Legged Robot State Estimation With Invariant Extended Kalman Filter Using Neural Measurement Network

Donghoon Youm[1], Hyunsik Oh[1], Suyoung Choi[1], Hyeongjun Kim[1], Jemin Hwangbo[1]

Abstract— This paper introduces a novel proprioceptive state estimator for legged robots that combines model-based filters and deep neural networks. Recent studies have shown that neural networks such as multi-layer perceptron or recurrent neural networks can estimate the robot states, including contact probability and linear velocity. Inspired by this, we develop a state estimation framework that integrates a neural measurement network (NMN) with an invariant extended Kalman filter. We show that our framework improves estimation performance in various terrains. Existing studies that combine model-based filters and learning-based approaches typically use real-world data. However, our approach relies solely on simulation data, as it allows us to easily obtain extensive data. This difference leads to a gap between the learning and the inference domain, commonly referred to as a sim-to-real gap. We address this challenge by adapting existing learning techniques and regularization. To validate our proposed method, we conduct experiments using a quadruped robot on four types of terrain: *flat*, *debris*, *soft*, and *slippery*. We observe that our approach significantly reduces position drift compared to the existing model-based state estimator.

I. INTRODUCTION

Recent breakthroughs in dynamic locomotion of legged robots have demonstrated the ability to navigate various terrains such as steel structures [1], mountainous terrains [2], and sandy fields [3]. These remarkable developments have stemmed from the advancements in various control methods. One of these control methodologies is model predictive control, which plans the future trajectories based on the current robot state. Therefore, the accurate estimation of the current robot states directly affects control performance. Another control methodology, learning-based control, estimates the robot states explicitly or implicitly through a neural network and then uses it for control [4], [5]. Regardless of the control methodology, accurate and reliable state estimation is a crucial stepping stone for dynamic control.

State estimation using proprioceptive sensors, including magnetic encoders and Inertial Measurement Unit (IMU), is a low-cost, lightweight, and power-efficient solution. These sensors provide direct access to joint angles, joint velocity, acceleration, and angular velocity. Processing these proprioceptive data with a state estimation algorithm can reconstruct the unknown states such as position, orientation, and linear velocity.

Researchers have shown that the state estimation accuracy can be further improved by adding exteroceptive sensors such as Lidar and cameras [6]. However, they are vulnerable

[1]Korea Advanced Institute of Science and Technology, Yuseong-gu, Daejeon, 34141, Republic of Korea `ydh0725, ohsik1008, swimchoy, kaist0914, jhwangbo@kaist.ac.kr`
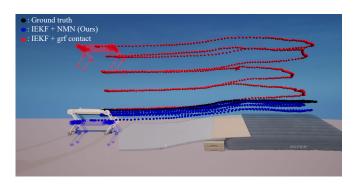
Fig. 1.   By combining our Neural Measurement Network (NMN) with a model-based filter, our method can reduce the position estimation error for the entire trajectory of the Raibo robot on an air mattress using only proprioceptive sensors by one-third compared to the purely model-based methods.

to lighting variations, low-texture environments, and motion blur. Consequently, they may not be a suitable estimator for dynamic control in which state estimation results are important. Research on proprioceptive-sensor-based estimators for legged robots has been actively pursued for these reasons.

Bloesch et al. [7] introduced an EKF, which uses IMU and contact motion models. This approach incorporates contact foot kinematics as a measurement, assuming non-slip contact with the feet. The following research has employed the Unscented Kalman Filter (UKF) or factor graph methods to improve estimation performance [8], [9]. A contact-aid Invariant Extended Kalman Filter (IEKF), applied by Hartley et al. [10] to legged robots, achieves enhanced convergence and robustness by utilizing the properties of the filter's state space formulated as a Lie group. Research has indicated that the IEKF offers superior convergence and consistency compared to the conventional quaternion-based extended Kalman filter (QEKF) [11].

Due to their underlying non-slip assumption, the aforementioned approaches are fragile to un-modeled situations such as foot slip or ambiguous contact in soft terrain, even when using contact sensors. To address these issues, Bloesch et al. [8] devised a stochastic filtering method, handling outlier measurements through the threshold of the Mahalanobis distance of innovation. Kim et al. [12] introduced a slip rejection method that identifies slip via the contact foot's velocity and modified the kinematics model's covariance, improving robustness against foot slip.

The data-driven method is another promising approach for state estimation. Recent research in inertial navigation systems (INS) and inertial odometry (IO) have leveraged ma-

chine learning to estimate the motion directly from IMU data. Pioneering work by Chen et al. [13] introduced IoNet, which used a two-layer bidirectional Long Short-Term Memory (Bi-LSTM) to predict a pedestrian's location transforms in polar coordinates from raw IMU data. Yan et al. [14] presented RoNIN, which infers the velocity and heading direction of pedestrians directly from raw IMU data, and they evaluated the performance of various backbone networks such as ResNet, LSTM, and Time Convolutional Network (TCN). In a different direction, Brossard et al. [15] and Zhang et al. [16] improved the estimation performance through learning-based methods to denoise raw IMU data rather than estimating the pose directly.

Many researchers have investigated combining the strengths of both model-based and learning-based estimation strategies. The fusion of the EKF framework and neural networks is applied to various domains including pedestrian tracking [17], [18], wheeled vehicles [19], [20], quadrotors [21], [22], and quadruped robots [23], [24]. Liu et al. [17] developed a neural network that learns relative displacement measurements and covariance from buffered IMU data, and integrated the neural network with the EKF. Buchanan et al.[23] expanded this work to quadruped robots by incorporating the factor graph framework. Although these approaches outperform existing model-based or learning-based methods, their reliance on relative displacement measurements restricts compatibility with the IEKF framework. This limitation occurs because the state space of the estimator using relative displacement measurements does not satisfy the Lie Group property. To this end, Lin et al. [24] proposed a learning-based contact estimator for quadruped robots and integrated it with contact-aid IEKF to bypass these limitations and the need for contact sensors.

Learning-based methods have either relied on well-established real-world datasets [25], [26] or required efforts to gather data, as a sufficient amount of data is essential for training. Unfortunately, the data collection and post-processing are extremely expensive and laborious, especially for legged robots. In this light, the existing research in legged robotics has been focusing on using only simulation data. However, methods for training a state estimator to estimate position and orientation relying solely on simulation data have not been rigorously studied.

We present a novel proprioceptive state estimator that integrates a neural measurement network (NMN) with an IEKF. Our approach capitalizes on extensive simulation datasets to train NMN while making use of the superior convergence and robustness of IEKF. In this study, we choose linear velocity expressed in the body frame and contact probability as an output of NMN and use them as observations in the correction step of the IEKF. The key challenge to our approach is the sim-to-real gap: the discrepancy between the simulation environment and the real world. To reduce this discrepancy, we test existing learning techniques, such as early stopping and domain randomization, in our training and meticulously analyze their efficiency. We also introduce a smoothness loss in supervised learning for the regularization
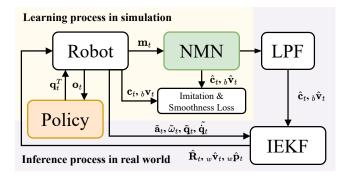

Fig. 2. **Schematic of the proposed framework:** To devise a learning-based state estimator, we first develop the policy for trajectory sampling. The NMN takes as input the proprioceptive sensor value and previous joint target and outputs contact probability and body linear velocity. These measurements are passed to the LPF and used to update the IEKF.

of NMN output. Under diverse scenarios such as flat, debris, soft, and slippery terrains, our approach shows superior state estimation performance to the existing model-based state estimator. We summarize our contributions as follows.

- We propose a novel proprioceptive state estimator that integrates a Neural Measurement Network (NMN) with IEKF.
- We analyze the efficacy of existing learning techniques in reducing the sim-to-real gap of NMN learned only through simulation data.
- We empirically demonstrate that NMN improves state estimation performance on flat, rough, soft, and slippery terrain.

II. METHOD

In this section, we introduce a state estimation framework that integrates the Invariant Extended Kalman Filter (IEKF) with a Neural Measurement Network (NMN). This framework aims to make use of model-based and data-driven state estimators. The Schematic of the proposed framework is depicted in Figure 2. During the learning process, we sample trajectories through the locomotion policy in the simulation and train the NMN. In the inference process, we use the NMN as a measurement model in IEKF. Hereinafter, $\tilde{(\cdot)}$ represents the sensor value, $(\cdot)$ represents the true value, and $\hat{(\cdot)}$ represents the estimated value.

A. Learning Process in Simulation

1) Architectures of Locomotion Policy and NMN: We first train the locomotion policy of GRU [27]-MLP architecture to control the quadruped robot. Thereafter, we freeze the policy network parameters and employ this policy throughout the training and inference processes. The observation of the locomotion policy is $\mathbf{o}_t \triangleq [\tilde{\mathbf{g}}_t, \tilde{\omega}_t, \tilde{\mathbf{q}}_t, \tilde{\dot{\mathbf{q}}}_t, \mathbf{q}_{t-1}^{des}, \mathbf{u}_t]^T$, where each component represents the roll-pitch vector of the body, body angular velocity, joint angles, joint velocities, previous positional joint targets, and velocity command at a given time t, respectively. Specifically, a vector $[\tilde{\mathbf{g}}_t, \tilde{\omega}_t, \tilde{\mathbf{q}}_t, \tilde{\dot{\mathbf{q}}}_t, \mathbf{q}_{t-1}^{des}]^T$ is first passed to the GRU and then concatenated with

the hidden state of GRU and $\mathbf{u}_t$. The following MLP takes this stacked vector to produce new positional joint targets. To assess the standalone performance of the estimator, we intentionally separated it from the controller so that the policy observations can remain independent of states estimated. Therefore, the roll-pitch vector is obtained directly from the IMU attached to the body.

The NMN consists of the GRU-MLP architecture and measures $\hat{\mathbf{c}}_t$ and $_b\hat{\mathbf{v}}_t$, which are the contact probabilities of feet and the body linear velocity. Inputs to the NMN are $\mathbf{m}_t \triangleq [\tilde{\mathbf{a}}_t, \tilde{\omega}_t, \tilde{\mathbf{q}}_t, \tilde{\dot{\mathbf{q}}}, \mathbf{q}_{t-1}^{des}]^T$, where $\tilde{\mathbf{a}}_t$ represents acceleration. The hidden dimension of GRU was 128, and the MLP network has a $[256 \times 128]$ structure for both locomotion policy and the NMN.

2) Simulation Details: We train the control policy and the NMN using the Raisim [28] simulator. The policy utilizing the PPO [29] is trained in the randomized terrain environment, as described by Lee et al. [30]. The terrain consists of flat, hills, steps, and stairs. Specifically, the amplitude of the hills follows $U(0.2, 1.4)$, the height of the steps follows $U(0.02, 0.18)$ cm, the height of the stairs follows $U(0.02, 0.18)$ cm, and the friction coefficient follows $U(0.4, 1.2)$. Additionally, to simulate unexpected slippage, we set the friction coefficient of the contact instance to the value sampled from $U(0.3, 0.4)$ with a 1% probability. Training details such as action space, reward functions, and control frequency are nearly identical to the work of Choi et al. [3].

The simulator and estimator operate at 4kHz and 500Hz, respectively. We generate trajectories through the pre-trained control policy and gather labels for the contact state and body linear velocity. For every iteration of the NMN training, all 400 environments are randomly initialized and generate a trajectory of 400 control steps. We inject zero mean Gaussian noise into $\mathbf{o}_t$ and $\mathbf{m}_t$ to simulate sensor noise. The noise ranges correspond to the specifications of the joint encoder and IMU. The simulation settings for the control policy training and the data collection are the same. As with our control policy, reinforcement learning-based controllers often bridge the sim-to-real gap by introducing an extensive diversity to the domain properties, including robot kinematics or inertial parameters [3], [5]. However, this strategy would be disadvantageous to the data-driven estimation scheme. We discover that applying domain randomization to our training can lower the velocity prediction accuracy once we have a sufficiently accurate robot model. We validate this result through an ablation study, which is detailed in the experimental results section.

3) Loss Functions: Our loss function consists of two loss terms – supervised learning loss L_{sp} and regularization loss L_{sm} for smoothness along the time axis, defined as follows:

$$
\begin{aligned}
L_{sp} &= L_{BCE}(\mathbf{c}_t, \hat{\mathbf{c}}_t) + L_{L1}(_b\mathbf{v}_t, _b\hat{\mathbf{v}}_t) \\
L_{sm} &= \frac{1}{N}(||_b\hat{\mathbf{v}}_t - _b\hat{\mathbf{v}}_{t-1}||^2 + \frac{1}{2}||_b\hat{\mathbf{v}}_t - 2_b\hat{\mathbf{v}}_{t-1} + _b\hat{\mathbf{v}}_{t-2}||^2) \quad (1) \\
L_{total} &= L_{sp} + \lambda L_{sm}
\end{aligned}
$$

The loss for supervised learning includes the binary cross-entropy loss for the contact probability and the mean absolute error loss for the body linear velocity. Inspired by Mysore et al. [31] and their CAPS proposal, we introduce the smoothness loss. This loss minimizes the first and second-order changes in the network's output to ensure a smoother result and narrow the sim-to-real gap. We apply the smoothness loss only to $_b\hat{\mathbf{v}}_t$ because the contact states are inherently discrete. In this context, λ is a balancing factor between two loss terms, and we have set this to 50 throughout training. We further investigate the efficacy of the regularization in the experimental results section.

We use the Adam optimizer [32] for the training, with a learning rate of $5e-4$ and 32 epochs during the first 200 iterations. If the training continues further, the NMN would overfit the simulator and increase the sim-to-real gap. To avoid this problem, we use the early-stopping and the advantage of this technique is discussed in more detail in the experimental results section.

B. Invariant Extended Kalman Filter

1) State Representation: Our framework is built upon the IEKF for a legged robot, which is a world-centric right-invariant EKF proposed by Hartley et al. [10]. For ease of explanation, in this section, we describe the state space and system dynamics with the IMU biases omitted, but we implemented them in practice. The IMU frame coincides with the robot body frame. State variable $\mathbf{X}_t \in \mathrm{SE}_{N+2}(3)$ forms a matrix Lie group consisting of body state and N contact points, which is defined as follows:

$$
\mathbf{X_t} \triangleq \begin{bmatrix} \mathbf{R_t} & \mathbf{v_t} & \mathbf{p_t} & \mathbf{d_t} \\ \mathbf{0}_{1,3} & 1 & 0 & 0 \\ \mathbf{0}_{1,3} & 0 & 1 & 0 \\ \mathbf{0}_{1,3} & 0 & 0 & 1 \end{bmatrix} \quad (2)
$$

where $\mathbf{R}_t \in \mathrm{SO}(3), \mathbf{v}_t, \mathbf{p}_t$, and $\mathbf{d}_t$ represent the rotation matrix, linear velocity, position of the body, position of the contact point expressed in world frame, respectively. In the state matrix, X_t can include multiple contact points. However, the system dynamics and measurement models for each contact point are independent. For better readability, we represent the state space for a single contact case.

2) System Dynamics: The IMU measurements are modeled as the sum of the true values and Gaussian noise, as follows:

$$
\begin{aligned}
\tilde{\omega}_t &= \omega_t + \mathbf{w}_t^g, \quad \mathbf{w}_t^g \sim \mathcal{N}(\mathbf{0}_{3,1}, \Sigma^{\mathbf{g}}) \\
\tilde{\mathbf{a}}_t &= \mathbf{a}_t + \mathbf{w}_t^a, \quad \mathbf{w}_t^a \sim \mathcal{N}(\mathbf{0}_{3,1}, \Sigma^{\mathbf{a}})
\end{aligned} \quad (3)
$$

where $\mathcal{N}$ represents Gaussian noise. The position of the contact point remains stationary as long as the slip does not happen. We model the uncertainties on the contact position by adding Gaussian noise to the velocity of the contact point.

$$
_W\tilde{\mathbf{v}}_C = \mathbf{0}_{3,1} = _C\mathbf{v}_C + \mathbf{w}_t^v, \quad \mathbf{w}_t^v \sim \mathcal{N}(\mathbf{0}_{3,1}, \Sigma^{\mathbf{v}}) \quad (4)
$$

The system dynamics given the IMU and contact measurements are as follows:

$$\frac{d}{dt}\mathbf{R}_t = \mathbf{R}_t(\tilde{\omega}_t - \mathbf{w}_t^g)_\times$$
$$\frac{d}{dt}\mathbf{v}_t = \mathbf{R}_t(\tilde{\mathbf{a}}_t - \mathbf{w}_t^a) + \mathbf{g}$$
$$\frac{d}{dt}\mathbf{p}_t = \mathbf{v}_t \tag{5}$$
$$\frac{d}{dt}\mathbf{d}_t = \mathbf{R}_t\mathbf{h}_R(\tilde{\mathbf{q}}_t)(-\mathbf{w}_t^v)$$

where $(\cdot)_\times$ denotes a 3×3 skew-symmetric matrix and $\mathbf{h}_R(\tilde{\mathbf{q}}_t)$ is the orientation of the contact frame with respect to the body frame. This can be calculated from encoder measurements, $\tilde{\mathbf{q}}_t$, and forward kinematics.

3) Right-Invariant Leg Kinematics Measurement Model: To mimic the real sensor data, we further modeled the joint encoder measurements $\tilde{\mathbf{q}}_t$ with Gaussian noise as follows:

$$\tilde{\mathbf{q}}_t = \mathbf{q}_t + \mathbf{w}_t^q, \quad \mathbf{w}_t^q \sim \mathcal{N}(\mathbf{0}_{12,1}, \Sigma^{\mathbf{q}}) \tag{6}$$

The contact point under the non-slip conditions can be calculated through forward kinematics and expressed using state variables as follows:

$$\mathbf{h}_p(\tilde{\mathbf{q}}_t) = \mathbf{R}_t^T(\mathbf{d}_t - \mathbf{p}_t) + \mathbf{J}_p(\tilde{\mathbf{q}}_t)\mathbf{w}_t^q \tag{7}$$

where $\mathbf{h}_p(\tilde{\mathbf{q}}_t)$ is the forward kinematics of contact foot, and $\mathbf{J}_p(\tilde{\mathbf{q}}_t)$ is the analytical Jacobian of the forward kinematics. As Barrau et al. [33] explained, the leg kinematics measurement model (7) follows right-invariant observation form, $\mathbf{Y}_t = \mathbf{X}_t^{-1}\mathbf{b} + \mathbf{V}_t$, where $\mathbf{Y}_t^T = \begin{bmatrix} \mathbf{h}_p^T(\tilde{\mathbf{q}}_t) & 0 & 1 & -1 \end{bmatrix}$, $\mathbf{b}^T = \begin{bmatrix} \mathbf{0}_{1,3} & 0 & 1 & -1 \end{bmatrix}$, $\mathbf{V}_t^T = \begin{bmatrix} (\mathbf{J}_p(\tilde{\mathbf{q}}_t)\mathbf{w}_t^q)^T & 0 & 0 & 0 \end{bmatrix}$.

4) Right-Invariant Neural Measurement Model: As shown in Figure 2, we construct the Neural Measurement Network (NMN) to measure contact probability $\hat{\mathbf{c}}_t$, and body linear velocity $_b\hat{\mathbf{v}}_t$. We utilize the learned-contact probability to determine the use of the leg kinematics measurement model. In addition, we formulate the learned-body-linear-velocity as follows:

$$_b\hat{\mathbf{v}}_t = \mathbf{R}_t^T\mathbf{v}_t + \mathbf{w}_t^{NN}, \quad \mathbf{w}_t^{NN} \sim \mathcal{N}(\mathbf{0}_{3,1}, \Sigma^{NN}) \tag{8}$$

where $\mathbf{w}_t^{NN}$ is Gaussian noise that represents the uncertainty of neural measurement. Our learned-body-linear-velocity formulation (8) also satisfies the right-invariant observation form, where $\mathbf{Y}_t^T = \begin{bmatrix} _b\hat{\mathbf{v}}_t & -1 & 0 & 0 \end{bmatrix}$, $\mathbf{b}^T = \begin{bmatrix} \mathbf{0}_{1,3} & -1 & 0 & 0 \end{bmatrix}$, $\mathbf{V}_t^T = \begin{bmatrix} (\mathbf{w}_t^{NN})^T & 0 & 0 & 0 \end{bmatrix}$.

5) Kalman Update: By utilizing the log-linear property of the right-invariant error and first-order approximation, full state and covariance update equations of the IEKF can be formulated as follows:

$$\bar{\mathbf{X}}_t^+ = \exp(\mathbf{K}_t\Pi\bar{\mathbf{X}}_t\mathbf{Y}_t)\bar{\mathbf{X}}_t$$
$$\mathbf{P}_t^+ = (\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)\mathbf{P}_t(\mathbf{I} - \mathbf{K}_t\mathbf{H}_t)^T + \mathbf{K}_t\bar{\mathbf{N}}_t\mathbf{K}_t^T \tag{9}$$

where $\bar{\mathbf{X}}_t$ is the estimated state from system dynamics, $\mathbf{P}_t$ is the covariance matrix, and $\Pi \triangleq \begin{bmatrix} \mathbf{I} & \mathbf{0}_{3,3} \end{bmatrix}$ is the auxiliary selection matrix. Kalman gain $\mathbf{K}_t$ is computed as the following:

$$\mathbf{S}_t = \mathbf{H}_t\mathbf{P}_t\mathbf{H}_t^T + \bar{\mathbf{N}}_t, \quad \mathbf{K}_t = \mathbf{P}_t\mathbf{H}_t^T\mathbf{S}_t^{-1} \tag{10}$$

We set the observation model as $\mathbf{H}_t = \begin{bmatrix} {}^q\mathbf{H}_t^T & {}^v\mathbf{H}_t^T \end{bmatrix}^T$ and $\mathbf{N}_t = \begin{bmatrix} {}^q\mathbf{N}_t & \mathbf{0}_{3,3}; & \mathbf{0}_{3,3} & {}^v\mathbf{N}_t \end{bmatrix}$, where ${}^q(\cdot)$ is for the leg kinematics model, and ${}^v(\cdot)$ represents the learned-body-linear-velocity model. The observation model for the leg kinematics measurement can be drawn from the linear update equation as follows:

$${}^q\mathbf{H}_t = \begin{bmatrix} \mathbf{0}_{3,3} & \mathbf{0}_{3,3} & -\mathbf{I} & \mathbf{I} \end{bmatrix}$$
$${}^q\bar{\mathbf{N}}_t = \bar{\mathbf{R}}_t\mathbf{J}_p(\tilde{\mathbf{q}}_t)\text{Cov}(\mathbf{w}_t^q)\mathbf{J}_p^T(\tilde{\mathbf{q}}_t)\bar{\mathbf{R}}_t^T \tag{11}$$

The observation model for the learned-body-linear-velocity measurement is as follows:

$${}^v\mathbf{H}_t = \begin{bmatrix} \mathbf{0}_{3,3} & \mathbf{I} & \mathbf{0}_{3,3} & \mathbf{0}_{3,3} \end{bmatrix}$$
$${}^v\bar{\mathbf{N}}_t = \bar{\mathbf{R}}_t\text{Cov}(\mathbf{w}_t^{NN})\bar{\mathbf{R}}_t^T \tag{12}$$

We refer the reader to Hartley et al. [10] for more details on the material described above and additional IMU bias augmentation explanation.

C. Inference Process in Real World

During the inference process, we set the covariance as follows:

$$\text{Cov}(\mathbf{w}_t^g, \mathbf{w}_t^a, \mathbf{w}_t^v, \mathbf{w}_t^q, \mathbf{w}_t^{NN}, \mathbf{w}_t^{b_g}, \mathbf{w}_t^{b_a})$$
$$= (10^{-5}, 10^{-1}, 10^{-4}, 10^{-6}, 10^{-5.5}, 10^{-10}, 10^{-10})\mathbf{I}_{3,3}$$
$$\text{Cov}(\mathbf{R}_0, \mathbf{v}_0, \mathbf{p}_0, \mathbf{b}_{g_0}, \mathbf{b}_{a_0})$$
$$= (10^{-8}, 10^{-8}, 10^{-8}, 10^{-10}, 10^{-10})\mathbf{I}_{3,3} \tag{13}$$

where $\mathbf{w}_{b_g}, \mathbf{w}_{b_a}$ represent bias uncertainties. The true uncertainty regarding the contact position should be $\mathbf{h}_R(\tilde{\mathbf{q}}_t)(-\mathbf{w}_t^v)$, which is indicated in (5). However, due to the unknown orientation of the contact frame, we approximate this to a constant value. We additionally utilize a first-order low-pass filter (LPF) in the inference process with cutoff frequencies of 40Hz for $\hat{\mathbf{c}}_t$ and 10Hz for $_b\hat{\mathbf{v}}_t$. We threshold $\hat{\mathbf{c}}_t$ to 0.5 to determine the contact state and use it in IEKF. We have found that although the robot remains stationary, the learned-body-linear-velocity measurement of the NMN has a bias with a precision on the order of 10^{-3}. To prevent position drift caused by this bias, we apply (8) as the measurement model only when $\hat{\mathbf{v}}_t$ exceeds 0.1.

III. EXPERIMENTAL RESULTS

We design experiments to investigate the following research questions: (1) Can our framework improve the state estimation performance using only simulation data? (2) How can we reduce the sim-to-real gap in our supervised approach?

We gather sensor measurements using Raibo and process these measurements offline through our framework. The IMU (Microstrain-3DM) delivers data at 166 Hz, and encoder measurements are recorded at 500 Hz. To calculate the estimation error, we collect ground truth data from a motion capture system (12 Vicon Vero V2.2 cameras) at 250Hz and then interpolate the motion capture data to 500Hz for compatibility with our estimation results. We align the initial state with the ground truth and set the IMU bias to zero.
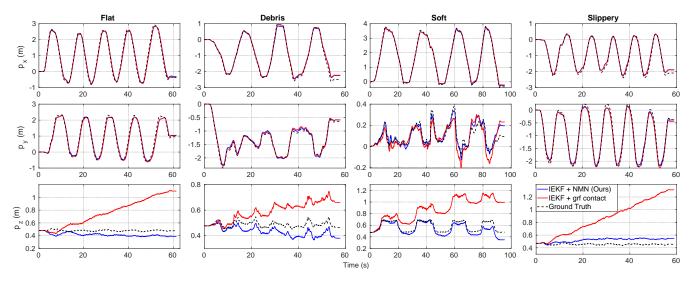
Fig. 3. **Position estimation comparison:** Proposed NMN vs. model-based GRF contact on four different terrains. Our NMN reduces the z-direction position drift compared to the model-based approach. Either method employs the slip rejection method.

Fig. 4. **Experiment Terrain Overview:** Upper left - *Flat*, Upper middle - *Soft*, Upper right - *Slippery*, Lower - *Debris*.

A. Experimental Details

1) Baselines: We select four baseline estimators as follows:

- **ground reaction force (GRF) contact:** The first baseline estimator is based on IEKF and utilizes the contact state calculated through the GRF for the leg kinematics measurement. In this estimator, we calculate the GRF from robot dynamics and filter it through an LPF with a cut-off frequency of 10Hz. The contact state is determined by thresholding the filtered GRF at 40N.
- **IEKF with NMN $\hat{\mathbf{c}}_t$:** The second baseline estimator is based on IEKF and utilizes the $\hat{\mathbf{c}}_t$ predicted by NMN for the leg kinematics measurement. This estimator does not utilize the predicted $_b\hat{\mathbf{v}}_t$ for the Kalman update, i.e., $\mathbf{H}_t =^q \mathbf{H}_t$ and $\bar{\mathbf{N}}_t =^q \bar{\mathbf{N}}_t$.
- **IEKF with NMN $_b\hat{\mathbf{v}}_t$:** The third baseline estimator is based on IEKF and utilizes the $_b\hat{\mathbf{v}}_t$ predicted by NMN for the body linear velocity measurement. This estimator does not utilize the predicted $\hat{\mathbf{c}}_t$ for the Kalman update, i.e., $\mathbf{H}_t =^v \mathbf{H}_t$ and $\bar{\mathbf{N}}_t =^v \bar{\mathbf{N}}_t$.
- **NN only:** The fourth baseline estimator is a GRU-MLP network trained using end-to-end regression method. We

train the GRU-MLP network to directly estimate relative position $\mathbf{R}_{t-1}^T(\mathbf{p}_t - \mathbf{p}_{t-1})$, orientation $Log(\mathbf{R}_{t-1}^T\mathbf{R}_t)$, and linear velocity $_b\hat{\mathbf{v}}_t$. The estimation result is calculated by integrating the network output. Jin et al. [20] described that learning the relative pose can be inconsistent when only the IMU measurements are available. However, we assume that access to leg kinematics can circumvent such problems by making $_b\hat{\mathbf{v}}_t$ and $\mathbf{g}_t$ to be observable.

2) Slip rejection: The slip rejection method proposed by Kim et al. [12] is suitable for use with any state estimators using leg kinematics measurement. Since our estimator utilizes leg kinematics measurement, we can apply this method to our estimator and study how it performs.

In the rejection method, we detect foot slip when the velocity of the contact foot is larger than the threshold, which we set to 0.4m/s. When the slip is detected, we increase the covariance of the contact foot velocity $Cov(\mathbf{w}_t^v)$ by a factor of ten so that the uncertainties of the leg kinematics model are reflected in the Kalman update.

B. Hardware Experiments

We test four different terrains to evaluate the proposed NMN and IEKF algorithms: *flat, debris, soft*, and *slippery*. Figure 4 illustrates the experimental setup.

- *Flat:* The robot traverses a circular trajectory of 49 meters of flat ground for 62 seconds with a friction coefficient of over 0.6.
- *Debris:* We simulate the debris terrain with randomly stacked wooden logs and bricks, and the robot repeatedly navigates obstacles over 26 meters for 59 seconds.
- *Soft:* The terrain consists of thin mats and air mats, and the robot makes four round trips of 33 meters for 98 seconds.
- *Slippery:* We disperse the boric acid powder on a flat surface, lowering the coefficient of friction to less than

TABLE I

ATE AND 10-SECOND RE ON STATE ESTIMATION RESULTS ACROSS VARIOUS ENVIRONMENT EXPERIMENTS

Terrain	Slip rejection	Method	ATE (pos)	ATE (vel)	ATE (ori)	RE (pos)	RE (vel)	RE (ori)
Flat	Off	**Proposed**	**0.2110**	0.0759	0.0824	0.2350	**0.0972**	0.0396
		w/ $_b\hat{v}_t$	0.2306	0.0781	0.0824	0.2351	0.0983	0.0399
		w/ $\hat{c}_t$	0.3585	0.0791	0.0824	0.2418	0.1035	0.0397
		GRF contact	0.4346	**0.0747**	0.0824	0.2399	0.0995	0.0397
		NN only	0.8142	0.1212	0.4100	0.8674	0.1727	0.1121
	On	**Proposed**	0.2142	0.0760	0.0814	**0.2332**	0.0972	**0.0396**
		w/ $\hat{c}_t$	0.3169	0.0789	0.0817	0.2390	0.1040	0.0398
		GRF contact	0.3459	0.0770	**0.0809**	0.2363	0.1031	0.0398
Debris	Off	**Proposed**	0.1798	0.0877	**0.0591**	0.1166	0.1281	0.0625
		w/ $_b\hat{v}_t$	0.1626	0.0917	0.0610	0.1173	0.1299	0.0618
		w/ $\hat{c}_t$	0.2630	0.1030	0.0597	0.1321	0.1563	0.0618
		GRF contact	0.3074	0.0884	0.0604	0.1087	0.1325	**0.0615**
		NN only	0.3983	0.1254	0.0784	0.1845	0.1932	0.1000
	On	**Proposed**	0.1738	0.0875	0.0599	0.1165	0.1280	0.0619
		w/ $\hat{c}_t$	**0.1290**	0.0882	0.0601	0.1020	0.1292	0.0617
		GRF contact	0.1478	**0.0848**	0.0608	**0.0933**	**0.1261**	0.0617
Soft	Off	**Proposed**	0.2450	**0.0799**	0.0296	**0.0772**	**0.1180**	0.0404
		w/ $_b\hat{v}_t$	0.2313	0.0835	0.0452	0.1031	0.1200	0.0408
		w/ $\hat{c}_t$	0.2251	0.0988	**0.0282**	0.1218	0.1451	**0.0398**
		GRF contact	0.7228	0.0858	0.0386	0.1282	0.1238	0.0402
		NN only	0.5087	0.1035	0.2266	0.3209	0.1551	0.1269
	On	**Proposed**	**0.2059**	0.0800	0.0362	0.0869	0.1182	0.0407
		w/ $\hat{c}_t$	0.3093	0.0819	0.0361	0.1059	0.1180	0.0401
		GRF contact	0.3115	0.0820	0.0401	0.1004	0.1196	0.0403
Slippery	Off	**Proposed**	0.1792	0.0694	**0.0467**	0.1421	0.0774	**0.0199**
		w/ $_b\hat{v}_t$	0.1751	0.0715	0.0675	0.1619	0.0783	0.0234
		w/ $\hat{c}_t$	0.5916	0.1016	0.0511	0.2075	0.1246	0.0216
		GRF contact	0.6675	0.0931	0.0527	0.2114	0.1151	0.0215
		NN only	0.4223	0.1015	0.2971	0.4018	0.1448	0.0747
	On	**Proposed**	**0.1623**	**0.0693**	0.0509	0.1432	**0.0773**	0.0208
		w/ $\hat{c}_t$	0.3807	0.0792	0.0526	0.1558	0.0948	0.0216
		GRF contact	0.4412	0.0729	0.0528	0.1556	0.0876	0.0215
Total Average	Off	**Proposed**	0.2038	0.0782	**0.0545**	0.1427	**0.1052**	**0.0406**
		w/ $_b\hat{v}_t$	0.1999	0.0812	0.0640	0.1544	0.1066	0.0415
		w/ $\hat{c}_t$	0.3596	0.0956	0.0554	0.1758	0.1324	0.0407
		GRF contact	0.5331	0.0855	0.0584	0.1721	0.1177	0.0407
		NN only	0.5359	0.1129	0.2530	0.4437	0.1665	0.1034
	On	**Proposed**	**0.1891**	**0.0782**	0.0571	0.1450	0.1052	0.0408
		w/ $\hat{c}_t$	0.2840	0.0821	0.0576	0.1507	0.1115	0.0408
		GRF contact	0.3116	0.0792	0.0587	0.1464	0.1091	0.0408

TABLE II

ABLATION STUDY: Impact of Smoothness Loss and Domain Randomization on Various Environment Experiments

Terrain	Method	ATE (pos)	ATE (vel)	ATE (ori)	RE (pos)	RE (vel)	RE (ori)
Flat	**Proposed**	**0.2110**	**0.0759**	0.0824	**0.2350**	**0.0972**	0.0396
	w/o smoothness	0.4592	0.0891	0.0820	0.2455	0.1120	0.0393
	w/ randomization	0.3556	0.0864	**0.0786**	0.2543	0.0992	**0.0389**
Debris	**Proposed**	0.1798	0.0877	0.0591	**0.1166**	**0.1281**	0.0625
	w/o smoothness	0.3650	0.0981	**0.0589**	0.1480	0.1418	0.0629
	w/ randomization	0.3121	0.0934	0.0603	0.1579	0.1325	**0.0623**
Soft	**Proposed**	0.2450	**0.0799**	0.0296	0.0772	**0.1180**	0.0404
	w/o smoothness	**0.1338**	0.0874	0.0305	**0.0771**	0.1285	0.0409
	w/ randomization	0.4432	0.0825	**0.0285**	0.1165	0.1205	**0.0398**
Slippery	**Proposed**	0.1792	**0.0694**	**0.0467**	0.1421	**0.0774**	**0.0199**
	w/o smoothness	**0.1785**	0.0845	0.0491	0.1644	0.1021	0.0201
	w/ randomization	0.1913	0.0882	0.0498	0.1788	0.0823	0.0206

0.3. The robot traverses a circular trajectory of 45 meters for 59 seconds.

Robots are often faced with ambiguous contact situations, except for the *flat*.

1) Estimation results: The estimated positions for each terrain are shown in Figure 3. Compared to the baseline, NMN shows a substantially small z-direction drift. The quantitative estimation results are presented in Table I. In this table, we highlight the best-performing metrics in bold. For methodologies employing Neural Networks (NN), we train the network from five different seeds and average the outcomes. The definitions for Absolute Trajectory Error (ATE) and Relative Error (RE) refer to [34], where RE is computed with 10-second sub-trajectories. The error units are rad, m/s, m for rotation, velocity, and position, respectively.

IEKF with NMN $\hat{c}_t$ method shows a lower ATE_{pos} and a higher ATE_{vel} than **IEKF with GRF contact** method. The difference between the two methods is trivial, indicating that the learned-contact is similar to that calculated through GRF. We suspect this small difference stems from the simulator's tendency to overestimate contact. The learned-contact is often more sensitive than the GRF thresholding scheme because, in the training environment, we can identify the contact even with a small interaction force and provide the simulated contact state as the ground-truth label. **IEKF with NMN $_b\hat{v}_t$** method has generally performed well among the estimator, especially for the terrains that cause unstable contact. This result suggests that state estimators for legged robots can improve estimation performance through the body linear velocity measurement model.

We find that our proposed method yields impressive outcomes for *soft*, which is not covered in training scenarios. While the slip rejection method does bolster the accuracy of estimations based on leg kinematics measurements, our strategy outperforms it in many cases. The **NN only** method shows high estimation errors across different terrains. Overall, we find that the NMN using only simulation data can improve the state estimation performance and exhibit adaptability to challenging terrains where unstable contacts occur, even including contact scenarios not previously encountered.

2) Ablation study: In this section, we delve into the effects of the smoothness loss, domain randomization, and early stopping, which are the techniques we introduced to mitigate the sim-to-real gap. Before conducting the ablation study, we would like to clarify key aspects of the proposed method. Our proposed method uses the smoothness loss function and does not employ domain randomization during the learning process. We conduct an ablation study on smoothness loss function and domain randomization.

The estimation results of the ablation study are presented in Table II. Our proposed method outperforms the variant estimators for most cases. For the ablation study on smoothness loss, the contact state is set the same as the proposed method. In the case without smoothness loss, the estimation accuracy decreases drastically. We also find that

TABLE III

ABLATION STUDY: AVERAGE STANDARD DEVIATION OF LINEAR VELOCITY ERROR ON FLAT TERRAIN EXPERIMENT

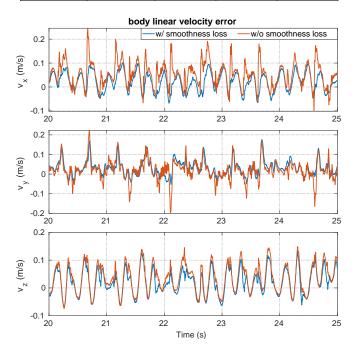| Terrain	Method	v_x	v_y	v_z
Flat	w/ L_{sm}	**0.0472**	**0.0386**	**0.0435**
	w/o L_{sm}	0.0540	0.0466	0.0499

Fig. 5. We compare the body linear velocity error on the *flat* experiment for each ablation study.

the error's standard deviation increases in Table III, and the network output appears to have more fluctuation and peaks in Figure 5. For the ablation study on domain randomization, applying domain randomization degenerates the estimation performance for most cases across different terrains. This implies that domain randomization might exacerbate the sim-to-real gap for our estimation framework, unlike the usual sim-to-real transfer in reinforcement learning-based controllers.

We further study the efficacy of early stopping and the effect of applying smoothness loss to contact prediction as a substitute for our velocity smoothing. To this end, we use the **IEKF with NMN $\hat{c}_t$** method to show how each part affects the contact prediction performance. The following L_{sm}^c is the alternative smoothness loss for $\hat{c}_t$, which is used instead of L_{sm} for training.

$$L_{sm}^c = \frac{1}{N}(||_b\hat{\mathbf{c}}_t -_b \hat{\mathbf{c}}_{t-1}||^2 + \frac{1}{2}||_b\hat{\mathbf{c}}_t - 2_b\hat{\mathbf{c}}_{t-1} +_b \hat{\mathbf{c}}_{t-2}||^2) \quad (14)$$

The results in Table IV show that the performance deteriorates considerably, when L_{sm}^c is used or as the training iteration extends. This phenomenon becomes clear upon viewing Figure 6. For the yellow line, which undergoes 1000 iterations without L_{sm}^c, the contact probability appears noisier than the red line. This noisy result is likely attributed to the

TABLE IV

ABLATION STUDY: IMPACT OF SMOOTHNESS LOSS AND EARLY STOPPING ON FLAT TERRAIN EXPERIMENT

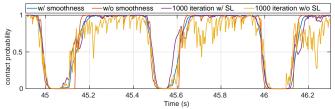| Terrain	iteration	Method	ATE (pos)	ATE (vel)	ATE (ori)	RE (pos)	RE (vel)	RE (ori)
Flat	200	w/o L_{sm}^c	**0.3585**	**0.0791**	**0.0824**	**0.2418**	**0.1035**	**0.0397**
		w/ L_{sm}^c	0.6404	0.0803	0.0859	0.2700	0.1049	0.0398
	1000	w/o L_{sm}^c	1.3755	0.1184	0.0935	0.3872	0.1401	0.0404
		w/ L_{sm}^c	1.1102	0.1037	0.1014	0.3625	0.1280	0.0405

Fig. 6. We compare the learned-contact probability that processes it through the LPF on the *flat* experiment for each ablation study.

NMN overfitting to simulation data, amplifying the sim-to-real gap. While actual contact states change quickly, the L_{sm}^c impedes the rate of change in contact probability, leading to an uptick in false positive contact. Prolonged training accentuates this gradual shift in contact probability, thereby diminishing estimation accuracy.

IV. CONCLUSION AND FUTURE WORK

In this study, we develop a state estimation framework that integrates a neural measurement network (NMN) with an invariant extended Kalman filter (IEKF) to make use of model-based and data-driven state estimators. NMN is composed of GRU-MLP and uses only proprioceptive sensors. The output of NMN is the contact probability and body linear velocity, which are used in the IEKF correction step. The proposed body linear velocity measurement model is suitable for the right-invariant observation form. Unlike previous studies, our approach does not use real-world data and relies solely on simulation data. Therefore, the sim-to-real gap needs to be reduced and to solve this, we analyze the efficacy of existing learning techniques, such as smoothness loss, domain randomization, and early stopping.

We collect data on various terrains with a quadrupedal robot Raibo and validate the estimator performance through Vicon data. These terrains include those with unstable and unclear contact conditions. Our method demonstrate its effectiveness by outperforming model-based state estimator. Additionally, the linear velocity measurement model obtained through learning can be integrated into not only IEKF but also other types of estimators. Therefore, our research results will be helpful in the development of state estimators used in many fields beyond legged robots.

ACKNOWLEDGMENT

References

[1] S. Hong, Y. Um, J. Park, and H.-W. Park, "Agile and versatile climbing on ferromagnetic surfaces with a quadrupedal robot," *Science Robotics*, vol. 7, no. 73, p. eadd1017, 2022.

[2] Y. Kim *et al.*, "Not only rewards but also constraints: Applications on legged robot locomotion," *arXiv preprint arXiv:2308.12517*, 2023.

[3] S. Choi *et al.*, "Learning quadrupedal locomotion on deformable terrain," *Science Robotics*, vol. 8, no. 74, p. eade2256, 2023.

[4] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind bipedal stair traversal via sim-to-real reinforcement learning," *arXiv preprint arXiv:2105.08328*, 2021.

[5] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.

[6] D. Wisth, M. Camurri, and M. Fallon, "Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 309–326, 2022.

[7] M. Bloesch *et al.*, "State estimation for legged robots: Consistent fusion of leg kinematics and imu," in *Robotics: science and systems*, vol. 17. MIT Press, 2012, pp. 17–24.

[8] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart, "State estimation for legged robots on unstable and slippery terrain," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 6058–6064.

[9] R. Hartley *et al.*, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4422–4429.

[10] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[11] J. Sola, "Quaternion kinematics for the error-state kalman filter," *arXiv preprint arXiv:1711.02508*, 2017.

[12] J.-H. Kim *et al.*, "Legged robot state estimation with dynamic contact event information," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6733–6740, 2021.

[13] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[14] S. Herath, H. Yan, and Y. Furukawa, "Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3146–3152.

[15] M. Brossard, S. Bonnabel, and A. Barrau, "Denoising imu gyroscopes with deep learning for open-loop attitude estimation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4796–4803, 2020.

[16] M. Zhang, M. Zhang, Y. Chen, and M. Li, "Imu data processing for inertial aided navigation: A recurrent neural network based approach," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3992–3998.

[17] W. Liu *et al.*, "Tlio: Tight learned inertial odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5653–5660, 2020.

[18] S. Sun, D. Melamed, and K. Kitani, "Idol: Inertial deep orientation-estimation and localization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 7, 2021, pp. 6128–6137.

[19] M. Brossard, A. Barrau, and S. Bonnabel, "Ai-imu dead-reckoning," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.

[20] Y. Jin, W.-A. Zhang, H. Sun, and L. Yu, "Learning-aided inertial odometry with nonlinear state estimator on manifold," *IEEE Transactions on Intelligent Transportation Systems*, 2023.

[21] K. Zhang *et al.*, "Dido: Deep inertial quadrotor dynamical odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9083–9090, 2022.

[22] G. Cioffi, L. Bauersfeld, E. Kaufmann, and D. Scaramuzza, "Learned inertial odometry for autonomous drone racing," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2684–2691, 2023.

[23] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon, "Learning inertial odometry for dynamic legged robot state estimation," in *Conference on robot learning*. PMLR, 2022, pp. 1575–1584.

[24] T.-Y. Lin, R. Zhang, J. Yu, and M. Ghaffari, "Legged robot state estimation using invariant kalman filtering and learned contact events," *arXiv preprint arXiv:2106.15713*, 2021.

[25] A. Antonini, W. Guerra, V. Murali, T. Sayre-McCord, and S. Karaman, "The blackbird dataset: A large-scale dataset for uav perception in aggressive flight," in *International Symposium on Experimental Robotics*. Springer, 2018, pp. 130–139.

[26] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[27] K. Cho *et al.*, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[28] J. Hwangbo, J. Lee, and M. Hutter, "Per-contact iteration method for solving contact dynamics," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 895–902, 2018.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

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

[31] S. Mysore, B. Mabsout, R. Mancuso, and K. Saenko, "Regularizing action policies for smooth control with reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1810–1816.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[33] A. Barrau and S. Bonnabel, "The invariant extended kalman filter as a stable observer," *IEEE Transactions on Automatic Control*, vol. 62, no. 4, pp. 1797–1812, 2016.

[34] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7244–7251.