[0, 1]$  and scaled to 8-bit intensity. In this representation, the horizontal position of the blob encodes

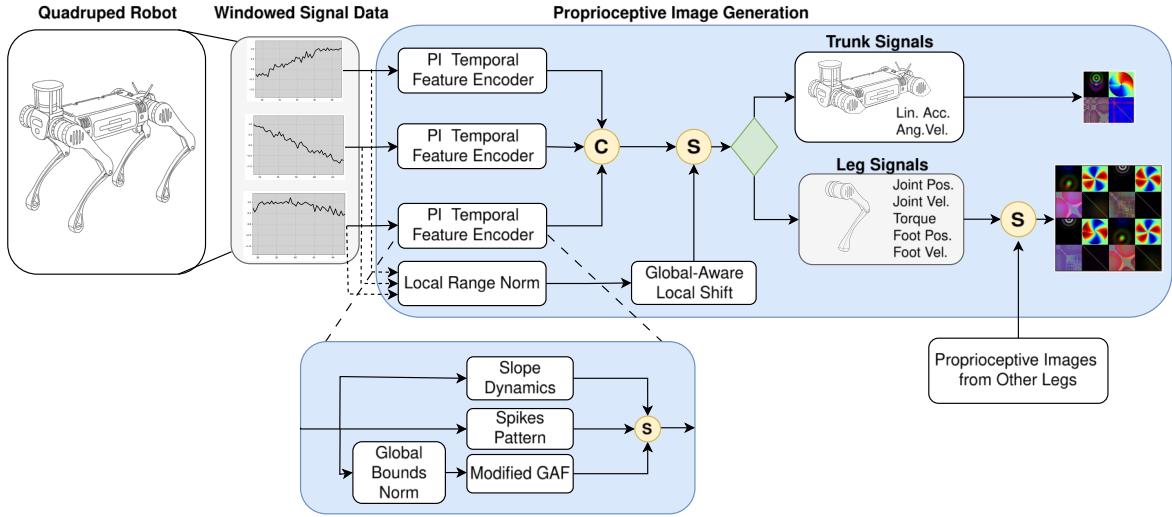


Fig. 2: Pipeline for constructing a Proprioceptive Image. Proprioceptive signal streams are grouped in triplets (X, Y, Z for trunk signals; HAA, HFE, KFE for leg signals) to form three-channel RGB inputs. Each windowed signal is processed by a temporal feature encoder that generates image-based encodings of slope dynamics, spike patterns, and GAF. In parallel, the local range of each signal triplet is used to create the final PI sub-image, named Global-Aware Local Shift Image, which is spatially concatenated with the PI Temporal Feature Encoder into a square image. Leg PIs are further arranged in a  $\begin{bmatrix} LF & RF \\ LH & RH \end{bmatrix}$  layout to represent all four legs, while trunk PIs are used individually. Yellow circles labeled ‘C’ indicate channel concatenation, and circles labeled ‘S’ denote spatial concatenation.

trend direction (slope), the vertical position encodes motion irregularity (jerk), and the ripple density reflects oscillatory content (peak density), providing a compact visual signature of local signal dynamics. Figure 3 presents three example images generated from different input signals.

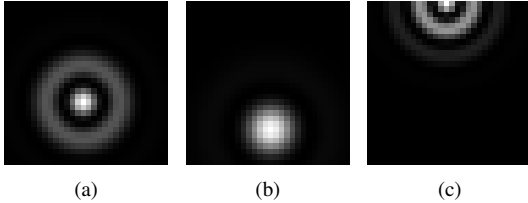


Fig. 3: Examples of Slope Dynamics images from simulated joint position signal of a quadruped robot trotting.

### C. Spikes Patterns

Given a temporal window  $x_t$  from the proprioceptive sensor, we form the pairwise difference matrix  $\Delta_{i,j} = x_j - x_i \in \mathbb{R}^{w \times w}$  and suppress noise via an adaptive gating threshold based on robust variability. Specifically, we compute the median  $m_x = \text{median}(x)$  and the median absolute deviation (MAD)

$$MAD_x = \text{median}(|x_t - m_x|) + \epsilon \quad (11)$$

and set the gating threshold as  $th_{MAD} = \alpha MAD_x$ , where  $\alpha > 0$  controls the sensitivity (smaller values detect more subtle changes, and larger values restrict detection to stronger transitions). A binary gate

$$\mathcal{M}_{i,j} = \mathbb{I}(|\Delta_{i,j}| > th_{MAD_x}) \quad (12)$$

selects only pairs exceeding this adaptive threshold. For the masked upper-triangular entries ( $i < j, \mathcal{M}_{i,j} = 1$ ), we compute their median  $m_\Delta$  and  $MAD_\Delta$  to form the following robust score

$$Z_{i,j} = \frac{\Delta_{i,j} - m_\Delta}{MAD_\Delta} \quad (13)$$

Scores are symmetrically clipped to  $[-3,3]$ , following the three-sigma principle [27] to limit extreme outliers while maintaining contrast for typical values. The final spike image  $I \in [0, 255]^{w \times w}$  is obtained as

$$C_{i,j} = \begin{cases} 128 + 127 \frac{\text{clip}(Z_{i,j}, -3, 3)}{3} & \mathcal{M}_{i,j} = 1 \\ 128 & \mathcal{M}_{i,j} = 0 \end{cases} \quad (14)$$

where bright pixels ( $\approx 255$ ) represent strong positive spikes, dark pixels ( $\approx 0$ ) strong negative spikes, and mid-gray ( $\approx 128$ ) either neutral changes or subthreshold differences. This adaptive MAD-based construction yields a symmetric, noise-resilient spike matrix that adjusts to local variability while preserving sensitivity to significant temporal transitions. Figure 4 illustrates three examples of proprioceptive images obtained from distinct input sequences.

### D. Polar Distance

In this sub-image, we employ a modified version of the GAF representation. GAF maps the input signal to an angular coordinate  $\phi_t = \arccos(x_t^G)$  and its matrix entries are defined as  $GAF = \cos(\phi_i + \phi_j)$ ,  $(i, j) = 1, \dots, w$ . While the standard GAF inherently captures static temporal information through its unique polar encoding, it suffers from ambiguity in distinguishing between increasing and decreasing trends. To overcome this limitation, we replace

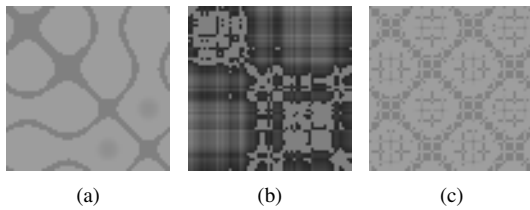


Fig. 4: Examples of Spikes pattern images from simulated joint position of a quadruped robot trotting.

the values along the main diagonal of the GAF matrix with the globally normalized sliding window sequence  $x_t^G$ , subsequently scaled to the pixel intensity range  $[0, 255]$ . This substitution embeds directional information directly into the matrix diagonal values, enhancing the interpretability of the encoded temporal dynamics. Figure 5 shows three representative proprioceptive images corresponding to different input conditions. Eq. 15 describes the GAF function:

$$D = \begin{bmatrix} x_1^G & \cos(\phi_1 + \phi_2) & \dots & \cos(\phi_1 + \phi_w) \\ \cos(\phi_2 + \phi_1) & x_2^G & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \cos(\phi_w + \phi_1) & \dots & \dots & x_w^G \end{bmatrix} \quad (15)$$

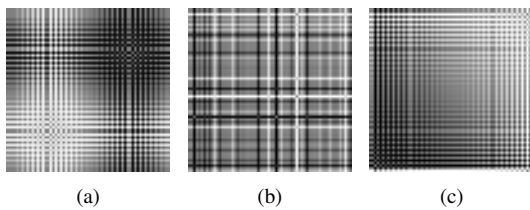


Fig. 5: Example of GAF images from simulated joint position signal of a quadruped robot trotting.

### E. Global-Aware Local Shift

We propose the cymatic encoding, inspired by physical vibration patterns observed on resonating plates and membranes, where standing wave interference produces stable, symmetric geometric patterns that depend on the excitation frequencies and phases. Such patterns are a natural 2D analog of harmonic decomposition in 1D signals: radial and angular wave modes act like spatial Fourier components, while their interference encodes complex multi-frequency relationships in a compact visual form.

This sub-image is the only one that does not follow the PI concatenation method as the other three sub-images. Instead of inputting the local temporal window and concatenating the PI images for each joint/axis, we input the normalized deviations  $\delta$  of three joint/axis proprioceptive signals as a 6-dimensional descriptor vector that controls the spatial pattern parameters. The first four components set the radial frequency  $k_r \in (1, 4)$ , angular frequency  $k_t \in (1, 4)$ , and

their respective phases  $\phi_r, \phi_t \in [0, 2\pi)$ . These determine the number of concentric rings and azimuthal lobes, directly modulating spatial symmetry. The fifth component sets a modulation scale  $\alpha \in [0, 1]$ , controlling the amplitude of a secondary interference field, while the sixth is a blend weight,  $m \in [0, 1]$  determining how strongly the primary and secondary fields combine.

The 2D spatial domain  $(X, Y) \in [-1, 1]^2$  is converted to polar coordinates

$$\mathcal{R} = \text{clip}(\sqrt{X^2 + Y^2}, 0, 1) \quad T = \text{atan2}(Y, X) \quad (16)$$

and two wave-interference components are defined as  $\mathcal{C}_1 = \sin(k_r \mathcal{R} + \phi_r) \cos(k_t T + \phi_t)$  and  $\mathcal{C}_2 = \sin(k_r \mathcal{R} - k_t T)$ .

The final cymatic field is a convex combination given as

$$\mathcal{C} = (1 - m)\mathcal{C}_1 + m\alpha\mathcal{C}_2 \quad (17)$$

masked outside the unit disk and normalized to  $[0, 255]$  for image representation. This mapping is special since it translates a low-dimensional descriptor into a high-complexity, structured, and interpretable 2D pattern, whose symmetry, periodicity, and interference texture vary smoothly with its parameters. Unlike arbitrary texture generators, cymatic patterns are rooted in physical wave phenomena, yielding repeatable geometric signatures that are both parameter-sensitive and visually distinct. This makes them particularly well-suited for encoding multi-feature signal descriptors into images for downstream convolutional processing, as small variations in the descriptor lead to smooth but discriminative changes in the spatial pattern. Figure 6 depicts three sample images illustrating how varying inputs produce distinct representations.

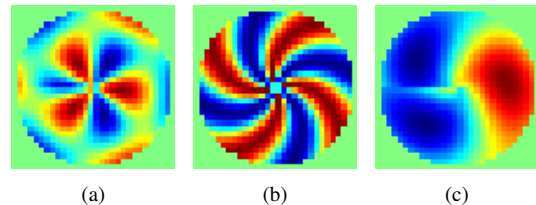


Fig. 6: Examples of Global-Aware Local Shift images from simulated joint position signal of a quadruped robot trotting.

### F. Contact Estimation

Following [24], contact estimation is framed as a multi-class classification problem, where each binary foot-ground configuration corresponds to a discrete class. To address this, we propose ConcatCNN, a multi-image convolutional architecture that jointly encodes and fuses proprioceptive image representations. The input modalities are joint positions (rad), joint velocities (rad/s), base linear acceleration ( $m/s^2$ ), base angular velocity (rad/s), foot positions (m), and foot velocities (m/s), all expressed in the base frame when applicable. Each input image is processed by a shared CNN encoder with two convolutional layers (kernel size  $3 \times 3$ , padding 1), each followed by Group Normalization

(GN) [28] and LReLU activation [29], with a  $2 \times 2$  max-pooling operation for downsampling. LReLU mitigates the risk of dead neurons in sparse proprioceptive inputs, while ReLU activations are retained in the fully connected layers for stability and efficiency. GN was chosen over Batch Normalization for its robustness to small batch sizes and heterogeneous signals streams, ensuring consistent feature scaling. The shared encoder outputs feature maps of size  $16 \times 5 \times 5$  per image, which are concatenated into a fused representation of size  $16N \times 5 \times 5$ , where  $N$  denotes the number of PIs provided as input. This representation is refined by a convolutional fusion layer (kernel size  $3 \times 3$ , 32 channels), flattened, and passed through a two-layer fully connected network (hidden dimension 128, ReLU activation) with dropout [30] for regularization. The final output layer predicts the contact state across the predefined classes. Overall, ConcatCNN enables efficient joint feature extraction and fusion by leveraging parameter sharing in the encoder and channel-wise concatenation for cross-signal integration, thereby enhancing the robustness of contact state classification. The complete architecture is depicted in Fig. 7, which highlights the shared encoders, fusion stage, and final prediction layers.

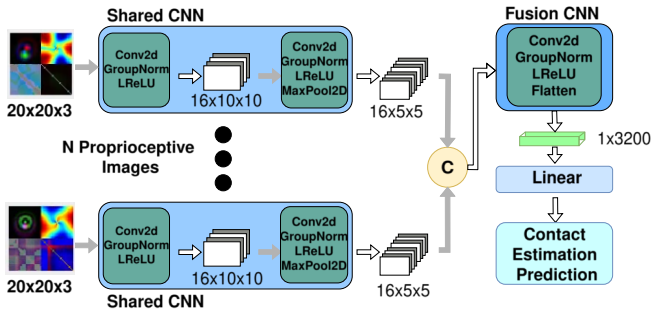


Fig. 7: Architecture of ConcatCNN for quadruped contact estimation. Each proprioceptive image is encoded by a shared CNN, and the resulting features are stacked and fused to jointly learn from multiple proprioceptive signals. In the case of fusing trunk and leg PIs, the concatenated four legs PI are separated. The fused representation is flattened and mapped through fully connected layers to predict discrete foot-ground contact states as a classification label.

### III. RESULTS

#### A. Real-World Dataset Results

In [24], a comprehensive open-source contact dataset was developed using the MIT Mini Cheetah robot, comprising approximately one million labeled data points collected over eight different terrains (asphalt, concrete, forest, grass, middle pebble, small pebble, rock road, and sidewalk) and multiple gaits. This dataset incorporates both contact and non-contact scenarios to enhance generalization. Ground truth contact labels were generated with a self-supervised algorithm that analyzes filtered foot height signals, thereby circumventing the need for additional external sensors during data collection.

The dataset was converted into PI representations using a sliding window size of  $w = 10$ , generating images  $I \in \mathbb{R}^{20 \times 20 \times 3}$ . The proposed contact estimation model, described in Section II-F, was trained with the same proprioceptive signal modalities as [24] to ensure a fair comparison. These modalities include joint positions, joint velocities, foot positions, foot velocities, trunk angular velocity, and trunk linear acceleration. Performance was assessed in terms of overall accuracy, F1 score, per-leg accuracy, and LegAvg accuracy (the mean of the individual leg accuracies). Table I summarizes the results across the different test sets, comparing our PI-based model against the approach in [24]. The PI model achieved consistently higher overall accuracy and F1 score, with the latter balancing precision and recall to account for potential class imbalance, while per-leg and LegAvg accuracies were nearly identical across methods, differing only marginally. This indicates that PI representations yield more robust global classification performance while preserving per-leg accuracy at a high level, despite relying on sequences of only 10 time steps compared to 150 in [24].

Dataset	Method	Acc(%)	F1	LF(%)	RH(%)	RF(%)	LH(%)	Leg Avg(%)
Concrete	PI CNN	<b>94.05</b>	<b>0.9337</b>	98.19	97.33	98.03	98.32	97.97
	Lin <i>et al.</i> [24]	93.04	0.4145	97.42	97.12	97.49	97.58	97.40
Forest	PI CNN	<b>91.09</b>	<b>0.8980</b>	96.49	97.66	96.89	97.07	97.03
	Lin <i>et al.</i> [24]	89.95	0.5416	96.39	97.54	97.11	96.81	96.96
Grass	PI CNN	<b>94.27</b>	<b>0.9333</b>	98.21	98.08	98.11	98.17	98.14
	Lin <i>et al.</i> [24]	92.87	0.5348	97.58	97.31	97.85	97.45	97.55
Asphalt Road	PI CNN	<b>95.20</b>	<b>0.9520</b>	98.63	98.42	98.71	98.88	98.66
	Lin <i>et al.</i> [24]	94.13	0.6023	97.74	98.19	98.46	98.28	98.17
Middle Pebble	PI CNN	<b>93.18</b>	<b>0.9270</b>	97.77	97.49	97.54	97.70	97.62
	Lin <i>et al.</i> [24]	92.45	0.4861	97.84	96.90	97.16	97.94	97.46
Small Pebble	PI CNN	<b>91.80</b>	<b>0.9032</b>	97.82	96.11	97.94	97.62	97.37
	Lin <i>et al.</i> [24]	90.65	0.4953	96.88	96.41	97.85	97.27	97.10
Sidewalk	PI CNN	<b>95.21</b>	<b>0.9446</b>	98.18	97.61	98.20	98.49	98.12
	Lin <i>et al.</i> [24]	93.71	0.4174	97.43	97.68	97.65	97.06	97.53
Rock Road	PI CNN	<b>93.72</b>	<b>0.9313</b>	97.69	98.57	97.39	98.79	98.11
	Lin <i>et al.</i> [24]	92.32	0.5338	97.32	97.67	97.29	97.55	97.46

TABLE I: Contact Estimation comparison across datasets from [24].

We also evaluated our approach using the same metrics as [25] [26], namely the foot-wise binary F1-score, the averaged F1-score (mean of the four leg-specific F1-scores), and the contact state accuracy over 16 possible states. The latter is a stricter metric than averaged accuracy, as a prediction is counted as correct only when the contact state of all four feet is simultaneously correctly predicted. To compute this metric for our model, the foot-wise predictions were converted into a 16-state representation. In addition, the real-world dataset provided by [24] was transformed into PIs using a sliding window of size  $w = 10$ , consistent with the results reported in Tab. I. Unlike the original setup, however, our model was trained on the merged training and validation splits (treated as a single training set) and evaluated on the merged test splits. The experimental results, presented in Fig. 8, demonstrate that our method outperforms all state-of-the-art data-driven methods (MI-HGNN [26], ECNN [25], CNN-aug [25], and CNN [24]) in the Legs-Avg F1 (from 0.9356 of MI-HGNN to 0.9732) and overall model accuracy (from 87.7% of MI-HGNN to 94.5%) with smaller standard deviation. These results show the potential of this representation to achieve higher accuracy with a less complex NN architecture compared to other methods [25], [26].

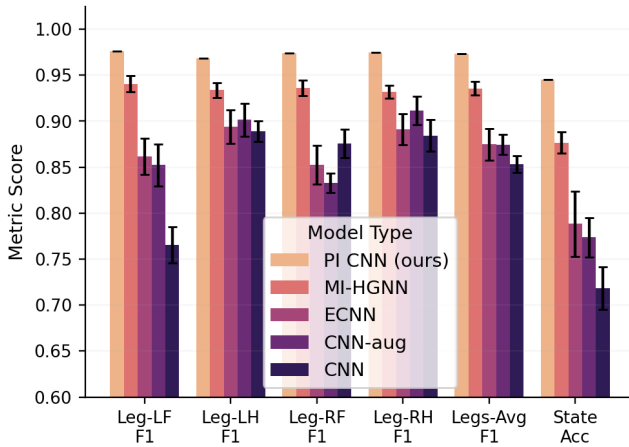
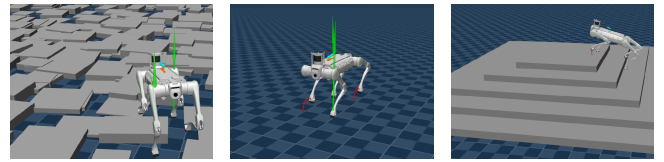


Fig. 8: Contact detection results on the real-world Mini-Cheetah contact dataset [24]: classification performance of five models on the unseen test set, trained with the entire training set. The mean and standard deviation across 8 runs are reported and compared with MI-HGNN [26], ECNN [25], CNN-aug [25] and CNN [24].

## B. Simulation Results

To further evaluate our PI formulation, we generated a dataset in the MuJoCo simulator [31] using a 60kg Unitree B2 quadruped robot. The dataset includes three environments (flat, uneven, and pyramid), two gaits (trot and crawl), and both stable ( $\mu=1.0$ ) and slippery ( $\mu=0.2$ ) terrain conditions, where  $\mu$  is the friction coefficient. The uneven and pyramid environments were randomized in relation to the boxes positions and the height of the pyramid. The robot traversed these environments under a gradient-based MPC controller developed in [32], with trunk linear velocities ranging from  $-1$  to  $1$  m/s and angular velocities from  $-0.3$  to  $0.3$  rad/s. Proprioceptive signals were recorded, including joint positions, joint velocities, foot positions, foot velocities, IMU signals and GRFs with Gaussian noise added to better approximate real-world sensing conditions. Contact estimation labels were directly extracted from the simulator and used as ground truth for training and evaluation. In total, the dataset comprises more than 1.5M synced data samples, organized by robot gait and environment stability. Each subset was further divided into training, validation, and test splits to enable fair comparison between our PI formulation, the approach of [24], and a baseline method based on GRF threshold [6], [18]. Figure 9 illustrates examples of scenes from the simulated dataset across the different environments.

Table II presents a comparison among three approaches: our PI-based CNN, the method of Lin et al. [24], and a threshold-based GRF baseline. The MuJoCo dataset is used with the same window sizes as described in Section III-A. Both models were trained on the stable environment and evaluated on three test sets: pure stable, slippery, and a fused (stable + slippery) dataset. The contact threshold was obtained by sweeping candidate values over the dataset’s



(a) Uneven Terrain (b) Flat Terrain (c) Stairs Terrain

Fig. 9: Mujoco Simulated Environment.

Test Dataset	Method	Trained	Acc(%)	F1	LF(%)	RH(%)	RF(%)	LH(%)	Leg Avg (%)
Trot - Stable	PI CNN	Stable	<b>99.46</b>	<b>0.9600</b>	99.86	99.86	99.90	99.80	98.85
	PI CNN	Fused	<b>99.39</b>	<b>0.9565</b>	99.80	99.83	99.88	99.78	99.82
	Lin et al. [24]	Stable	96.69	0.5840	99.00	98.96	99.22	99.14	99.08
	Lin et al. [24]	Fused	96.79	0.7291	98.99	98.91	98.88	99.05	98.96
Trot - Slippery	GRF TH	-	-	-	69.741	69.58	69.41	69.62	69.95
	PI CNN	Stable	<b>88.41</b>	<b>0.7165</b>	96.36	96.09	95.99	95.50	95.98
	PI CNN	Fused	<b>96.16</b>	<b>0.8925</b>	98.96	99.13	98.85	98.79	98.93
	Lin et al. [24]	Stable	86.69	0.3510	94.31	94.62	95.67	95.64	95.06
Trot - Fused	Lin et al. [24]	Fused	91.24	0.5708	96.85	96.67	97.71	97.26	97.12
	GRF TH	-	-	-	66.14	65.75	64.81	65.22	65.485
	PI CNN	Stable	<b>93.77</b>	<b>0.7899</b>	98.05	97.93	97.89	97.58	97.86
	PI CNN	Fused	<b>97.73</b>	<b>0.9103</b>	99.36	99.47	99.35	99.27	99.36
Crawl - Stable	Lin et al. [24]	Stable	91.45	0.3893	96.55	96.74	97.32	97.16	96.94
	Lin et al. [24]	Fused	93.61	0.5923	97.84	97.73	98.05	98.12	97.93
	GRF TH	-	-	-	67.93	67.66	67.10	67.105	67.53
	PI CNN	Stable	<b>98.85</b>	<b>0.9982</b>	99.76	99.71	99.67	99.70	99.71
Crawl - Slippery	PI CNN	Fused	<b>98.87</b>	<b>0.9462</b>	99.73	99.73	99.68	99.71	99.71
	Lin et al. [24]	Stable	96.73	0.7347	99.03	99.03	99.28	99.19	99.13
	Lin et al. [24]	Fused	96.17	0.7595	99.01	98.66	99.20	99.11	99.00
	GRF TH	-	-	-	82.45	82.81	81.62	81.49	82.09
Crawl - Fused	PI CNN	Stable	<b>83.48</b>	<b>0.6308</b>	95.64	96.15	94.39	94.07	95.06
	PI CNN	Fused	<b>93.18</b>	<b>0.8575</b>	98.62	98.36	98.11	97.57	98.16
	Lin et al. [24]	Stable	78.48	0.4098	92.89	92.14	94.94	93.78	93.44
	Lin et al. [24]	Fused	84.34	0.6021	94.75	95.10	96.40	96.21	95.61
Crawl - Slippery	GRF TH	-	-	-	77.54	78.82	74.28	74.09	76.18
	PI CNN	Stable	<b>94.26</b>	<b>0.7685</b>	98.52	98.64	98.10	98.03	98.32
	PI CNN	Fused	<b>97.17</b>	<b>0.8934</b>	99.40	99.32	99.21	99.08	99.25
	Lin et al. [24]	Stable	90.24	0.5374	96.71	96.54	97.75	97.49	97.12
Crawl - Fused	Lin et al. [24]	Fused	92.79	0.6840	97.55	97.71	98.38	98.34	97.99
	GRF TH	-	-	-	79.96	80.87	78.18	77.71	79.18

TABLE II: Contact Estimation comparison across datasets from Mujoco. Stable Data is the environment without slippery terrain, and Fused Data is the combined datasets of slippery and stable terrains.

GRF in the Z axis and selecting the value that maximized agreement with the ground-truth contact labels. The PI-CNN consistently outperformed the baseline across all test conditions, including the slippery environment, despite never being exposed to slippery data during training. These results highlight that PI provides a richer and more discriminative feature representation, enabling superior learning compared to directly using raw proprioceptive signals.

## C. Implementation Details

All tests were performed on a computer with Intel i9 CPU with 64 GB of RAM and NVIDIA RTX3090 GPU running Ubuntu 20.04 LTS Linux. A dropout rate of 0.3 was applied in the fully connected layers as an additional form of regularization. Training was carried out using Cross-Entropy loss for multi-class classification, optimized with adamW [33], initial learning rate of  $1 \cdot 10^{-4}$ , trained through 40 epochs with batch size of 64. To improve convergence, a ReduceLROnPlateau scheduler was employed, which decreased the learning rate by a factor of 0.2 when the validation loss failed to improve for 3 consecutive epochs.

## IV. DISCUSSION

Although PI encodings provide rich, learnable patterns, their generation is relatively slow (1.5 ms per image for  $w =$

10) due to the current CPU-based Python implementation. Increasing image resolution can provide richer feature content and improve accuracy, but it also increases encoding cost. In our experiments, the model achieved a mean GPU inference time of 8.5 ms ( $\approx 120Hz$ ). We therefore expect that a multi-threaded C++ implementation will substantially reduce generation time, making the formulation more suitable for real-time robotic systems. Beyond runtime, PIs derived from IMU signals exhibit Brownian-walk-like drift, which over long experiments leads to saturated, repetitive patterns. Finally, the kernel functions used to encode each PI component were heuristically tuned for contact estimation, suggesting that a more systematic study is needed to generalize the encoding to other tasks and robotic platforms.

## V. CONCLUSIONS

This paper presented a novel formulation of proprioceptive signals into an image representation. Each image representation encodes four important types of temporal information related to dynamic measurement behavior and is concatenated based on the morphology of the quadruped robot. The PI was evaluated on the contact estimation problem, using a simple CNN architecture trained on both simulated and real-world data, achieving the best results in comparison with state-of-the-art methods. The PI is the first step toward deeper representations of robot sensing. For future work, we expect that the PI can be integrated into other learning structures such as GNN similar to [26] and also applied as input for reinforcement learning techniques. Moreover, combining a PI with exteroceptive sensors, such as cameras, could yield a unified representation of both the robot's internal state and its external environment, further enhancing perception and state learning. Finally, we aim to explore the applicability of PI in other robotic platforms, including humanoids and manipulators.

## REFERENCES

- [1] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4960–4964, 2015.
- [2] D. Amodei *et al.*, "Deep speech 2 : End-to-end speech recognition in english and mandarin," in *International Conference on Machine Learning*, 2015.
- [3] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Interspeech*, 2019.
- [4] D. G. Llauradó, C. Mateu, J. Planes, and R. Vicens, "Using convolutional neural networks for classification of malware represented as images," *Journal of Computer Virology and Hacking Techniques*, vol. 15, pp. 15 – 28, 2018.
- [5] M. Camurri, M. F. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, and C. Semini, "Probabilistic contact estimation and impact detection for state estimation of quadruped robots," *IEEE Robotics and Automation Letters*, vol. 2, pp. 1023–1030, 2017.
- [6] Y. Nisticò, J. C. V. Soares, L. Amatucci, G. Fink, and C. Semini, "Muse: A real-time multi-sensor state estimator for quadruped robots," *IEEE Robotics and Automation Letters*, vol. 10, pp. 4620–4627, 2025.
- [7] Z. Wang and T. Oates, "Encoding time series as images for visual inspection and classification using tiled convolutional neural networks," in *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2014.
- [8] R. A. Horn and C. R. Johnson, "Matrix analysis," 1985.
- [9] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, "Recurrence plots of dynamical systems," *EPL*, vol. 4, pp. 973–977, 1987.
- [10] Y. Zhang, Y. Hou, S. Zhou, and K. Ouyang, "Encoding time series as multi-scale signed recurrence plots for classification using fully convolutional networks," *Sensors (Basel, Switzerland)*, vol. 20, 2020.
- [11] S.-H. Park, N. S. Syazwany, and S.-C. Lee, "Meta-feature fusion for few-shot time series classification," *IEEE Access*, vol. 11, pp. 41 400–41 414, 2023.
- [12] S. Quan, M. Sun, X. Zeng, X. Wang, and Z. Zhu, "Time series classification based on multi-dimensional feature fusion," *IEEE Access*, vol. 11, pp. 11 066–11 077, 2023.
- [13] C.-L. Yang, C.-Y. Yang, Z.-X. Chen, and N.-W. Lo, "Multivariate time series data transformation for convolutional neural network," *2019 IEEE/SICE International Symposium on System Integration (SII)*, pp. 188–192, 2019.
- [14] C.-L. Yang, Z.-X. Chen, and C.-Y. Yang, "Sensor classification using convolutional neural network by encoding multivariate time series as two-dimensional colored images," *Sensors (Basel, Switzerland)*, vol. 20, 2019.
- [15] Z. Wang and T. Oates, "Imaging time-series to improve classification and imputation," in *International Joint Conference on Artificial Intelligence*, 2015.
- [16] B. yu Wang, T. Jiang, X. Zhou, B. Ma, F. Zhao, and Y. Wang, "Time-series classification based on fusion features of sequence and visualization," *Applied Sciences*, 2020.
- [17] Y. Li, H. tao Yang, J. Wang, C. Zhang, Z. Liu, and H. Chen, "An image fusion method based on special residual network and efficient channel attention," *Electronics*, 2022.
- [18] G. Fink and C. Semini, "Proprioceptive sensor fusion for quadruped robot state estimation," *IROS*, pp. 10 914–10 920, 2020.
- [19] M. Menner and K. Berntorp, "Simultaneous state estimation and contact detection for legged robots by multiple-model Kalman filtering," *2024 European Control Conference (ECC)*, pp. 2768–2773, 2024.
- [20] M. Maravagakis, D. E. Argiropoulos, S. Piperakis, and P. E. Trahanias, "Probabilistic contact state estimation for legged robots using inertial information," *ICRA*, pp. 12 163–12 169, 2023.
- [21] T. Zhang, W. Cao, C. Liu, T. Zhang, J. Li, and S. E. Li, "Robust state estimation for legged robots with dual beta Kalman filter," *IEEE Robotics and Automation Letters*, vol. 10, pp. 7955–7962, 2024.
- [22] D. Youm, H. Oh, S. Choi, H. Kim, S. Jeon, and J. Hwangbo, "Legged robot state estimation with invariant extended Kalman filter using neural measurement network," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 670–676.
- [23] S. Lee, H.-B. Kim, and K.-S. Kim, "Legged robot state estimation using invariant neural-augmented Kalman filter with a neural compensator," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025, pp. 15 445–15 452.
- [24] T.-Y. Lin, R. Zhang, J. Yu, and M. Ghaffari, "Legged robot state estimation using invariant Kalman filtering and learned contact events," in *Conference on Robot Learning*, 2021.
- [25] D. Ordoñez-Apraez, G. Turrisi, V. Kostic, M. Martin, A. Agudo, F. Moreno-Noguer, M. Pontil, C. Semini, and C. Mastalli, "Morphological symmetries in robotics," *The International Journal of Robotics Research*, vol. 44, no. 10-11, pp. 1743–1766, 2025.
- [26] D. Butterfield, S. S. Garimella, N.-J. Cheng, and L. Gan, "MI-HGNN: Morphology-informed heterogeneous graph neural network for legged robot contact perception," *ICRA*, pp. 10 110–10 116, 2025.
- [27] R. V. Brill, "Applied statistics and probability for engineers," *Technometrics*, vol. 46, pp. 112 – 113, 2004.
- [28] Y. Wu and K. He, "Group normalization," *International Journal of Computer Vision*, vol. 128, pp. 742 – 755, 2018.
- [29] A. L. Maas, "Rectifier nonlinearities improve neural network acoustic models," 2013.
- [30] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *ArXiv*, vol. abs/1207.0580, 2012.
- [31] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," *IROS*, pp. 5026–5033, 2012.
- [32] G. Turrisi, V. Modugno, L. Amatucci, D. Kanoulas, and C. Semini, "On the benefits of gpu sample-based stochastic predictive controllers for legged locomotion," in *IROS*, 2024, pp. 13 757–13 764.
- [33] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2017.