Proprioceptive Invariant Robot State Estimation

Tzu-Yuan Lin, Tingjun Li, Wenzhe Tong, Maani Ghaffari

arXiv:2311.04320v2 [cs.RO] 20 Feb 2024

***Abstract*—This paper reports on developing a real-time invariant proprioceptive robot state estimation framework called DRIFT. A didactic introduction to invariant Kalman filtering is provided to make this cutting-edge symmetry-preserving approach accessible to a broader range of robotics applications. Furthermore, this work dives into the development of a proprioceptive state estimation framework for dead reckoning that only consumes data from an onboard inertial measurement unit and kinematics of the robot, with two optional modules, a contact estimator and a gyro filter for low-cost robots, enabling a significant capability on a variety of robotics platforms to track the robot's state over long trajectories in the absence of perceptual data. Extensive real-world experiments using a legged robot, an indoor wheeled robot, a field robot, and a full-size vehicle, as well as simulation results with a marine robot, are provided to understand the limits of DRIFT.**

***Index Terms*—State estimation, symmetry-preserving observer, invariant Kalman filtering, proprioceptive odometry, dead reckoning, field robotics.**

I. Introduction

Autonomous robots can greatly benefit humanity by taking over dangerous and tedious jobs, such as extraplanetary exploration, rescue missions in disaster scenes, warehouse logistics, and daily home assistance for elders [1]. In order to accomplish the above tasks, autonomous robots must be able to navigate reliably in various environments and maintain stability. Modern motion planning and control algorithms rely heavily on accurate estimations of the robot's states, i.e., orientation, position, and velocity. Subsequently, accurate state estimation is a key capability in robot autonomy.

To handle noisy and biased data, the fusion of different sensory data is necessary and is the standard practice in robotics [2]. Exteroceptive sensors such as cameras and LiDARs provide perceptual information that can assist with global localization and navigation. However, these types of sensors often operate at a low update rate (10-30 Hz) and are sensitive to perceptually degraded scenarios such as illumination changes. On the other hand, proprioceptive sensors such as Inertial Measurement Units (IMU) and joint encoders operate regardless of external factors such as lighting conditions and often at high frequencies (100-1000 Hz or higher). The high-frequency information is often important for agile navigation in challenging terrains. In addition, being agnostic to illumination changes, proprioceptive sensors can offer robust sensing modalities in extreme or featureless environments such as dense fog or sand storms. As such, designing an accurate and real-time proprioceptive state estimation algorithm is desirable.

Funding for M. Ghaffari was in part provided by NSF Award No. 2118818. This work was also partly supported by the Amazon Research Awards. *(Corresponding author: Tzu-Yuan Lin.)*

The authors are with the University of Michigan, Ann Arbor, MI 48109, USA. {tzuyuan, tingjunl, wenzhet, maanigj}@umich.edu.

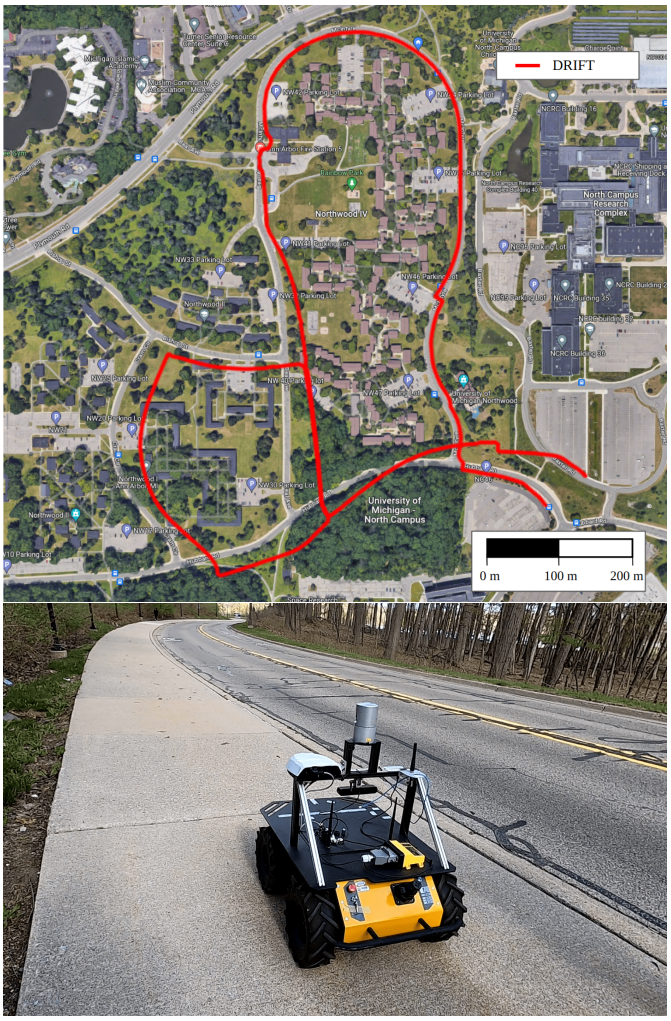

Fig. 1. Estimated trajectory from DRIFT overlapped with the satellite image at the University of Michigan North Campus. A Clearpath Robotics Husky robot was driven on the sidewalk for 55 minutes, with a total path of around 3 kilometers. Consuming proprioceptive measurements only, DRIFT can produce highly accurate estimations for long-horizon operations. This experiment demonstrates the potential of DRIFT to be a reliable odometry system in perceptually degraded situations.

Contrary to Simultaneous Localization and Mapping (SLAM) systems, which focus on achieving globally consistent estimations, this proprioceptive estimator aims at obtaining high-accuracy local estimates without relying on perceptual inputs, serving as a reliable state tracking source for perception and control systems in visually challenging environments [3–6].

To develop proprioceptive state estimators, one often deals with nonlinear robot dynamics and measurement models. The extended Kalman Filter (EKF) [2, 7–9] can be easily modified and has been implemented on various robotic platforms [10–12]. To address the linearization error induced in the EKF framework, many researchers have looked into leveraging

symmetry [13] in the filtering framework. In particular, Barrau and Bonnabel [14, 15] developed an invariant observer, coined Invariant extended Kalman Filter (InEKF), by modeling the state evolution on matrix Lie Groups and tracking the error on the corresponding Lie algebra. The theory of the invariant observer design is founded on estimation error being invariant under the action of a matrix Lie Group. Precisely saying, they show that if the propagation model satisfies the group affine property [14], it induces a log-linear property of the error, i.e., the nonlinear error can be exactly recovered from the time-varying linear differential equation. The InEKF has been shown to achieve better convergence and consistency in many applications, including SLAM [16–18], legged robot state estimation [13, 19–21], visual-inertial odometry [22, 23], underwater navigation [24] and aerial robot navigation [25, 26].

Current implementations of the InEKF remain tailored for specific applications, and modifying existing libraries for different robotics applications remains cumbersome. In this work, we propose *DRIFT: Dead Reckoning In Field Time*, a real-time symmetry-preserving state estimation library that is directly applicable to various robotic platforms, including legged robots, indoor and outdoor wheeled robots, full-size vehicles, and marine robots.

A. Contributions

This work evolves from our prior work [27], in which we developed the contact estimation module for legged robot state estimation. In this work, we propose a unified framework that works for various robotic platforms. We additionally propose the gyro filter module for low-cost robots. Furthermore, we release our new real-time software, which supports various robots by default and allows easy expansion with new modalities. Lastly, we conduct extensive field experiments to assess the limits of the proposed framework on a variety of platforms, including a full-size offroad vehicle.

In particular, this work has the following contributions.

i. An open-source symmetry-preserving robot state estimation library that works in real-time on a variety of robotic platforms. The software can be found in https://github.com/UMich-CURLY/drift.
ii. A didactic introduction to invariant Kalman filtering.
iii. A proprioceptive state estimation framework for dead reckoning that only consumes data from an onboard Inertial Measurement Unit (IMU) and kinematics for the robot, with two optional modules, a contact estimator and a gyro filter for low-cost robots.
iv. Real-world experiments on legged robots, indoor and outdoor wheeled robots, full-size vehicles, and simulation results on marine robots to verify the proposed framework.

B. Outline

The remainder of this article is organized as follows. Sec. II reviews the literature on different state estimation methods. Sec. III provides some necessary backgrounds for developing an invariant filtering algorithm. Then, a didactic introduction to the invariant Kalman filtering is laid out in Sec. IV. Sec. V details DRIFT's framework and methodology. Sec. VI presents extensive evaluations using different robotic platforms. Limitations and future work ideas are discussed in Sec. VII, and Sec. VIII concludes the article.

II. Literature Review

A. Symmetry in Robot State Estimation

Symmetry in Lie Groups has been explored since the 1970s [28–31]. Despite some early efforts on state estimation using Lie Groups [32, 33], the standard extended Kalman filter remained the mainstream method [2, 7, 9] for decades due to its simplicity. With the standard EKF, the non-linear state dynamics and the measurement model are repeatedly linearized using the Taylor series at the current estimate [2, 34]. Although intuitive and easy to implement, the standard EKF can suffer from degrading performance when the dynamics become highly nonlinear, and bad initialization of states can cause the filter to diverge. In addition, because the linearization is evaluated at the current estimate, unobservable states can lead to incorrect linearization, causing spurious correlations.

The standard EKF evolves the state on Euclidean spaces; however, in robotics applications, the state often evolves on a manifold. Alternatively, the state evolution can be modeled using Lie groups, such as quaternion [35, 36] and $\mathrm{SO}(3)$ [37, 38]. One of the early symmetry-preserving observers was proposed by Aghannan and Rouchon [39], where they designed an invariant observer for Lagrangian systems that is invariant under changes of the configuration coordinate. While inspiring, this method requires heavy modeling of the Lagrangian dynamics and might not be easily transferable to complex robot dynamics.

Contrary to the standard EKF, which tracks the state directly, the error-state (or indirect) extended Kalman Filter (ErEKF) [40–43] tracks the state errors and their evolution. By tracking the error, the dynamics in ErEKF become linear under small error assumptions [40]. However, despite the near-linear dynamics, the ErEKF still parameterizes the state in Euclidean spaces. To be more specific, the attitude is often represented using Euler angles. These representations contain singularities (i.e., do not cover the entire rotation manifold), and the filter can consequently be trapped in the famous Gimbal lock. The multiplicative (or quaternion) EKF (MEKF) [41, 43, 44] addresses this problem by tracking the rotation using quaternions and updating the state using multiplicative actions. The orientation error is modeled as a 3-vector on the tangent space of $\mathrm{SO}(3)$ to maintain the nonsingular covariance [41]. The MEKF has been successfully implemented on many platforms, including aerial vehicles [45–47], legged robots [48, 49], and marine robots [50]. Nevertheless, the linearized error propagation of the MEKF still depends on the estimated states, which can degrade the filter performance.

In invariant observer design, the estimated error is invariant under the action of a matrix Lie group [51]. This resulted in developing the InEKF [14, 15, 52, 53] where the state is represented using a matrix Lie group, and the error is tracked in the corresponding Lie algebra. If the system satisfies the "group-affine" property, the estimation error follows a "log-linear" dynamics in the Lie algebra [14, 15, 53]. Namely,

the system linearization is independent of the state estimates. Because of this invariance property, the InEKF can achieve stronger convergence on many state estimation applications, including SLAM [16–18], legged robot [19–21, 54], visual-inertial odometry [22, 23], underwater robots [24] and aerial vehicles [25, 26, 55]. In addition to filtering, some researchers also investigated the potential of symmetry-preserving structures in optimization-based smoothing frameworks [56–59]. The invariant smoother framework can perform better than InEKF at the cost of higher computational time [56, 58].

More recently, a generalization of the InEKF framework, the Equivariant Filter (EqF), has been proposed by van Goor et al. [60] and Mahony and Trumpf [61]. In EqF, the state evolves on a homogeneous space (smooth manifolds with transitive Lie group actions). By finding an appropriate lift, the system dynamics can be realized on symmetry Lie groups. The correction, on the other hand, is derived by linearizing the error kinematics with respect to a fixed origin on the homogeneous space [60–62]. Because the EqF does not require the system to be modeled explicitly on Lie groups, it relaxes the group-affine constraint posed in the InEKF and works with any system with equivariant properties. In fact, it can be reduced to the InEKF when the corresponding state evolves on matrix Lie groups and satisfies the group-affine property [61, 62]. Moreover, since the state is modeled on homogeneous spaces, EqF opens up the opportunity to estimate the input bias without breaking the symmetry of the system [63, 64], which is not feasible in the invariant framework [15, 19]. The EqF demonstrates great potential in different applications [65–67] and can be integrated into DRIFT in the future.

B. Legged Robots

The controller of legged robots often requires accurate and high-frequency state estimations to maintain the robot's stability. This can be achieved by fusing multiple onboard sensors such as IMU, joint encoders, and cameras.

1) State Estimation: One simple way to obtain the state of a legged robot is through leg kinematics, known as leg odometry [68]. By assuming the contact foot to be static in the world frame, one can infer the robot's state using forward kinematic functions and encoder measurements. Leg odometry is simple and has been successfully implemented on many early legged robotic platforms, including CMU Amhler [68], RHex hexapod [69], and MARLO [70]. However, it often performs poorly due to noisy encoder data, inaccuracies in kinematic modeling, and foot slip [68]. This results in noisy velocity and unbounded position and orientation estimates. To address this, Lin et al. [69] used prior terrain information and three non-colinear contact feet to estimate the instantaneous robot's pose through kinematics. Although achieving better accuracy, this algorithm cannot operate when two or more feet are in the aerial phase. As a result, it significantly hampers any agile movements of the robot.

Modern legged robots are often equipped with extra sensors like IMU, camera, and LiDAR. Lin et al. [71] fused leg kinematics with IMU measurements using an EKF for the RHex hexapod. They showed that by fusing the two measurements, the algorithm outperforms methods that only use one modality in isolation. In addition to IMU and leg odometry, researchers also explored fusing different sensor modalities within an EKF framework, including fusing leg odometry with a magnetometer and GPS [12], and leg odometry with optical flow [72]. By fusing multiple sensor measurements, the above methods improve the estimation accuracy of pure leg odometry. However, they still require heavy modeling of the robot's dynamics, which can be cumbersome and inaccurate.

Bloesch et al. [73] proposed observability-constrained ErEKF to fuse IMU and kinematic measurements. In this framework, the process model does not depend on the robot or the gait motion. Instead, the readings from an inertial sensor are integrated to obtain the prediction (also known as the IMU strapdown modeling). Hence, the need for complicated dynamic modeling is circumvented [40]. Another innovation of this work is the state augmentation of the contact foot position. This augmented position serves as a fixed reference position in the world frame. Therefore, the mismatch between new kinematic measurements and the predicted contact state can then be used as the correction model. IMU strapdown has become a common approach for modern state estimators [3, 48, 74, 75]. Rotella et al. [49] includes the foot orientation measurements from forward kinematics in an ErEKF for humanoid robots. However, this method only works when the foot does not rotate during the contact phase.

In the context of optimization-based methods instead of filtering, Hartley et al. [76] proposed a factor graph formulation to fuse leg odometry with IMU measurements on a bipedal Cassie robot. They introduced new forward kinematics and pre-integrated contact factors to include additional constraints posed by the non-slip contact assumption. Hartley et al. [3] further incorporated visual information from semi-direct visual odometry (SVO) [77] to form a visual-inertial-legged smoother. Instead of using SVO as an additional module, Wisth et al. [78] proposed VILENS (Visual Inertial LEgged Navigation System), which tightly integrates visual features from an RGB-D camera into the cost function. VILENS is implemented on an ANYmal robot and showed improved performance over previous methods. Built upon VILENS, the same group introduced the foot velocity integration factor in [79]. By integrating the foot velocity instead of position, a slow-varying velocity bias can be inferred from vision to account for foot slip events. Kim et al. [80] exploited the body velocity estimation from a camera and dropped the need for the contact estimation module. Their method was implemented on an MIT mini cheetah and shown to achieve superior performance on slippery terrains. When only proprioceptive sensors were presented, Kim et al. [58] proposed a slip-rejection mechanism in addition to a proprioceptive smoother framework using an MIT mini cheetah. This method improved the robustness of state estimation even in the absence of camera information. Smoothing approaches have the privilege of leveraging past information during optimization. However, this comes at the cost of higher computational efficiency than filtering, especially during long-horizon experiments, where the past factors could grow unboundedly. Moreover, the above methods did not take advantage of the existing symmetry structure present in the problem, which improves the performance

and consistency of the estimator.

Hartley et al. [19] introduced the InEKF to the legged robot community by designing a contact-aided InEKF filtering. Similar to the work of Bloesch et al. [73], the IMU strapdown model was used for propagation, and the contact states were augmented for correction. However, in this work, the states were tracked on the Lie group $\mathrm{SE}_k(3)$ [53], and the error dynamics evolved in the corresponding Lie algebra. Hartley et al. [19] showed that the contact-inertial process model satisfies the group-affine property [15], and the resulting linearization matrix is independent of the estimated state. The contact-aided InEKF demonstrated state-of-the-art performance on a bipedal Cassie robot and has since been modified for different applications [20, 21, 54, 81].

Existing open-sourced libraries legged robot state estimation, include Pronto proposed by Camurri et al. [10], EKF-based framework for both exteroceptive and proprioceptive sensor fusion. The framework supports multiple bipedal and quadruped robots. However, Pronto is based on the traditional EKF and the state-dependent linearization could lead to sub-optimal results. Another example is Cerberus proposed by Yang et al. [82], which is a factor graph framework for visual-inertial-leg odometry. One key feature of Cerberus is the ability to estimate kinematic parameters online. However, Cerberus relies on visual information for the kinematic parameters estimation, which can lead to degraded performance in visually challenging environments. Moreover, Cerberus is reported to operate at $20\,\mathrm{Hz}$, which might be insufficient for some control algorithms. In addition to the above two libraries, Hartley et al. [19] also released their source code for contact-aided InEKF for Cassie. In DRIFT, we adopt the software by Hartley et al. [19], and our previous work [20], and enable modular extensions to different modalities.

2) Contact Estimation: Reliable contact estimation that accurately captures the zero velocity events on foot is of crucial importance for legged robot state estimators. Model-based approaches segment the touchdown event of robot legs or prosthetic legs by thresholding of the estimated Ground Reaction Force (GRF) from the general equation of motion [83–85]. Although this method can detect touchdown events, the estimated GRF is often noisy and unreliable, especially for lightweight robots. De Luca et al. [86], Haddadin et al. [87] proposed a Generalized Momentum (GM) method for detecting contact events on robot manipulators. This GM-based method is, in fact, a filtered version of the work of Focchi et al. [83]. Although GM-based methods mitigate the noise problem in GRF estimation, an empirical threshold on the cut-off frequency is still required.

Hwangbo et al. [88] used a probabilistic representation of the contact state and a Hidden Markov Model (HMM) to fuse the dynamics and kinematics for contact estimation by adopting a Monte-Carlo sampling algorithm to compute the transition model and verifying against GM-based methods. Jenelten et al. [89] expanded the HMM method and focused on slippage detection. They demonstrate ANYmal [90], a quadruped robot, walking stably on slippery ground. The above two methods aim to detect contact as early as possible for the controller to maintain stability; however, we aim to detect contact intervals for state estimation on various terrains.

Bledt et al. [91] leveraged the GM-based methods and the probabilistic representation of contact states by using a Kalman filter to fuse the gait phase scheduler information from the controller with the GRF and demonstrate that estimated contacts can assist the controller in reducing the bouncing event upon touchdown. However, this method assumes the leg phase to be periodic as it uses the gait scheduler information in the prediction step of the Kalman filter. It could experience a loss in performance when the phase is heavily violated as the robot interacts with uneven terrains.

Camurri et al. [92] used logistic regression to learn the GRF threshold for contact detection. This work compares against heuristic-based thresholding on GRF using a base state estimator. The result shows that the logistic regression classifier can double the performance of the state estimator. However, compared to deep learning methods, the performance of a logistic regression classifier gets saturated as the amount of data increases [93, 94]. Moreover, this method requires a specific training procedure for different gait, loading conditions of the robot, and individual terrain properties.

Rotella et al. [95] used a fuzzy C-means clustering for the probability of contacts in all six end-effector degrees of freedom. They integrate the contact estimator with a base state estimator and show their approach performs considerably better than implementations purely based on measured normal force. However, this method assumes contact wrench sensors and additional IMU are available at each end-effector. Furthermore, this method was only tested in simulation. Its performance on real robots remains unknown. Piperakis et al. [96] proposed an unsupervised learning method for humanoid gait phase estimation by employing Gaussian Mixture Models (GMMs) for clustering and comparing to the ground-truth data and leg odometry. However, this work also assumes the availability of wrench/force sensors at each end-effector, and the clustering result is affected by the gait and data density.

The above methods either assume the availability of wrench/force sensors or are restricted by the nature of simple regression and are thus unable to generalize to different scenarios. In contrast, our work proposes a multi-modal deep learning-based contact estimator that does not require contact sensors and can generalize well to different gaits and terrain properties. Moreover, as more data becomes available, the network performance can be improved.

C. Wheeled Robots

Wheeled robots and autonomous vehicles state estimation problems often treat odometer measurements and dynamics constraints as given, which increases the accuracy and robustness of the localization systems [97, 98]. However, such methods require high computational power to process image flow. The IMU-based algorithms are more universal and approachable for wheeled robots, which can contribute to emergency events detection such as collision and slippage [99, 100].

For outdoor wheeled robots, GPS plays an important role in exteroceptive localization which uses multiple satellites to obtain the robot's position in the earth coordinates. Never-

theless, the GPS accuracy is limited by radio signal transmission time, and the slow-down effect of water vapor and other particles in the atmosphere counts for the propagation delay [101]. The occlusions and reflections of and between the buildings, namely multipath fading, will also reduce the GPS localization accuracy. To overcome these limitations, beacon-based methods, such as RFID [102, 103], QR-code [104, 105], WiFi or Bluetooth [106–109], and UWB [110, 111], are used for urban and indoor environments localization.

In terms of proprioceptive state estimation for four-wheeled chassis robots, typically autonomous vehicles, the measurements are IMU readings, steering wheel angle, and wheel angular velocity, which is the most common sensor configuration for vehicle velocity estimation [112, 113]. Nonlinear observer (NLO), nonlinear unknown input observer (NUIO), and reduced nonlinear observer (RNLO) are proposed for vehicle lateral and longitudinal velocity estimation by Imsland et al. [112, 114], Guo et al. [115]. Wenzel et al. [116] invented the dual extended Kalman filter (DEKF) for combined estimation of vehicle states and vehicle parameters.

Mobile robot filtering-based localization performance can be increased by augmenting uncertain parameters into the state [117]. Nevertheless, with model-based filters such as the Kalman filter, the performance highly depends on how accurately the stochastic model can capture the underlying physical process. Kwon et al. [118] proposed a combined Kalman filter-perturbation estimator (CKF) for wheeled robots, which does not require any uncertainty model except for the noise statistics. The perturbation estimator generates equivalent perturbations based on the nominal state equation and action model of the robot, which is simultaneously corrected by the estimates of perturbation. The perturbation estimator's integral control module leads to a decrease in the localization error.

A dead-reckoning system has an unavoidable drift over time because, first, the integration of sensors' random walk and, secondly, inaccuracies of the system modeling for the action and observation models. Welte et al. [119] presented a dead-reckoning model that can fuse all common sensors and then merge the redundant ones, increasing the robustness compared with the traditional Zero Velocity Update (ZUPT) method. The work also proposed a method to compensate for the systematic sensor and observation model errors, which relies on the Rauch-Tung-Striebel smoothing method to estimate the robot state and further calibrate the system model parameters.

D. Marine Robots

Underwater environments are often featureless with limited visibility, and objects underwater are frequently subject to unexpected external forces from ocean currents or surges. This poses extra challenges for Autonomous Underwater Vehicles (AUVs) state estimation. To make matters worse, some common sensors for ground robots, such as joint encoders and GPS, cannot be directly applied to AUVs. However, many widely used modern underwater localization systems depend on accurate dead reckoning solutions [120]. *We position dead reckoning algorithms such as DRIFT as a submodule of a more complex localization system.* Comprehensive reviews on underwater state estimators can be found in [120–122].

There are several specialized sensors for underwater navigation. Because the water pressure can roughly translate to the depth below sea level, a pressure sensor can be used to obtain depth measurements. In addition to pressure sensing, acoustic sensors such as Doppler Velocity Logs (DVL) [123] also play a big role in underwater navigation. A DVL can provide ground-referenced velocity measurements. This is achieved by emitting directional acoustic beams to the seabed and using the Doppler effect on the reflected acoustic signals. Unlike the above sensors that are mounted directly on the robot, transponder systems [124], such as a Long Baseline system (LBL), use floating beacons to provide positioning measurements from triangulation. Although transponder systems allow direct positioning measurements, they require additional beacons to be deployed, limiting the operation range of AUVs.

EKF has been implemented on multiple AUVs due to its simplicity [11, 125, 126]. Karras et al. [127] proposed a Dual Unscented Kalman Filter (DUKF) framework to fuse measurements from a laser sensor and an IMU. The filter jointly estimated the 6D pose of the robot and its dynamic parameters. The DUKF was still sensitive to initialization, and a specific closed-loop control scheme was required during initialization. Instead of online optimization of the dynamic parameters, Hegrenaes and Hallingstad [128] formulated a more accurate vehicle model in a model-aided EKF framework. The kinetic model was integrated to provide velocity measurements for filter correction. Sea current velocity was treated as a slowly varying bias and was estimated along with the robot's state. The method was evaluated using a HUGIN 4500 AUV on various field tests. The IMU strapdown modeling was also adopted for marine robotic applications to circumvent complex modeling of the vehicle. Using an ErEKF framework, Miller et al. [129] fused multiple sensors, including an IMU, an LBL, a DVL, a pressure sensor, and an attitude sensor for UAV state estimation. Additional calibration parameters, such as the speed of sound in seawater, were also estimated to reduce systematic errors. This framework was shown to be robust to sporadic sensor failures. However, a prolonged failure of the DVL or LBL can degrade the performance, and the requirement of LBL still limits the operation range.

Potokar et al. [24] used an InEKF to estimate the 6D pose of an AUV. The velocity readings from a DVL, and depth measurements from a pressure sensor were fused with the IMU strapdown model. Because the depth measurements do not obey the left-invariant formulation, which we will discuss later in Sec. III, pseudo measurements with infinity covariances were introduced to allow the incorporation of such measurements. This framework was evaluated in a simulation environment and was shown to have better convergence over the MEKF. The algorithm was implemented in Python and was demonstrated to have real-time performance. DRIFT is a unified state estimator that directly supports multiple robots. Modular implementation in C++ enables faster computation on resourced-constrained devices and allows easy expansion with new modalities. In addition, for convenience, we provide a ROS wrapper for real-time communication on robots.

III. Preliminaries and notation

Matrix Lie groups [2, 130, 131] provide natural (exponential) coordinates that exploits symmetries of the space [61, 132–136]. State estimation is the problem of determining a robot's position, orientation, and velocity that are vital for robot control [2]. An interesting class of state estimators that can be run at high frequency, e.g., 2 kHz, are based on InEKF [14, 15, 19, 53]. The theory of invariant observer design is based on the estimation error being invariant under the action of a matrix Lie group. The fundamental result is that by correct parametrization of the error variable, a wide range of nonlinear problems can lead to log-linear error dynamics [14, 51, 53].

Consider a deterministic Linear Time-Invariant (LTI) process model $\dot{x} = Ax + Bu$. Let $\bar{x}$ be an estimate of x, i.e., $\dot{\bar{x}} = A\bar{x} + Bu$. Define the error $e := x - \bar{x}$. Then $\dot{e} = \dot{x} - \dot{\bar{x}} = A(x - \bar{x}) = Ae$ is an autonomous differential equation. Given an initial condition $e(0) = e_0$, we can solve for the error at any time via $e(t) = \exp(At)e_0$. In other words, error propagation is independent of the system trajectory, i.e., state estimate. The Invariant EKF [14] generalizes this observation to linear-exponential systems, e.g., rigid body systems, using the framework of matrix Lie groups.

A. Process Dynamics on Lie Groups

A process dynamics evolving on the Lie group $\mathcal{G}$, for state $X_t \in \mathcal{G}$, is

$$\frac{d}{dt}X_t = f_{u_t}(X_t). \tag{1}$$

$\bar{X}_t$ denotes an estimate of the state. The state estimation error is defined using right or left multiplication of X_t^{-1}.

Definition 1 (Left and Right Invariant Error)**.** *The right- and left-invariant errors between two trajectories X_t and $\bar{X}_t$ are:*

$$\begin{aligned} \eta_t^r &= \bar{X}_t X_t^{-1} = (\bar{X}_t L)(X_t L)^{-1} \quad \textit{(Right-Invariant)} \\ \eta_t^l &= X_t^{-1}\bar{X}_t = (L\bar{X}_t)^{-1}(LX_t), \quad \textit{(Left-Invariant)} \end{aligned} \tag{2}$$

where $L \in \mathcal{G}$ is an arbitrary element of the group.

Theorem 1 (Autonomous Error Dynamics [14])**.** *A system is group affine if the dynamics, $f_{u_t}(\cdot)$, satisfies:*

$$f_{u_t}(X_1X_2) = f_{u_t}(X_1)X_2 + X_1 f_{u_t}(X_2) - X_1 f_{u_t}(I)X_2 \tag{3}$$

for all $t > 0$ and $X_1, X_2 \in \mathcal{G}$. Furthermore, if this condition is satisfied, the right- and left-invariant error dynamics are trajectory-independent and satisfy:

$$\begin{aligned} &\frac{d}{dt}\eta_t^r = g_{u_t}(\eta_t^r) \quad \textit{where} \quad g_{u_t}(\eta^r) = f_{u_t}(\eta^r) - \eta^r f_{u_t}(I), \\ &\frac{d}{dt}\eta_t^l = g_{u_t}(\eta_t^l) \quad \textit{where} \quad g_{u_t}(\eta^l) = f_{u_t}(\eta^l) - f_{u_t}(I)\eta^l. \end{aligned} \tag{4}$$

I denotes the group identity element (I_n for $n \times n$ identity matrix). Define A_t to be a $\dim\mathfrak{g} \times \dim\mathfrak{g}$ matrix, where $\mathfrak{g}$ is the Lie algebra (tangent space at the identity) of $\mathcal{G}$, satisfying

$$g_{u_t}(\exp(\xi)) := (A_t\xi)^\wedge + \mathcal{O}(\|\xi\|^2). \tag{5}$$

For all $t \geq 0$, let $\xi_t \in \mathbb{R}^{\dim\mathfrak{g}}$ be the solution of the linear differential equation $\frac{d}{dt}\xi_t = A_t\xi_t$.

Theorem 2 (Log-Linear Property of the Error [14])**.** *Consider the right-invariant error, η_t, between two trajectories (possibly far apart). For arbitrary initial error $\xi_0 \in \mathbb{R}^{\dim\mathfrak{g}}$, if $\eta_0 = \exp(\xi_0)$, then for all $t \geq 0$, $\eta_t = \exp(\xi_t)$; that is, the nonlinear estimation error η_t can be exactly recovered from the time-varying linear differential equation.*

For an example of the difference with Euler angle parametrization, see Hartley et al. [19, Sec. 4].

B. Associated Noisy System

A noisy process dynamics evolving on the Lie group take the following form.

$$\frac{d}{dt}X_t = f_{u_t}(X_t) + X_t w_t^\wedge,$$

where $w_t^\wedge \in \mathfrak{g}$ is a continuous white noise whose covariance matrix is denoted by Q_t. Then the equivalent noisy error dynamics can be shown to be

$$\begin{aligned} \frac{d}{dt}\eta_t^r &= g_{u_t}(\eta_t^r) - (\bar{X}_t w_t^\wedge \bar{X}_t^{-1})\eta_t^r \\ &= g_{u_t}(\eta_t^r) - (\mathrm{Ad}_{\bar{X}_t} w_t)^\wedge \eta_t^r, \\ \frac{d}{dt}\eta_t^l &= g_{u_t}(\eta_t^l) - w_t^\wedge \eta_t^l, \end{aligned} \tag{6}$$

where for every $X \in \mathcal{G}$, the adjoint action, $\mathrm{Ad}_X : \mathfrak{g} \to \mathfrak{g}$, is a Lie algebra isomorphism that enables change of frames (matrix similarity).

C. Invariant Observation Model

During the correction step of a Kalman filter, the error is updated using incoming sensor measurements. If observations take a particular form, then the linearized observation model and the innovation will also be autonomous. This happens when the measurement, Y_{t_k}, can be written as either

$$\begin{aligned} Y_{t_k} &= X_{t_k} b + V_{t_k} \quad \text{(Left-Invariant Observation)} \quad \text{or} \\ Y_{t_k} &= X_{t_k}^{-1} b + V_{t_k} \quad \text{(Right-Invariant Observation).} \end{aligned} \tag{7}$$

b is a constant vector and V_{t_k} is a vector of Gaussian noise.

IV. Invariant Kalman Filtering

A. Left-Invariant Extended Kalman Filter

We now derive the Left-Invariant Extended Kalman Filter (LI-EKF). In particular, the filter tracks the state mean and covariance as parameters and corrects the error via a "linear" update rule. As we will see, the innovation is in the Lie algebra, and the update rule follows the group operation for integration.

1) Propagation: In the propagation (or prediction) step, the mean can be directly computed using the process model. However, for the covariance propagation, we use the left-invariant log-linear error dynamics. As such, the state's mean

evolves on the group while the covariance is tracked in the Lie algebra.

$$\begin{aligned}
&\frac{d}{dt}\bar{X}_t = f_{u_t}(\bar{X}_t), \quad t_{k-1} \le t < t_k,\\
&\frac{d}{dt}\eta_t^l = g_{u_t}(\eta_t^l) - w_t^\wedge \eta_t^l \implies \frac{d}{dt}\xi_t^l = A_t^l \xi_t^l - w_t,\\
&\frac{d}{dt}P_t^l = A_t^l P_t^l + P_t^l {A_t^l}^\mathsf{T} + Q_t.
\end{aligned} \tag{8}$$

2) Update: In the update (or correction) step, we use $\eta = \exp(\xi) \approx I + \xi^\wedge$ and neglect the higher order terms to derive the linear update error equation as follows.

$$\begin{aligned}
\bar{X}_{t_k}^+ &= \bar{X}_{t_k} \exp\left(L_{t_k}\left(\bar{X}_{t_k}^{-1} Y_{t_k} - b\right)\right),\\
X_{t_k}^{-1}\bar{X}_{t_k}^+ &= X_{t_k}^{-1}\bar{X}_{t_k} \exp\left(L_{t_k}\left(\bar{X}_{t_k}^{-1}(X_{t_k} b + V_{t_k}) - b\right)\right),\\
\eta_{t_k}^{l+} &= \eta_{t_k}^l \exp\left(L_{t_k}\left((\eta_{t_k}^l)^{-1} b - b + \bar{X}_{t_k}^{-1} V_t\right)\right),\\
{\xi_{t_k}^{l+}}^\wedge &= {\xi_{t_k}^l}^\wedge + \left(L_{t_k}\left((I - {\xi_{t_k}^l}^\wedge) b - b + \bar{X}_{t_k}^{-1} V_t\right)\right)^\wedge,\\
\xi_{t_k}^{l+} &= \xi_{t_k}^l + L_{t_k}\left(-{\xi_{t_k}^l}^\wedge b + \bar{X}_{t_k}^{-1} V_t\right).
\end{aligned} \tag{9}$$

To arrange all terms according to the vector form of ξ, define the measurement Jacobian, H, such that $\boxed{H\xi = \xi^\wedge b}$. Then it follows that

$$\begin{aligned}
\bar{X}_{t_k}^+ &= \bar{X}_{t_k} \exp\left(L_{t_k}\left(\bar{X}_{t_k}^{-1} Y_{t_k} - b\right)\right),\\
\xi_{t_k}^{l+} &= (I - L_{t_k} H)\xi_{t_k}^l + L_{t_k}\bar{X}_{t_k}^{-1} V_t,\\
P_{t_k}^{l+} &= (I - L_{t_k} H) P_{t_k}^l (I - L_{t_k} H)^\mathsf{T} + L_{t_k}\bar{N}_k L_{t_k}^\mathsf{T},
\end{aligned} \tag{10}$$

where $\bar{N}_k := \bar{X}_{t_k}^{-1}\,\mathrm{Cov}(V_k)\,\bar{X}_{t_k}^{-\mathsf{T}}$.

3) LI-EKF Result: To summarize, we have the following two steps (as usual for a Kalman filter).

1) LI-EKF Propagation:

$$\begin{aligned}
&\frac{d}{dt}\bar{X}_t = f_{u_t}(\bar{X}_t), \quad t_{k-1} \le t < t_k,\\
&\frac{d}{dt}P_t^l = A_t^l P_t^l + P_t^l {A_t^l}^\mathsf{T} + Q_t.
\end{aligned} \tag{11}$$

2) LI-EKF Update:

$$\begin{aligned}
\bar{X}_{t_k}^+ &= \bar{X}_{t_k} \exp\left(L_{t_k}\left(\bar{X}_{t_k}^{-1} Y_{t_k} - b\right)\right),\\
P_{t_k}^{l+} &= (I - L_{t_k} H) P_{t_k}^l (I - L_{t_k} H)^\mathsf{T} + L_{t_k}\bar{N}_k L_{t_k}^\mathsf{T},
\end{aligned} \tag{12}$$

where

$$L_{t_k} = P_{t_k}^l H^\mathsf{T} S^{-1}, \quad S = H P_{t_k}^l H^\mathsf{T} + \bar{N}_k. \tag{13}$$

Given these equations, once we know A_t^l and H matrices, we can implement the LI-EKF.

B. Right-Invariant Extended Kalman Filter

Next, we derive the Right-Invariant Extended Kalman Filter (RI-EKF) similarly.

1) Propagation: In the propagation step, the mean can be directly computed using the process model. However, for the covariance propagation, we use the right-invariant log-linear error dynamics. As such, the state's mean evolves on the group while the covariance is tracked in the Lie algebra.

$$\begin{aligned}
&\frac{d}{dt}\bar{X}_t = f_{u_t}(\bar{X}_t), \quad t_{k-1} \le t < t_k,\\
&\frac{d}{dt}\eta_t^r = g_{u_t}(\eta_t^r) - (\mathrm{Ad}_{\bar{X}_t} w_t)^\wedge \eta_t^r,\\
&\qquad \implies \frac{d}{dt}\xi_t^r = A_t^r \xi_t^r - \mathrm{Ad}_{\bar{X}_t} w_t,\\
&\frac{d}{dt}P_t^r = A_t^r P_t^r + P_t^r {A_t^r}^\mathsf{T} + \mathrm{Ad}_{\bar{X}_t} Q_t \mathrm{Ad}_{\bar{X}_t}^\mathsf{T}.
\end{aligned} \tag{14}$$

2) Update: We use $\eta = \exp(\xi) \approx I + \xi^\wedge$ and neglect the higher order terms to derive the linear update error equation as follows.

$$\begin{aligned}
\bar{X}_{t_k}^+ &= \exp\left(L_{t_k}\left(\bar{X}_{t_k} Y_{t_k} - b\right)\right)\bar{X}_{t_k},\\
\eta_{t_k}^{r+} &= \exp\left(L_{t_k}\left(\eta_{t_k}^r b - b + \bar{X}_{t_k} V_t\right)\right)\eta_{t_k}^r,\\
{\xi_{t_k}^{r+}}^\wedge &= {\xi_{t_k}^r}^\wedge + \left(L_{t_k}\left((I + {\xi_{t_k}^r}^\wedge) b - b + \bar{X}_{t_k} V_t\right)\right)^\wedge,\\
\xi_{t_k}^{r+} &= \xi_{t_k}^r + L_{t_k}\left({\xi_{t_k}^r}^\wedge b + \bar{X}_{t_k} V_t\right).
\end{aligned} \tag{15}$$

To arrange all terms according to the vector form of ξ, define the measurement Jacobian, H, such that $\boxed{H\xi = -\xi^\wedge b}$. Then it follows that

$$\begin{aligned}
\bar{X}_{t_k}^+ &= \exp\left(L_{t_k}\left(\bar{X}_{t_k} Y_{t_k} - b\right)\right)\bar{X}_{t_k},\\
\xi_{t_k}^{r+} &= (I - L_{t_k} H)\xi_{t_k}^r + L_{t_k}\bar{X}_{t_k} V_t,\\
P_{t_k}^{r+} &= (I - L_{t_k} H) P_{t_k}^r (I - L_{t_k} H)^\mathsf{T} + L_{t_k}\bar{N}_k L_{t_k}^\mathsf{T},
\end{aligned} \tag{16}$$

where $\bar{N}_k := \bar{X}_{t_k}\,\mathrm{Cov}(V_k)\bar{X}_{t_k}^\mathsf{T}$.

3) RI-EKF Result: To summarize, we have the following two steps.

1) RI-EKF Propagation:

$$\begin{aligned}
&\frac{d}{dt}\bar{X}_t = f_{u_t}(\bar{X}_t), \quad t_{k-1} \le t < t_k,\\
&\frac{d}{dt}P_t^r = A_t^r P_t^r + P_t^r {A_t^r}^\mathsf{T} + \mathrm{Ad}_{\bar{X}_t} Q_t \mathrm{Ad}_{\bar{X}_t}^\mathsf{T}.
\end{aligned} \tag{17}$$

2) RI-EKF Update:

$$\begin{aligned}
\bar{X}_{t_k}^+ &= \exp\left(L_{t_k}\left(\bar{X}_{t_k} Y_{t_k} - b\right)\right)\bar{X}_{t_k},\\
P_{t_k}^{r+} &= (I - L_{t_k} H) P_{t_k}^r (I - L_{t_k} H)^\mathsf{T} + L_{t_k}\bar{N}_k L_{t_k}^\mathsf{T},
\end{aligned} \tag{18}$$

where

$$L_{t_k} = P_{t_k}^r H^\mathsf{T} S^{-1}, \quad S = H P_{t_k}^r H^\mathsf{T} + \bar{N}_k. \tag{19}$$

Given these equations, once we know A_t^r and H matrices, we can implement the RI-EKF.

C. Switching Between Left and Right-Invariant Errors

We can switch between the left and right error forms through the use of the adjoint map.

$$\begin{aligned}
\eta_t^r &= \bar{X}_t X_t^{-1} = \bar{X}_t \eta_t^l \bar{X}_t^{-1},\\
\exp(\xi_t^r) &= \bar{X}_t \exp(\xi_t^l)\bar{X}_t^{-1} = \exp(\mathrm{Ad}_{\bar{X}_t}\xi_t^l),\\
\xi_t^r &= \mathrm{Ad}_{\bar{X}_t}\xi_t^l.
\end{aligned} \tag{20}$$

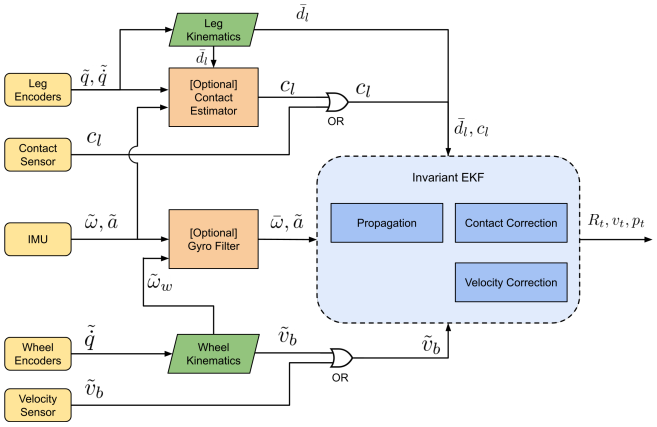

Fig. 2. DRIFT takes measurements from an IMU and encoders as inputs. The angular velocities and linear accelerations are used in the propagation model. The encoder measurements are passed through kinematic functions and applied in the correction model. Two optional modules, a contact estimator and a gyro filter, are provided for low-cost robots to enhance their performance.

Remark 1. *This transformation is exact, which means that we can easily switch between the covariance of the left and right invariant errors using* $P_t^r = \mathrm{Ad}_{\bar{X}_t} P_t^l \mathrm{Ad}_{\bar{X}_t}^\mathsf{T}$.

V. DRIFT: A Symmetry-Preserving Robot State Estimation Library

DRIFT can be used as a standalone C++ program, and we additionally provide an optional Robot Operating System (ROS) wrapper for easy communication between programs. The library can also be easily expanded with new sensor modalities. In addition, we provide two additional modules, a contact estimator and a gyro filter, to enable the proposed framework to be applied to low-cost robotic platforms. Fig. 2 shows the architecture implemented in this work. The proposed framework can be seamlessly extended to other existing and emerging symmetry-preserving filters in the future.

A. State Representation

In most robotics applications, we are interested in tracking the orientation, velocity, and position of the robot with respect to a fixed world frame. We denote $R_t \in \mathrm{SO}(3)$, the rotation matrix that takes the coordinate frame from the body to the world, $v_t \in \mathbb{R}^3$, and $p_t \in \mathbb{R}^3$ as the body velocity and position in the world frame. Following Barrau [53], we can define our state as a direct isometries group $X_t \in \mathrm{SE}_{l+2}(3)$:

$$X_t := \begin{bmatrix} R_t & v_t & p_t & d_{1t} & \cdots & d_{lt} \\ 0_{l+2,3} & & & I_{l+2} & & \end{bmatrix}, \tag{21}$$

where $d_{lt} \in \mathbb{R}^3$ is the augmented vectors, e.g., foot contact position in the world frame for the legged robot or landmark positions. In the minimal setup for wheeled or marine robots, the navigation state simply reduces to $\mathrm{SE}_2(3)$.

B. IMU Propagation

An IMU measures the angular velocity and the linear acceleration in the body frame. By integrating the measurements from an IMU, we can predict the robot's motion, enabling filter propagation without dealing with complicated dynamics [19, 40].

In DRIFT, we model the IMU as corrupted by additive white Gaussian noise.

$$\begin{aligned} \tilde{\omega}_t &= \omega_t + w_t^g, \quad w_t^g \sim \mathcal{GP}(0_{3,1}, \Sigma^g \delta(t-t')), \\ \tilde{a}_t &= a_t + w_t^a, \quad w_t^a \sim \mathcal{GP}(0_{3,1}, \Sigma^a \delta(t-t')), \end{aligned} \tag{22}$$

where, $\mathcal{GP}$ represents a Gaussian process and $\delta(t-t')$ is the Dirac delta function. With this, we define the input u_t to the system as: $u_t := \left[\tilde{\omega}_t^\mathsf{T}, \tilde{a}_t^\mathsf{T}\right]^\mathsf{T}$.

1) Continuous Dynamics: We can write down the continuous dynamic process model:

$$\frac{d}{dt}R_t = R_t(\tilde{\omega}_t - w_t^g)_\times, \frac{d}{dt}v_t = R_t(\tilde{a}_t - w_t^a) + g, \frac{d}{dt}p_t = v_t. \tag{23}$$

Here, $(\cdot)_\times$ denotes a 3×3 skew-symmetric matrix, and g is the known gravity vector.

For legged robots, we assume the foot contact position in the world frame remains constant during the same contact period. As a result, the dynamics is only affected by the rotated white Gaussian noise:

$$\frac{d}{dt}d_t = R_t h_R(\tilde{q}_t)(-w_t^d), \tag{24}$$

where $\tilde{q}_t$ is the encoder measurements, and $h_R(\tilde{q}_t)$ is the orientation of the contact frame in the IMU (body) frame calculated from forward kinematics. For the sake of readability, we derive the propagation model with one contact augmentation in the following derivation. The propagation model for robots without foot contacts can be easily deduced by removing the corresponding column and row for the contact position.

Collecting the above equations into the matrix form and separating the noise term, we can obtain the deterministic dynamic function $f_{u_t}(\cdot)$:

$$\begin{aligned} \frac{d}{dt}X_t &= \begin{bmatrix} R_t(\tilde{\omega}_t)_\times & R_t\tilde{a}_t + g & v_t & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ &\quad - \begin{bmatrix} R_t & v_t & p_t & d_t \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (w_t^g)_\times & w_t^a & 0 & h_R(\tilde{q}_t)(w_t^v) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ &:= f_{u_t}(X_t) - X_t w_t^\wedge. \end{aligned} \tag{25}$$

We can verify that $f_{u_t}(\cdot)$ satisfies (3) and thus is group-affine, and from Theorem 1, the right-invariant error dynamics is given by (4). Using $g_{u_t}(\exp(\xi)) := (A_t\xi)^\wedge + \mathcal{O}(\|\xi\|^2)$ from Theorem 1 and following the derivation from Sec. IV-B, we can obtain the state-independent linearized error dynamics:

$$\frac{d}{dt}\xi_t^r = A_t^r \xi_t^r - \mathrm{Ad}_{\bar{X}_t} w_t, A_t^r = \begin{bmatrix} 0 & 0 & 0 & 0 \\ (g)_\times & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{26}$$

In summary, with $f_{u_t}(\cdot)$, A_t^r, and $Q_t = \mathrm{Cov}(w_t)$, the IMU propagation can then be computed using (17).

2) Integration: In practice, IMU provides discrete samplings of the continuous inputs. We assume the inputs to be constant between two timestamps and perform the propagation by integrating between time t_k and t_{k+1}:

$$\begin{aligned}
\bar{R}_{t_{k+1}} &= \bar{R}_{t_k} \exp(\omega_{t_k} \Delta t) \\
\bar{v}_{t_{k+1}} &= \bar{v}_{t_k} + g\Delta t + \bar{R}_{t_k}(\int_{t_k}^{t_{k+1}} \exp(\omega_{t_k} t) dt) a_{t_k} \\
\bar{p}_{t_{k+1}} &= \bar{p}_{t_k} + \bar{v}_{t_k}\Delta t + \frac{1}{2} g \Delta t^2 \\
&\quad + \bar{R}_{t_k}(\int_{t_k}^{t_{k+1}} \int_{t_k}^{t_\tau} \exp(\omega_{t_k} t) dt d\tau) a_{t_k}
\end{aligned} \tag{27}$$

The exponential map and the integration for $\mathrm{SO}(3)$ has closed-form solutions [19, 43], leading to

$$\begin{aligned}
\bar{R}_{t_{k+1}} &= \bar{R}_{t_k} \Gamma_0(\omega_{t_k} \Delta t) \\
\bar{v}_{t_{k+1}} &= \bar{v}_{t_k} + g\Delta t + \bar{R}_{t_k} \Gamma_1(\omega_{t_k} \Delta t) a_{t_k} \\
\bar{p}_{t_{k+1}} &= \bar{p}_{t_k} + \bar{v}_{t_k}\Delta t + \frac{1}{2} g \Delta t^2 + \bar{R}_{t_k} \Gamma_2(\omega_{t_k} \Delta t) a_{t_k},
\end{aligned} \tag{28}$$

with

$$\begin{aligned}
\Gamma_0(\phi) &= I + \frac{\sin(\|\phi\|)}{\|\phi\|}(\phi)_\times + \frac{1-\cos(\|\phi\|)}{\|\phi\|^2}(\phi)_\times^2 \\
\Gamma_1(\phi) &= I + \frac{1-\cos(\|\phi\|)}{\|\phi\|^2}(\phi)_\times + \frac{\|\phi\| - \sin(\|\phi\|)}{\|\phi\|^3}(\phi)_\times^2 \\
\Gamma_2(\phi) &= \frac{1}{2} I + \frac{\|\phi\| - \sin(\|\phi\|)}{\|\phi\|^3}(\phi)_\times \\
&\quad + \frac{\|\phi\|^2 + 2\cos(\|\phi\|) - 2}{2\|\phi\|^4}(\|\phi\|)_\times^2.
\end{aligned} \tag{29}$$

We can propagate the covariance using a discretized state transition matrix $\Phi^r = \exp(A^r \Delta t)$ and

$$P_{k+1} = \Phi^r P_k \Phi^{r\mathsf{T}} + \mathrm{Ad}_{\bar{X}_k} Q_d \mathrm{Ad}_{\bar{X}_k}^\mathsf{T}. \tag{30}$$

3) Bias Augmentation: In practice, IMU is often corrupted by slow-varying biases. To compensate for this, augmenting the biases as additional states in the system is common. However, no existing matrix Lie group can describe this additional state with the pose without violating the group-affine property [53]. Adding IMU biases breaks the symmetry and makes the linearization matrix dependent on the estimated states [15, 19]. Nevertheless, this imperfect InEKF still outperforms traditional approaches like EKF [19].

To account for the biases, we model the IMU as being corrupted by additive low-frequency signals and white Gaussian noise:

$$\begin{aligned}
\tilde{\omega}_t &= \omega_t + b_t^g + w_t^g, \quad w_t^g \sim \mathcal{GP}(0_{3,1}, \Sigma^g \delta(t-t')), \\
\tilde{a}_t &= a_t + b_t^a + w_t^a, \quad w_t^a \sim \mathcal{GP}(0_{3,1}, \Sigma^a \delta(t-t')).
\end{aligned} \tag{31}$$

The system's state then becomes a tuple of the state defined in (21) and the augmented biases:

$$(X_t, \theta_t) \in \mathrm{SE}_{l+2}(3) \times \mathbb{R}^6, \ \theta_t \coloneqq \begin{bmatrix} b_t^g \\ b_t^a \end{bmatrix}. \tag{32}$$

The new augmented state has a right-invariant error in the form of: $e_t^r = (\bar{X}_t X_t^{-1}, \bar{\theta}_t - \theta_t) \coloneqq (\eta_t, \zeta_t)$. With the bias augmented, we can obtain the continuous system dynamics:

$$\begin{aligned}
\frac{d}{dt} R_t &= R_t(\tilde{\omega}_t - b_t^g - w_t^g)_\times, \\
\frac{d}{dt} v_t &= R_t(\tilde{a}_t - b_t^a - w_t^a) + g, \\
\frac{d}{dt} p_t &= v_t.
\end{aligned} \tag{33}$$

The contact position is not affected by biases. Therefore, for legged robots, we have the same contact dynamics as (24). Since the biases are slow-varying signals, we can model their dynamics using random walks. That is, white Gaussian noises govern the dynamics as follows.

$$\begin{aligned}
\frac{d}{dt} b_t^g &= w_t^{bg}, \quad w_t^{bg} \sim \mathcal{GP}(0_{3,1}, \Sigma^{bg} \delta(t-t')), \\
\frac{d}{dt} b_t^a &= w_t^{ag}, \quad w_t^{ag} \sim \mathcal{GP}(0_{3,1}, \Sigma^{ag} \delta(t-t')).
\end{aligned} \tag{34}$$

With the new continuous dynamics, we can obtain our linearized error dynamics following Section IV-B:

$$\frac{d}{dt}\begin{bmatrix} \xi_t \\ \zeta_t \end{bmatrix} = A_t \begin{bmatrix} \xi_t \\ \zeta_t \end{bmatrix} + \begin{bmatrix} Ad_{\bar{X}_t} & 0 \\ 0 & I \end{bmatrix} w_t, \tag{35}$$

$$A_t = \begin{bmatrix} 0 & 0 & 0 & 0 & -\bar{R}_t & 0 \\ (g)_\times & 0 & 0 & 0 & -(\bar{v}_t)_\times \bar{R}_t & -\bar{R}_t \\ 0 & I & 0 & 0 & -(\bar{p}_t)_\times \bar{R}_t & 0 \\ 0 & 0 & 0 & 0 & -(\bar{d}_t)_\times \bar{R}_t & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{36}$$

C. Velocity Correction

Body velocity measurements provide a generic correction model that can work on any robotic platform. Specifically, the filer requires the ground-referenced body velocity [21, 24]. For some robotic applications, ground-referenced body velocity can be directly measured from the sensors, such as DVL for marine robots. However, accurate body velocity measurement is not always readily available. For wheeled robots with no lateral movements, one practical solution is to use the encoder measurements along with the nonholonomic constraints (i.e., velocity constraints that cannot be integrated) to form pseudo measurements for correction:

$$v_w = \begin{bmatrix} \frac{r(\dot{q}_r + \dot{q}_l)}{2} & 0 & 0 \end{bmatrix}^\mathsf{T}, \tag{37}$$

where $\dot{q}_r$ and $\dot{q}_l$ are angular velocity readings from the wheel encoders and r is the wheel radius. The nonholonomic constraints provide the pseudo measurements in the y and z components under the assumption that the robot does not drift side-way nor jump up vertically.

1) Correction Model: In DRIFT, the velocity correction model is implemented independently of the velocity source. This allows the library to be generic and to be applied to any robot with any source of velocity measurements. Here, we again assume the velocity measurements to be corrupted by white Gaussian noise:

$$\tilde{v}_t = v_t + w_t^v \quad w_t^v \sim \mathcal{GP}(0_{3,1}, \Sigma^v \delta(t-t')). \tag{38}$$

The body velocity is measured in the body frame. As a result, a right-invariant measurement model is used here.

$$\begin{aligned} Y_{t_k} &= X_{t_k}^{-1} b + V_{t_k}, \\ \begin{bmatrix} \tilde{v}_{t_k} \\ -1 \\ 0 \end{bmatrix} &= \begin{bmatrix} R_{t_k}^{\mathsf{T}} & -R_{t_k}^{\mathsf{T}} v_{t_k} & -R_{t_k}^{\mathsf{T}} p_{t_k} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} + \begin{bmatrix} w_{t_k}^{v} \\ 0 \\ 0 \end{bmatrix}. \end{aligned} \tag{39}$$

We proceed to find H as follows.

$$\begin{aligned} H\xi_k^r &= -\xi_k^{r\wedge} b \\ H \begin{bmatrix} \xi_k^{\omega} \\ \xi_k^{v} \\ \xi_k^{p} \end{bmatrix} &= - \begin{bmatrix} \xi_k^{\omega\wedge} & \xi_k^{v} & \xi_k^{p} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} \xi^{v} \\ 0 \\ 0 \end{bmatrix} \\ H &= \begin{bmatrix} 0_{1,3} & I & 0 \\ 0_{1,3} & 0 & 0 \\ 0_{1,3} & 0 & 0 \end{bmatrix}. \end{aligned}$$

Then, we can perform the update step using (18), where after eliminating zeros $H = \begin{bmatrix} 0_{1,3} & I & 0 \end{bmatrix}$, and $N = \bar{R}_{t_k} \operatorname{Cov}(w^v) \bar{R}_{t_k}^{\mathsf{T}}$.

2) Observability Analysis: Theorem 2 reveals the log-linear property of the error dynamics. Consequently, one can perform linear observability analysis using the linear error dynamic matrix [53], which, in our case, is time-invariant and nilpotent (with a degree of three):

$$\Phi = \exp(A\Delta t) = \begin{bmatrix} I & 0 & 0 \\ (g)_\times \Delta t & I & 0 \\ \frac{1}{2}(g)_\times \Delta t^2 & 0 & I \end{bmatrix}.$$

Accordingly, the observability matrix becomes:

$$\mathcal{O} = \begin{bmatrix} H \\ H\Phi \\ H\Phi^2 \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 & I & 0 \\ (g)_\times \Delta t & I & 0 \\ 2(g)_\times \Delta t & I & 0 \\ \vdots & \vdots & \vdots \end{bmatrix}.$$

The roll and pitch are observable, as well as the velocity. However, since gravity only contains a value in the z axis, the yaw angle is not observable.

D. Contact Correction

Foot contacts provide additional constraints for legged robot state estimation by assuming the contact foot velocity in the world frame to be zero over the contact period. This allows the estimator to observe the body velocity from the forward kinematics models. In fact, this has become a core module for many modern legged robot state estimation algorithms [10, 19, 73]. In DRIFT, we integrate the contact-inertial process model for legged robots proposed by Hartley et al. [19]. We assume the encoder measurements to be affected by zero-mean white Gaussian noise:

$$\tilde{q}_t = q_t + w_t^q, \quad w_t^q \sim \mathcal{GP}(0_{3,1}, \Sigma^q \delta(t - t')). \tag{40}$$

1) Contact State Augmentation: When a contact point is detected, we append it to the state using the forward kinematics and the current position estimate:

$$\bar{d}_{lt} = \bar{p}_t + \bar{R}_t h_p(\tilde{q}_t),$$

where $h_p(\cdot)$ is the forward kinematics function that maps the encoder measurements to the foot position in the body frame. The corresponding covariance can be augmented using

$$\begin{aligned} P_{t_k}^{\text{new}} &= F_{t_k} P_{t_k} F_{t_k}^{\mathsf{T}} + G_{t_k} \operatorname{Cov}(w_{t_k}^q) G_{t_k}^{\mathsf{T}}, \\ F_{t_k} &= \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \\ 0 & 0 & I \end{bmatrix}, \quad G_{t_k} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \bar{R}_{t_k} J_p(\tilde{q}_{t_k}), \end{bmatrix} \end{aligned}$$

where $J_p(\cdot)$ is the Jacobian of $h_p(\cdot)$.

The augmented contact state remains in the state throughout the contact phase. When the contact constraint breaks (i.e., when the foot is lifted), the contact state is marginalized via

$$\bar{X}_{t_k}^{\text{new}} = M \bar{X}_{t_k}, \bar{P}_{t_k}^{\text{new}} = M \bar{P}_{t_k} M^{\mathsf{T}}, M = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \end{bmatrix}. \tag{41}$$

2) Correction Model: Once the foot enters the contact phase, the augmented contact position follows (24) in the propagation step. When a new encoder measurement is obtained, the right-invariant correction model is given by

$$\begin{aligned} Y_{t_k} &= X_{t_k}^{-1} b + V_{t_k} \\ \begin{bmatrix} h_p(\tilde{q}_{t_k}) \\ 0 \\ 1 \\ -1 \end{bmatrix} &= \begin{bmatrix} R_{t_k}^{\mathsf{T}} & -R_{t_k}^{\mathsf{T}} v_{t_k} & -R_{t_k}^{\mathsf{T}} p_{t_k} & -R_{t_k}^{\mathsf{T}} d_{t_k} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix} \\ &\quad + \begin{bmatrix} J_p(\tilde{q}_{t_k}) w_{t_k}^q \\ 0 \\ 0 \end{bmatrix}, \end{aligned}$$

The linearized measurement matrix H and the noise matrix N are as follows.

$$\begin{aligned} H &= \begin{bmatrix} 0 & 0 & -I & I \end{bmatrix}, \\ N &= \bar{R}_{t_k} J_p(\tilde{q}_{t_k}) \operatorname{Cov}(w_{t_k}^q) J_p(\tilde{q}_{t_k})^{\mathsf{T}} \bar{R}_{t_k}^{\mathsf{T}}. \end{aligned}$$

E. Contact Estimator

The contact correction model provides a means to correct the predicted state reliably. However, obtaining reliable contact estimation is often challenging, and false contact detection can introduce additional biases into the system. Moreover, for some low-cost legged robots, foot contact sensors are often not readily available. In the spirit of making the proposed framework more generalizable to different robotic platforms, we propose an additional contact detection module for robots without dedicated contact sensors [27].

The proposed contact estimator is a self-supervised lightweight neural network that only takes measurements from an IMU and joint encoders as input. The neural network can operate at 830 Hz on an NVIDIA Jetson AGX Xavier, which can be easily attached to existing robots.

1) Contact State Representation: To design the network, we define our contact state C as a vector of binary values:

$$c = \begin{bmatrix} c_{RF} & c_{LF} & c_{RH} & c_{LH} \end{bmatrix}, \quad \text{for quadruped robots}$$
$$c = \begin{bmatrix} c_R & c_L \end{bmatrix}, \quad \text{for bipedal robots}$$

where $c_l \in \{0, 1\}$ for each leg l, with 1 indicate a firm contact and 0 as no contact.

Depending on the robot's motion, the contact states can have a finite set of combinations, from all feet on the ground to all in the air. As a result, we can address the problem using a classification framework. That is, for each timestamp, we estimate which foot configurations are posed by the robot. In order to follow typical classification pipelines, we map our contact state vector c to an integer S by treating c as a binary value and using binary-to-decimal conversion as the function. For quadruped robot, $S \in \{0, 1, \ldots, 15\}$, and for bipedal, $S \in \{0, 1, 2, 3\}$.

2) Input Data: The contact estimator depends solely on proprioceptive measurements, including joint encoders, IMU, and kinematics. For a synchronized time n, we concatenated the above measurements as $z_n = \begin{bmatrix} q_n & \dot{q}_n & a_n & \omega_n & d_{ln} & \dot{d}_{ln} \end{bmatrix}$. In order for the network to incorporate temporal information, we append previous $w - 1$ measurements to the current data: $D_n = \begin{bmatrix} z_{n-w}^\mathsf{T} & z_{n-w+1}^\mathsf{T} & \ldots & z_n^\mathsf{T} \end{bmatrix}^\mathsf{T}$. Therefore, at each time n, the network takes in a 2D matrix D_n and estimates the current contact configuration S_n.

3) Network Structure: The network comprises 2 convolutional blocks and 3 fully connected layers. Each convolutional block consists of 2 one-dimensional convolutions and a one-dimensional max pooling. We choose a one-dimensional kernel to improve efficiency and reduce memory usage. For nonlinearity, we employ ReLU as the activation function. To prevent the network from overfitting, we add a dropout mechanism at the end of the second convolutional block.

With the deep features extracted from the convolutional blocks, the fully connected layers convert them into the contact configuration S_n. Again, to prevent the network from overfitting, we employ dropout mechanisms for the first 2 fully connected layers. Finally, the cross-entropy loss is applied for the classification task:

$$L(P_i) = -\log \frac{\exp(P_i)}{\sum_j \exp(P_j)}, \tag{42}$$

where P_j is the probability output from the network of state j, and P_i is the probability of the ground truth state.

F. Gyro Filter

Low-cost IMUs can deteriorate performance. This is especially worse for states that are not observable, such as the yaw angle. To mitigate this issue, we propose an additional gyro filter that fuses angular velocity measurements from different sources, such as additional IMUs or kinematic models. The gyro filter is a linear Kalman filter, with its state defined as:

$$x := \begin{bmatrix} \omega^\mathsf{T} & b^{g\mathsf{T}} \end{bmatrix}^\mathsf{T}, \tag{43}$$

where ω is a 3-vector representing the angular velocity in the base IMU frame, and b^g denotes the corresponding biases.

1) Propagation: In the propagation step, we assume the measurement biases to be slowing-varying quantities, which remain the same between two timestamps. As a result, we can formulate an integration model by taking the differences between two consecutive measurements and canceling out the biases as:

$$x_{k+1} = x_k + \begin{bmatrix} \tilde{\omega}_{k+1}^\alpha - \tilde{\omega}_k^\alpha, \\ 0_{3\times 1} \end{bmatrix}, P_k = P_k + Q_k. \tag{44}$$

Here, $\tilde{\omega}^\alpha$ denotes the angular velocity measurements from one sensing source, P_k is the filter covariance, and Q_k is the process noise covariance.

2) Correction: A second source of angular velocity measurements, $\tilde{\omega}^\beta$, can be used to correct the filter using a linear measurement model:

$$\tilde{\omega}^\beta = Hx.$$

The corresponding H matrix is $\begin{bmatrix} I_{3\times 3} & I_{3\times 3} \end{bmatrix}$ if the measurement is biased, and $\begin{bmatrix} I_{3\times 3} & 0_{3\times 3} \end{bmatrix}$ if the measurement is unbiased. With this model, the filter can be corrected following the standard EKF correction algorithm [2].

VI. Experimental Results

A. Contact Estimation

To train the proposed contact estimator, we create open-source contact data sets using an MIT Mini Cheetah robot [137]. We record all proprioceptive sensor measurements, including joint encoders data, foot position and velocities, IMU measurements, and estimated joint torques from an MIT controller [138]. All data are upsampled to $1000\,\mathrm{Hz}$ to match the IMU frequency. The data sets are created over 8 different terrains, including asphalt roads, concrete, forest, grass, middle pebbles, small pebbles, rock road, and sidewalks. In addition to the above terrains, several sequences of the robot walking in the air (i.e., not having contact with the ground while operating the same gait.) are recorded to provide negative examples to the network. We collect around 1,000,000 data points and reserve some sequences for state estimation tests. The rest of the data sets are separated into testing, validation, and training sets with the ratio of 15%, 15%, and 70%. Fig. 3 shows examples of the different terrain types in the contact data sets.

The ground truth labels are generated using an offline algorithm, which takes the robot's foot height in the hip frame as input. The algorithm applies a low-pass filter and uses future and past data points around the current time stamp to extract local minima and maxima. Moreover, we observe a bouncing effect on the robot's foot upon touchdown after inspecting slow-motion videos of Mini Cheetah's walking patterns. The bouncing results in a sudden change of foot height in the signal, causing false positives which can be removed by applying the low-pass filter.

We present the accuracy, false positive rate (FPR), and false negative rate (FNR) of the proposed method against other baselines. For baselines, we implement a model-based approach [83–85], where the estimated ground reaction force (GRF) is computed via the general equation of motion with

Fig. 3. Left: The configuration of an MIT Mini Cheetah robot for contact data collection. Right: Different terrain types in the contact data set.

TABLE I
ACCURACY COMPARISON AGAINST BASELINES.

Sequence	Method	% Accuracy					% False Positive Rate	% False Negative Rate
		Leg RF	Leg LF	Leg RH	Leg LH	Leg Avg	Leg Avg	Leg Avg
Concrete Test Sequence	GRF Thresholding	73.43	70.02	71.69	70.04	71.30	37.07	13.24
	Gait Cycle	85.66	84.98	84.68	85.11	85.11	22.95	**0.00**
	Proposed Method	**98.34**	**97.87**	**97.95**	**98.56**	**98.18**	**1.45**	2.51
Grass Test Sequence	GRF Thresholding	82.55	78.93	84.62	82.48	82.14	26.87	**0.63**
	Gait Cycle	92.41	92.38	91.04	90.55	91.59	10.95	3.53
	Proposed Method	**98.08**	**97.57**	**97.73**	**97.73**	**97.78**	**2.35**	1.98
Forest Test Sequence	GRF Thresholding	80.99	80.09	82.75	83.24	81.77	26.54	1.84
	Gait Cycle	83.03	82.56	84.44	84.28	83.58	24.71	**0.08**
	Proposed Method	**97.05**	**96.62**	**97.24**	**97.40**	**97.08**	**2.82**	3.12

a low-pass filter. A fixed threshold is set to the estimated GRF to detect the contact events. We would like to highlight that because there is no direct access to the motor current on the Mini Cheetah robot, we use the torque command from the controller to approximate the actual torque on the actuators. We denote this method as GRF thresholding. In addition, we also obtain the gait cycle command from the MIT controller [138] to serve as a second contact estimation baseline.

Table I lists the accuracy, FPR, and FNR of the compared methods on three test sequences. We can see that the proposed method achieves the highest accuracy across all sequences and the GRF thresholding has the worst performance. Although the baselines obtain slightly lower FNRs, our method maintains significantly lower FPRs. Lower FPR is crucial for state estimation tasks as false positive contact events can introduce biased measurements into the system.

B. Legged Robot

We present the state estimation results using DRIFT with the proposed contact estimator for the Mini Cheetah robot. The robot walks on an outdoor grassy field, where we use a motion capture system to obtain the ground truth pose. We deploy the proposed contact estimator, as well as the two baseline contact estimation methods with DRIFT. Fig. 4 shows the estimated trajectory. Qualitatively, DRIFT with the proposed contact estimator produces a trajectory closer to the ground truth. In addition, the two baseline methods produce extra height (z) drifts. We argue this is the result of the high false positive rates of the baseline methods. This experiment confirms DRIFT's support for accurate legged robot state estimation. Moreover, the contact estimation module allows robots without contact sensors, such as the MIT Mini Cheetah, to obtain reliable contact estimation results, which are essential to contact-aided state estimation algorithms.

C. Indoor Wheeled Robot

For indoor applications, we deploy DRIFT on a Fetch robot. A picture of the Fetch robot is shown in Fig. 5. Fetch is equipped with a low-cost IMU and wheel encoders on each wheel, which we use as the input to DRIFT. We collected data sets in a lab environment, where Fetch drove on a polished concrete surface for several sequences. We obtained the ground truth poses from a motion capture system. We additionally implement a MEKF [43] to serve as a baseline algorithm. Both DRIFT and MEKF take only IMU and encoder measurements as input and estimate the 6D pose of the robot. The low-cost IMU on Fetch can cause the unobservable yaw angle to degrade and further reduce the accuracy of the overall state estimation. As a result, we implemented an additional gyro filter described in Sec. V-F to fuse the yaw angular velocity measurement from the IMU with the encoder measurements.

Fig. 6 shows a top-down view of the estimated trajectory on the `01_square` sequence. We observe that because of the inaccurate yaw angular velocity measurements, both DRIFT and MEKF experience significant drift after turning. However, with the help of the gyro filter, we can obtain a much better

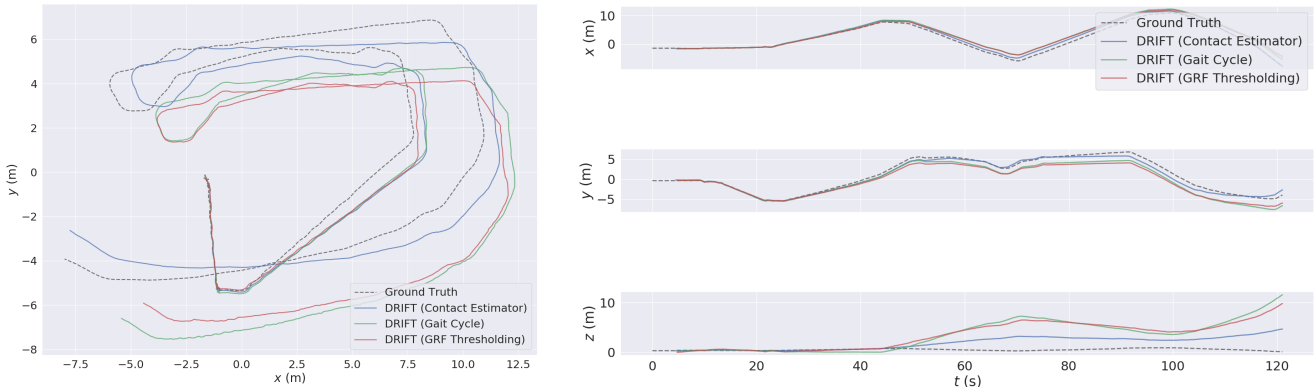

Fig. 4. The estimated trajectories from DRIFT using different contact estimation methods on the Mini Cheetah data set. The robot walks on a grassy field with a motion capture system, which is used for ground truth capturing. With the proposed contact estimator, DRIFT produces the best trajectory. Contrarily, the two baseline methods introduce significant drift in the height (z) axis. This figure is generated with the aid of the Python package evo [139].

TABLE II
THE RMSE OF RELATIVE POSE ERROR (RPE) FOR FETCH. WE REPORT THE RPE IN THE UNIT OF DRIFT PER METER.

		01˙square	02˙um	03˙eight	04˙random	05˙right˙turn	06˙left˙turn	07˙origin	**Avg**	**Std**
	Trajectory Length (m)	10.59	14.15	11.61	38.91	15.64	22.29	11.26	17.78	N/A
	Duration (sec)	43.720	49.62	34.62	99.06	39.72	51.71	46.68	52.16	N/A
MEKF [43]	Trans. (m/m)	0.1012	0.0634	0.0855	0.0990	0.0853	**0.0435**	0.1126	0.0844	0.0182
	Rot. (°/m)	**1.0555**	5.5752	3.8387	6.9809	3.9911	1.6304	**2.4502**	3.6460	1.1016
DRIFT	Trans. (m/m)	0.0688	0.0587	0.0629	0.0863	0.0857	0.0436	0.0786	0.0692	0.0156
	Rot. (°/m)	1.9766	**5.3704**	**3.6134**	**6.6857**	3.6606	**1.5580**	2.4741	3.6198	1.8587
DRIFT (Gyro Filter)	Trans. (m/m)	**0.0565**	**0.0585**	**0.0566**	**0.0801**	**0.0613**	0.0438	**0.0558**	**0.0590**	0.0127
	Rot. (°/m)	1.5723	5.3865	3.8561	6.7959	**1.9193**	1.9246	3.4866	**3.5631**	1.2172

Fig. 5. A Fetch robot used in the indoor experiments. The figure shows the robot at the Department of Naval Architecture and Marine Engineering on the University of Michigan campus. Fetch is a differential drive robot that is commonly used for indoor service robot research.

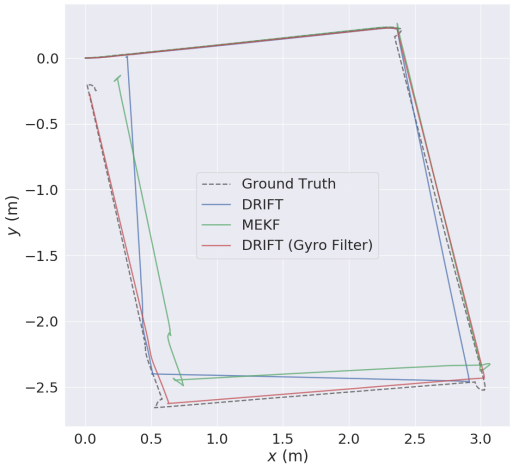

Fig. 6. The bird's-eye view of the estimated trajectory on the `01_square` sequence in the Fetch data set. Overall, the estimated trajectory from DRIFT is smoother than MEKF. Among them, DRIFT with the gyro filter has the best performance.

trajectory estimation. In addition, we notice that the estimated trajectory from DRIFT is much smoother than MEKF.

Since DRIFT is an odometry framework instead of a full SLAM system, we follow Sturm et al. [140] and report the root-mean-square error (RMSE) of the Relative Pose Error (RPE), in the unit of drift per meter in Table II. DRIFT performs consistently better than the baseline. With the aid of the gyro filter, the position estimation is further improved, which likely results from a more accurate yaw angle estimation.

D. Outdoor Wheeled Robot

We deploy DRIFT on a Clearpath Robotics Husky robot. Husky is a four-wheeled differential-drive robot, as shown in Fig. 1. The robot is equipped with two VectorNav VN-100 IMUs and wheel encoders on each side. We conducted both short and long-horizon experiments using the Husky robot.

TABLE III
THE RMSE OF RELATIVE POSE ERROR (RPE) FOR HUSKY SHORT-HORIZON EXPERIMENT. WE REPORT THE RPE IN THE UNIT OF DRIFT PER METER.

		1˙rectangle	2˙batman	3˙broken˙cat	4˙um	5˙type˙z	6˙rocker	7˙peace	8˙cheetah	9˙m	**Avg**	**Std**
	Trajectory Length (m)	20.56	38.47	90.43	39.11	59.89	39.46	40.56	50.37	63.72	49.17	N/A
	Duration (sec)	49.78	77.92	154.57	67.09	90.48	67.49	66.50	92.01	100.07	85.10	N/A
MEKF [43]	Trans. (m/m)	**0.0477**	0.0812	0.1139	0.0625	**0.0652**	0.0680	0.0749	0.0829	**0.0755**	0.0747	0.0384
	Rot. (°/m)	1.3738	**4.8355**	2.5033	1.6061	1.9539	1.6396	1.4865	1.6416	1.3962	2.0485	1.0650
DRIFT	Trans. (m/m)	0.0482	**0.0781**	**0.0881**	**0.0564**	0.0678	**0.0627**	**0.0728**	**0.0817**	**0.0755**	**0.0701**	0.0351
	Rot. (°/m)	**1.0152**	4.8559	**1.7097**	**1.3381**	**1.3089**	**1.3126**	**1.1606**	**1.6321**	**0.8657**	**1.6888**	1.2657

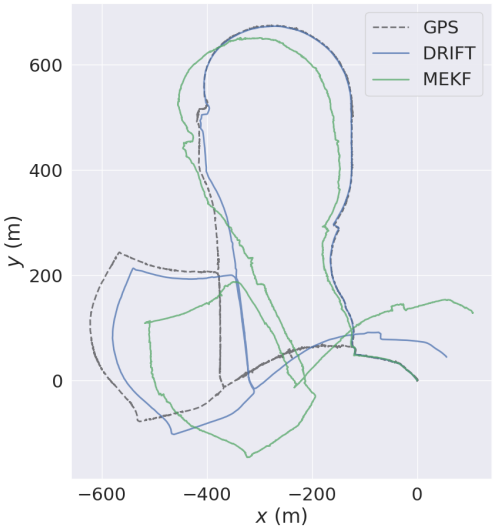

Fig. 7. Husky long-horizon experiment. The robot starts and ends at the bottom right corner. While being purely proprioceptive, DRIFT can obtain accurate estimates after 3 kilometers of operation.

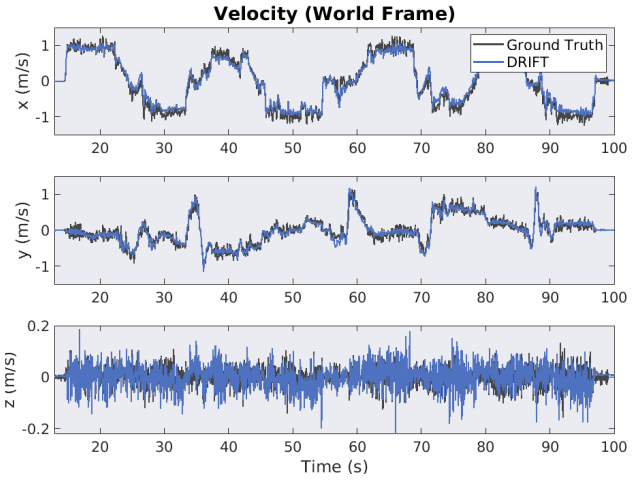

Fig. 8. The velocity estimated from DRIFT overlapped with the ground truth velocity for the 9_M sequence. The estimated velocity converges to the ground truth velocity, which agrees with the observability analysis in Sec. V.

Fig. 9. The Polaris MRZR from the Neya Systems. The full-size military vehicle was tested in an off-road forest area. The environment was featureless and the ground was covered in leaves.

The short-horizon data were collected on an outdoor grassy field with a motion capture system, which we used to obtain ground truth poses. For the long-horizon experiment, we drove the robot on the sidewalk for 55 minutes. The total trajectory path is around 3 kilometers. Since no motion capture system is possible for such large-scale experiments, we used the GPS signals as a proxy for the ground truth trajectory. To better assess the performance of DRIFT, we again used a MEKF [43] as the baseline algorithm.

Table III reports the RPE of the short-horizon experiment. On average, DRIFT performs better than the MEKF. This is expected as both the state-transition matrix A and linearized measurement matrix H of the InEKF are independent of the estimated states, while the corresponding matrices in the MEKF depend on the estimated states.

The benefit of using InEKF becomes even more obvious in long-horizon operations. Fig. 1 and 7 show the bird's-eye view of the estimated trajectory for the long-horizon experiment. We can see DRIFT performs significantly better than MEKF, with the final drift in the xy plane to be $71.41\,\mathrm{m}$ and $166.40\,\mathrm{m}$, respectively. We highlight that DRIFT uses proprioceptive data only. This demonstrates DRIFT's capability of being a reliable odometry source for visual-SLAM systems in perceptually degraded environments, even for long-horizon operations.

We also compare the estimated velocity in the world frame with the ground truth velocity in Fig. 8. The ground truth velocity is obtained by differentiating the ground truth pose from the motion capture system. We see the estimated velocity converges to the true velocity, which agrees with the observability analysis in Sec. V.

E. Off-road Vehicle

DRIFT also supports full-size vehicles with a single shaft encoder. We evaluate DRIFT on data gathered with the support of Neya Systems from their full-size military test vehicle,

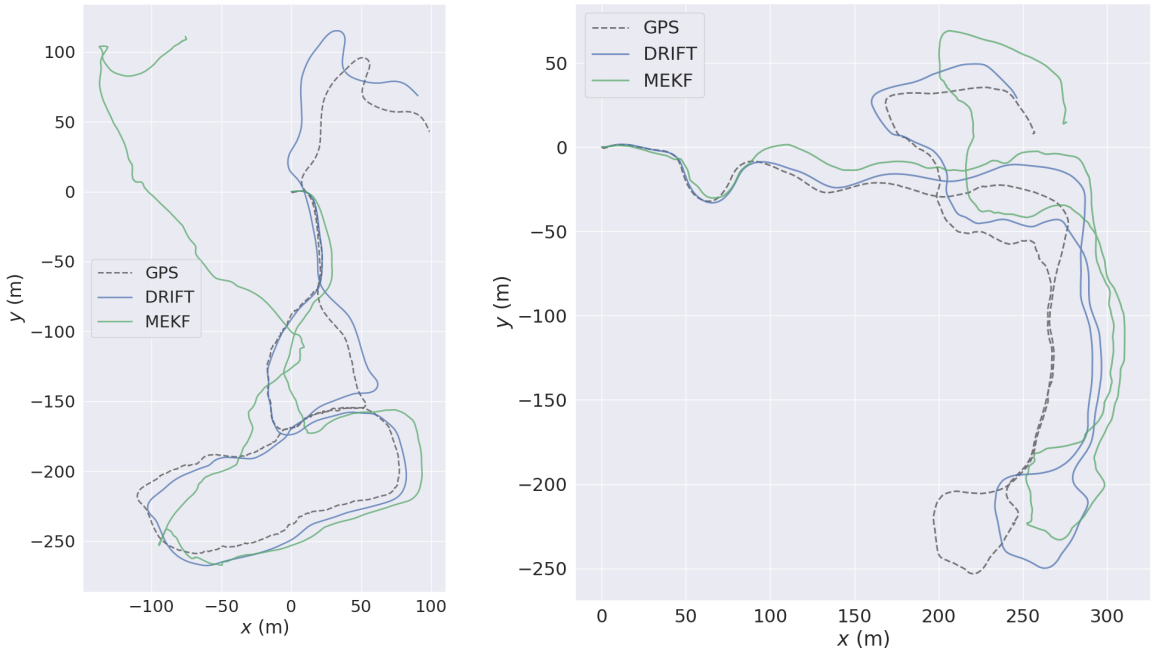

Fig. 10. Left: `1_follow`. Right: `2_leader_follow`. Bird's-eye views of two sequences from the off-road data set. The data set was collected using a full-size military test vehicle in a featureless forest area. Qualitatively, we see DRIFT outperforms MEKF in both sequences. Since drift only relies on proprioceptive measurements, it can serve as a robust odometry source for modern SLAM systems in such featureless environments.

the Polaris MRZR. The data was collected with support from Neya Systems at two of their off-road test facilities. As shown in Fig. 9, the test facility is a featureless forest area with leafy ground. In this experiment, we obtain GPS signals as approximations to the ground truth positions.

Compared to a motion capture system, GPS provides less accurate positional signals. As a result, we compute the final drift and drift percentage in Table IV instead of the RPE. We define the drift percentage as Final Drift/Trajectory Length. From the table, we see that DRIFT achieves low drift percentages across the three sequences and outperforms the MEKF.

Fig. 10 shows the bird's-eye view of two off-road sequences, with trajectory lengths of 1.48 km and 1.26 km. DRIFT performs significantly better than MEKF in both sequences. This experiment demonstrates that with only an IMU and shaft encoder as inputs, DRIFT can produce highly accurate odometry results for full-size vehicles in off-road environments. These off-road environments are often featureless or with repetitive features. Since DRIFT only relies on proprioceptive measurements, it can serve as a reliable odometry source for perception systems in such scenarios.

F. Marine Robot

For marine robots, ground-reference velocity can be obtained via a DVL. The DVL is an acoustic sensor that emits directional acoustic beams to the seabed. The seabed-referenced velocity is then computed using the Doppler effect of the reflected acoustic signals. This enables us to correct the predicted state using the velocity correction method described in Sec. V-C.

To verify the efficacy of the proposed framework on marine robots, we set up experiments in an open-sourced marine simulation software, Stonefish [141]. An IQUA Robotics Girona500 Autonomous Underwater Vehicle (AUV) was simulated with an IMU and a DVL attached to the robot, as shown in Fig. 11. The sensor data is corrupted by unbiased Gaussian noises. We used the noise parameters from the VectorNav VN-100 IMU manual to assign the simulation noise parameters. Specifically, we applied $\sigma = \rho \cdot \sqrt{f_s}$ to recover the standard deviation, σ, from the noise density, ρ, and operating frequency, f_s. Table V lists the noise parameters used in the simulation. We set the IMU and DVL frequencies to $200\,\mathrm{Hz}$ and $20\,\mathrm{Hz}$.

Fig. 11 shows the estimated trajectories from the underwater data set. Qualitatively, we see both DRIFT and MEKF perform well on this sequence. In particular, DRIFT performs slightly better in position estimation in the x and y axes, as well as the roll and pitch estimation. Moreover, the roll and pitch estimations converge to the true values after a perturbation around $36\,\mathrm{sec}$, which agrees with our observability analysis in Sec. V-C.

G. Runtime Analysis

We perform runtime evaluations using a personal laptop with an Intel i5-11400H CPU and an NVIDIA Jetson AGX Xavier (CPU). The Jetson AGX Xavier is commonly used by many robotic platforms. We record the processing time for every propagation and correction method in the InEKF filter and the gyro filter, as presented in Table VI. DRIFT can operate at an extremely high frequency using CPU-only computation, even on the resourced-constrained Jetson AGX

TABLE IV
FINAL DRIFT AND DRIFT PERCENTAGE OF THE OFF-ROAD VEHICLE DATA SET, WHERE THE FULL-SIZE VEHICLE OPERATES IN A FOREST AREA FOR MORE THAN 1 KILOMETER. THE DRIFT PERCENTAGE IS COMPUTED BY (FINAL DRIFT)/(TRAJECTORY LENGTH).

		1˙follow	2˙leader˙follow	3˙long˙run	**Avg**
	Trajectory Length (m)	1481.9	1264.0	1785.4	1510.4
	Duration (sec)	317.0	424.7	605.8	449.2
MEKF [43]	Final Drift (m)	187.4	29.6	392.1	203.0
	Drift Percentage (%)	12.7	2.3	22.0	12.3
DRIFT	Final Drift (m)	**30.2**	**25.8**	**97.3**	**51.1**
	Drift Percentage (%)	**2.0**	**2.0**	**5.5**	**3.18**

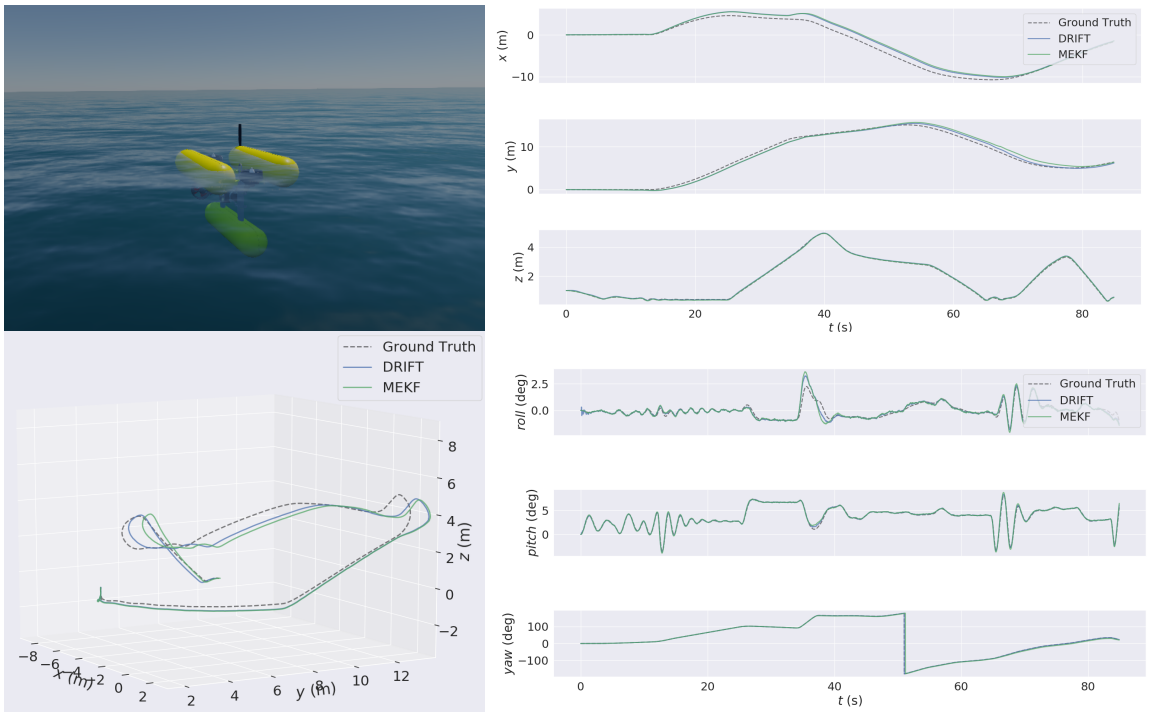

Fig. 11. Top Left: A Girona500 AUV simulated in the Stonefish simulator [141]. Bottom Left: The 3D estimated trajectories from the underwater data set. Top Right: The estimated trajectories projected to the x, y, and z axes. From the plots, we see DRIFT performs slightly better than MEKF. Bottom Right: The estimated roll, pitch, and yaw angles. We observe the roll and pitch estimations converge to the true values after a perturbation around 36 sec.

TABLE V
NOISE PARAMETERS USED IN THE UNDERWATER SIMULATION.

Measurement Type	Noise Parameter
Angular Velocity	$0.0035\ ^\circ/\mathrm{sec}/\sqrt{\mathrm{Hz}}$
Linear Acceleration	$0.0014\ \mathrm{m/sec}/\sqrt{\mathrm{Hz}}$
DVL Sensor	$0.02626\ \mathrm{m/sec}$

TABLE VI
RUNTIME EVALUATION OVER DATASETS ON A LAPTOP CPU AND AN NVIDIA JETSON AGX XAVIER. FOR THE JETSON AGX XAVIER, ALL COMPUTATIONS ARE OPERATED ON THE CPU ONLY.

	i5-11400H		AGX Xavier (CPU)	
Unit: μs	mean	std	mean	std
InEKF				
propagation	11.33	4.00	18.35	4.19
propagation with contact	10.32	4.76	22.56	7.21
velocity correction	9.91	4.80	18.46	6.66
contact correction	17.46	9.78	29.39	13.07
Gyro Filter				
propagation	2.57	3.46	3.96	2.28
correction	2.85	2.89	4.64	4.40

Xavier. For the optional contact estimator, the inference speed on an NVIDIA RTX 3090 GPU is approximately 1100 Hz, and the inference speed on a Jetson AGX Xavier (GPU) is around 830 Hz after TensorRT optimization.

VII. DISCUSSION AND FUTURE WORK

One limitation of DRIFT is the assumption of nonholonomic constraints. These constraints can be detached from the robot's actual behavior. Learning such constraints provides a way to use sensory inputs instead of assumptions [27, 142, 143]. Moreover, the nonholonomic constraints are violated when the robot drifts. Slip detection and friction estimation are challenging and necessary tasks for robot state estimation [99].

Instead of relying on the nonholonomic assumption, marine robots can obtain ground-referenced velocity measurements from a DVL. However, it is worth noticing that a reliable velocity measurement requires a sufficient number of emitted beams to have a line of sights on the seabed, known as *bottom-lock* [123]. If the robot operates at an extreme roll and pitch angle, or if the robot is too far away from the seabed, the bottom-lock might be lost. A typical DVL can obtain bottom-lock for ranges up to 500 m, depending on the frequency of the sensor [144]. In this work, however, we assumed the DVL can always obtain the bottom-lock.

Lastly, DRIFT is a real-time dead-reckoning system. While dead-reckoning might be sufficient for some applications, in many cases it can not replace a localization or SLAM system, where global consistency is the objective. As a result, we position DRIFT as an odometry module of a perception system or a dead-reckoning system for applications that do not require global consistency. A natural next step will then be to incorporate DRIFT into a visual SLAM system for real-time globally consistent state estimation.

VIII. Conclusion

We developed DRIFT, an open-source, real-time proprioceptive state estimator for multiple robotic platforms. DRIFT is based on the invariant Kalman filtering, leading to significantly better consistency in robot state estimation tasks. Optionally, DRIFT provides two additional modules, a contact estimator and a gyro filter, to assist operations on low-cost robots. We evaluate DRIFT using various robotic platforms with real-world experiments, including a legged robot, an indoor service robot, a field robot, and a full-size vehicle. Additionally, we report simulation results of an underwater vehicle. These experiments verify DRIFT's capability of being a reliable odometry source, even for long-horizon operations. Future directions include extending the framework to support a more general equivariant filter [60, 64], broader incorporation with symmetry-preserving perception and control algorithms [13], and incorporating DRIFT into a more comprehensive localization system for global consistency.

References

[1] R. Carroll, "Nasa-developed spherical robots to the rescue," *NASA's Spinoff Publication*, 2023. [Online]. Available: https://www.nasa.gov/directorates/spacetech/spinoff/NASA_Developed_Spherical_Robots_to_the_Rescue

[2] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[3] R. Hartley, M. G. Jadidi, L. Gan, J.-K. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2018, pp. 3783–3790.

[4] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon, "Online LiDAR-SLAM for legged robots with robust registration and deep-learned loop closure," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2020, pp. 4158–4164.

[5] S. Teng, D. Chen, W. Clark, and M. Ghaffari, "An error-state model predictive control on connected matrix Lie groups for legged robot control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2022, pp. 8850–8857.

[6] J. Jang, S. Teng, and M. Ghaffari, "Convex geometric trajectory tracking using Lie algebraic MPC for autonomous marine vehicles," *IEEE Robot. Autom. Lett.*, pp. 1–8, 2023.

[7] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982.

[8] Y. Song and J. W. Grizzle, "The extended Kalman filter as a local asymptotic observer for nonlinear discrete-time systems," in *Proc. Amer. Control Conf.*, 1992, pp. 3365–3369.

[9] L. A. McGee, *Discovery of the Kalman filter as a practical tool for aerospace and industry*. National Aeronautics and Space Administration, 1985, vol. 86847.

[10] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, "Pronto: A multi-sensor state estimator for legged robots in real-world scenarios," *Frontiers in Robotics and AI*, vol. 7, p. 68, 2020.

[11] P.-M. Lee, B.-H. Jun, K. Kim, J. Lee, T. Aoki, and T. Hyakudome, "Simulation of an inertial acoustic navigation system with range aiding for an autonomous underwater vehicle," *IEEE J. Ocean. Eng.*, vol. 32, no. 2, pp. 327–345, 2007.

[12] J. A. Cobano, J. Estremera, and P. G. De Santos, "Location of legged robots in outdoor environments," *Robot. and Auton. Syst.*, vol. 56, no. 9, pp. 751–761, 2008.

[13] M. Ghaffari, R. Zhang, M. Zhu, C. E. Lin, T.-Y. Lin, S. Teng, T. Li, T. Liu, and J. Song, "Progress in symmetry preserving robot perception and control through geometry and learning," *Front. Robot. AI*, vol. 9, p. 232, 2022.

[14] A. Barrau and S. Bonnabel, "The invariant extended Kalman filter as a stable observer," *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 1797–1812, 2017.

[15] ——, "Invariant Kalman filtering," *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

[16] T. Zhang, K. Wu, J. Song, S. Huang, and G. Dissanayake, "Convergence and consistency analysis for a 3-D invariant-ekf SLAM," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 733–740, 2017.

[17] A. Barrau and S. Bonnabel, "An EKF-SLAM algorithm with consistency properties," *arXiv preprint arXiv:1510.06263*, 2015.

[18] Y. Song, Z. Zhang, J. Wu, Y. Wang, L. Zhao, and S. Huang, "A right invariant extended Kalman filter for object based SLAM," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 1316–1323, 2021.

[19] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended Kalman filtering for robot state estimation," *Int. J. Robot. Res.*, vol. 39, no. 4, pp. 402–430, 2020.

[20] H.-C. Lin and M. Mistry, "Contact surface estimation via haptic perception," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2020, pp. 5087–5093.

[21] S. Teng, M. W. Mueller, and K. Sreenath, "Legged robot state estimation in slippery environments using invariant extended Kalman filter with velocity update," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2021, pp. 3104–3110.

[22] K. Wu, T. Zhang, D. Su, S. Huang, and G. Dissanayake, "An invariant-ekf vins algorithm for improving consistency," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2017, pp. 1578–1585.

[23] C. Liu, C. Jiang, and H. Wang, "Ingvio: A consistent invariant filter for fast and high-accuracy gnss-visual-inertial odometry," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1850–1857, 2023.

[24] E. R. Potokar, K. Norman, and J. G. Mangelson, "Invariant extended Kalman filtering for underwater navigation," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5792–5799, 2021.

[25] H. Chen, H. Bai, and C. N. Taylor, "Invariant-EKF design for quadcopter wind estimation," in *Proc. Amer. Control Conf.* IEEE, 2022, pp. 1236–1241.

[26] N. Y. Ko, W. Youn, I. H. Choi, G. Song, and T. S. Kim, "Features of invariant extended Kalman filter applied to unmanned aerial vehicle navigation," *Sensors*, vol. 18, no. 9, p. 2855, 2018.

[27] T.-Y. Lin, R. Zhang, J. Yu, and M. Ghaffari, "Legged robot state estimation using invariant Kalman filtering and learned contact events," in *Proceedings of the 5th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, A. Faust, D. Hsu, and G. Neumann, Eds., vol. 164. PMLR, 08–11 Nov 2022, pp. 1057–1066.

[28] V. Jurdjevic and H. J. Sussmann, "Control systems on lie groups," *Journal of Differential equations*, vol. 12, no. 2, pp. 313–329, 1972.

[29] R. W. Brockett, "Lie theory and control systems defined on spheres," *SIAM Journal on Applied Mathematics*, vol. 25, no. 2, pp. 213–225, 1973.

[30] ——, "System theory on group manifolds and coset spaces," *SIAM Journal on control*, vol. 10, no. 2, pp. 265–284, 1972.

[31] J. Grizzle and S. Marcus, "The structure of nonlinear control systems possessing symmetries," *IEEE Trans. Autom. Control*, vol. 30, no. 3, pp. 248–258, 1985.

[32] A. S. Willsky and S. I. Marcus, "Estimation for bilinear stochastic systems," in *Variable Structure Systems with Application to Economics*

and Biology: Proceedings of the Second US-Italy Seminar on Variable Structure Systems, May 1974. Springer, 1975, pp. 116–137.
[33] T. E. Duncan, "Some filtering results in riemann manifolds," *Information and Control*, vol. 35, no. 3, pp. 182–195, 1977.
[34] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.
[35] S. Salcudean, "A globally convergent angular velocity observer for rigid body motion," *IEEE Trans. Autom. Control*, vol. 36, no. 12, pp. 1493–1497, 1991.
[36] J. Thienel and R. M. Sanner, "A coupled nonlinear spacecraft attitude controller and observer with an unknown constant gyro bias and gyro noise," *IEEE Trans. Autom. Control*, vol. 48, no. 11, pp. 2011–2015, 2003.
[37] R. Mahony, T. Hamel, and J.-M. Pflimlin, "Complementary filter design on the special orthogonal group SO(3)," in *Proc. IEEE Conf. Decision Control*. IEEE, 2005, pp. 1477–1484.
[38] T. Hamel and R. Mahony, "Attitude estimation on so [3] based on direct inertial measurements," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2006, pp. 2170–2175.
[39] N. Aghannan and P. Rouchon, "An intrinsic observer for a class of lagrangian systems," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 936–945, 2003.
[40] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey, "Circumventing dynamic modeling: Evaluation of the error-state Kalman filter applied to mobile robot localization," in *Proc. IEEE Int. Conf. Robot. and Automation*, vol. 2. IEEE, 1999, pp. 1656–1663.
[41] N. Trawny and S. I. Roumeliotis, "Indirect Kalman filter for 3d attitude estimation," *University of Minnesota, Dept. of Comp. Sci. & Eng., Tech. Rep*, vol. 2, p. 2005, 2005.
[42] V. Madyastha, V. Ravindra, S. Mallikarjunan, and A. Goyal, "Extended Kalman filter vs. error state Kalman filter for aircraft attitude estimation," in *AIAA Guidance, Navigation, and Control Conference*, 2011, p. 6615.
[43] J. Sola, "Quaternion kinematics for the error-state Kalman filter," *arXiv preprint arXiv:1711.02508*, 2017.
[44] F. L. Markley, "Attitude error representations for Kalman filtering," *Journal of guidance, control, and dynamics*, vol. 26, no. 2, pp. 311–317, 2003.
[45] D. P. Koch, D. O. Wheeler, R. W. Beard, T. W. McLain, and K. M. Brink, "Relative multiplicative extended Kalman filter for observable GPS-denied navigation," *Int. J. Robot. Res.*, vol. 39, no. 9, pp. 1085–1121, 2020.
[46] J. K. Hall, N. B. Knoebel, and T. W. McLain, "Quaternion attitude estimation for miniature air vehicles using a multiplicative extended Kalman filter," in *2008 IEEE/ION Position, Location and Navigation Symposium*. IEEE, 2008, pp. 1230–1237.
[47] R. C. Leishman and T. W. McLain, "Multiplicative extended Kalman filter for relative rotorcraft navigation," *Journal of Aerospace Information Systems*, vol. 12, no. 12, pp. 728–744, 2015.
[48] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. D. Remy, and R. Siegwart, "State estimation for legged robots-consistent fusion of leg kinematics and IMU," *Robotics*, vol. 17, pp. 17–24, 2013.
[49] N. Rotella, M. Bloesch, L. Righetti, and S. Schaal, "State estimation for a humanoid robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2014, pp. 952–958.
[50] F. Bonin-Font, J.-P. Beltran, and G. Oliver, "Multisensor aided inertial navigation in 6dof auvs using a multiplicative error state Kalman filter," in *2013 MTS/IEEE OCEANS-Bergen*. IEEE, 2013, pp. 1–7.
[51] S. Bonnabel, P. Martin, and P. Rouchon, "Non-linear symmetry-preserving observers on lie groups," *IEEE Trans. Autom. Control*, vol. 54, no. 7, pp. 1709–1713, 2009.
[52] S. Bonnabel, "Left-invariant extended Kalman filter and attitude estimation," in *Proc. IEEE Conf. Decision Control*. IEEE, 2007, pp. 1027–1032.
[53] A. Barrau, "Non-linear state error based extended Kalman filters with applications to navigation," Ph.D. dissertation, Mines Paristech, 2015.
[54] Y. Gao, C. Yuan, and Y. Gu, "Invariant filtering for legged humanoid locomotion on a dynamic rigid surface," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 4, pp. 1900–1909, 2022.
[55] M. Barczyk and A. F. Lynch, "Invariant observer design for a helicopter uav aided inertial navigation system," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 3, pp. 791–806, 2012.
[56] P. Chauchat, A. Barrau, and S. Bonnabel, "Invariant smoothing on lie groups," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2018, pp. 1703–1710.
[57] P. Chauchat, S. Bonnabel, and A. Barrau, "Invariant smoothing with low process noise," in *Proc. IEEE Conf. Decision Control*. IEEE, 2022, pp. 4758–4763.
[58] J.-H. Kim, S. Hong, G. Ji, S. Jeon, J. Hwangbo, J.-H. Oh, and H.-W. Park, "Legged robot state estimation with dynamic contact event information," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6733–6740, 2021.
[59] J. Huai, Y. Lin, Y. Zhuang, and M. Shi, "Consistent right-invariant fixed-lag smoother with application to visual inertial SLAM," in *Proc. AAAI Nat. Conf. Artif. Intell.*, vol. 35, no. 7, 2021, pp. 6084–6092.
[60] P. van Goor, T. Hamel, and R. Mahony, "Equivariant filter (EqF): A general filter design for systems on homogeneous spaces," in *Proc. IEEE Conf. Decision Control*. IEEE, 2020, pp. 5401–5408.
[61] R. Mahony and J. Trumpf, "Equivariant filter design for kinematic systems on Lie groups," *IFAC-PapersOnLine*, vol. 54, no. 9, pp. 253–260, 2021.
[62] P. van Goor, "Equivariant filters for visual spatial awareness," Ph.D. dissertation, The Australian National University (Australia), 2023.
[63] A. Fornasier, Y. Ng, C. Brommer, C. Böhm, R. Mahony, and S. Weiss, "Overcoming bias: Equivariant filter design for biased attitude estimation with online calibration," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 12 118–12 125, 2022.
[64] A. Fornasier, Y. Ng, R. Mahony, and S. Weiss, "Equivariant filter design for inertial navigation systems with input measurement biases," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2022, pp. 4333–4339.
[65] P. van Goor and R. Mahony, "An equivariant filter for visual inertial odometry," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2021, pp. 14 432–14 438.
[66] ——, "Eqvio: An equivariant filter for visual inertial odometry," *arXiv preprint arXiv:2205.01980*, 2022.
[67] Y. Luo, C. Guo, and J. Liu, "Equivariant filtering framework for inertial-integrated navigation," *Satellite Navigation*, vol. 2, no. 1, pp. 1–17, 2021.
[68] G. P. Roston and E. P. Krotkov, *Dead Reckoning Navigation For Walking Robots*. Department of Computer Science, Carnegie-Mellon University, 1991.
[69] P.-C. Lin, H. Komsuoglu, and D. E. Koditschek, "A leg configuration measurement system for full-body pose estimates in a hexapod robot," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 411–422, 2005.
[70] X. Da, O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle, "From 2d design of underactuated bipedal gaits to 3d implementation: Walking with speed tracking," *IEEE Access*, vol. 4, pp. 3469–3478, 2016.
[71] P.-C. Lin, H. Komsuoglu, and D. E. Koditschek, "Sensor data fusion for body state estimation in a hexapod robot with dynamical gaits," *IEEE Trans. Robot.*, vol. 22, no. 5, pp. 932–943, 2006.
[72] S. P. Singh, P. J. Csonka, and K. J. Waldron, "Optical flow aided motion estimation for legged locomotion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2006, pp. 1738–1743.
[73] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. Remy, and R. Siegwart, "State estimation for legged robots: Consistent fusion of leg kinematics and IMU," in *Proc. Robot.: Sci. Syst. Conf.*, 2012.
[74] M. F. Fallon, M. Antone, N. Roy, and S. Teller, "Drift-free humanoid state estimation fusing kinematic, inertial and lidar sensing," in *IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 112–119.
[75] X. Xinjilefu, S. Feng, W. Huang, and C. G. Atkeson, "Decoupled state estimation for humanoids using full-body dynamics," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2014, pp. 195–201.
[76] R. Hartley, J. Mangelson, L. Gan, M. G. Jadidi, J. M. Walls, R. M. Eustice, and J. W. Grizzle, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2018, pp. 4422–4429.
[77] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Trans. Robot.*, vol. 33, no. 2, pp. 249–265, 2016.
[78] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4507–4514, 2019.
[79] ——, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2020, pp. 392–398.
[80] Y. Kim, B. Yu, E. M. Lee, J.-h. Kim, H.-w. Park, and H. Myung, "Step: State estimator for legged robots using a preintegrated foot velocity factor," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 4456–4463, 2022.
[81] Y. Gao, C. Yuan, and Y. Gu, "Invariant extended Kalman filtering for hybrid models of bipedal robot walking," *IFAC-PapersOnLine*, vol. 54,

no. 20, pp. 290–297, 2021.
[82] S. Yang, Z. Zhang, Z. Fu, and Z. Manchester, "Cerberus: Low-drift visual-inertial-leg odometry for agile locomotion," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2023, pp. 4193–4199.
[83] M. Focchi, V. Barasuol, I. Havoutis, J. Buchli, C. Semini, and D. G. Caldwell, "Local reflex generation for obstacle negotiation in quadrupedal locomotion," in *Nature-Inspired Mobile Robotics*. World Scientific, 2013, pp. 443–450.
[84] S. A. Fakoorian, D. Simon, H. Richter, and V. Azimi, "Ground reaction force estimation in prosthetic legs with an extended Kalman filter," in *Annual IEEE Systems Conference (SysCon)*. IEEE, 2016, pp. 1–6.
[85] G. Fink and C. Semini, "Proprioceptive sensor fusion for quadruped robot state estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2020, pp. 10 914–10 920.
[86] A. De Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the dlr-iii lightweight manipulator arm," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2006, pp. 1623–1630.
[87] S. Haddadin, A. Albu-Schaffer, A. De Luca, and G. Hirzinger, "Collision detection and reaction: A contribution to safe physical human-robot interaction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2008, pp. 3356–3363.
[88] J. Hwangbo, C. D. Bellicoso, P. Fankhauser, and M. Hutter, "Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2016, pp. 3872–3878.
[89] F. Jenelten, J. Hwangbo, F. Tresoldi, C. D. Bellicoso, and M. Hutter, "Dynamic locomotion on slippery ground," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4170–4176, 2019.
[90] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch *et al.*, "Anymal-a highly mobile and dynamic quadrupedal robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2016, pp. 38–44.
[91] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2018, pp. 4399–4406.
[92] M. Camurri, M. 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 Robot. Autom. Lett.*, vol. 2, no. 2, pp. 1023–1030, 2017.
[93] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[94] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.
[95] N. Rotella, S. Schaal, and L. Righetti, "Unsupervised contact learning for humanoid estimation and control," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2018, pp. 411–417.
[96] S. Piperakis, S. Timotheatos, and P. Trahanias, "Unsupervised gait phase estimation for humanoid robot walking," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2019, pp. 270–276.
[97] K. J. Wu, C. X. Guo, G. Georgiou, and S. I. Roumeliotis, "Vins on wheels," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2017, pp. 5155–5162.
[98] H.-W. Chae, J.-H. Choi, and J.-B. Song, "Robust and autonomous stereo visual-inertial navigation for non-holonomic mobile robots," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9613–9623, 2020.
[99] X. Yu, S. Teng, T. Chakhachiro, W. Tong, T. Li, T.-Y. Lin, S. Koehler, M. Ahumada, J. M. Walls, and M. Ghaffari, "Fully proprioceptive slip-velocity-aware state estimation for mobile robots via invariant Kalman filtering and disturbance observer," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2023, pp. 1–8.
[100] L. Xiong, X. Xia, Y. Lu, W. Liu, L. Gao, S. Song, Y. Han, and Z. Yu, "IMU-based automated vehicle slip angle and attitude estimation aided by vehicle dynamics," *Sensors*, vol. 19, no. 8, p. 1930, 2019.
[101] R. Bajaj, S. L. Ranaweera, and D. P. Agrawal, "GPS: location-tracking technology," *Computer*, vol. 35, no. 4, pp. 92–94, 2002.
[102] D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose, "Mapping and localization with RFID technology," in *Proc. IEEE Int. Conf. Robot. and Automation*, vol. 1. IEEE, 2004, pp. 1015–1020.
[103] H. Chae and K. Han, "Combination of RFID and vision for mobile robot localization," in *International Conference on Intelligent Sensors, Sensor Networks and Information Processing*. IEEE, 2005, pp. 75–80.
[104] H. Zhang, C. Zhang, W. Yang, and C.-Y. Chen, "Localization and navigation using QR code for mobile robot in indoor environment," in *IEEE International conference on robotics and biomimetics (ROBIO)*. IEEE, 2015, pp. 2501–2506.
[105] S. J. Lee, J. Lim, G. Tewolde, and J. Kwon, "Autonomous tour guide robot by using ultrasonic range sensors and QR code recognition in indoor environment," in *IEEE International Conference on Electro/Information Technology*. IEEE, 2014, pp. 410–415.
[106] C. Yang and H.-R. Shao, "Wifi-based indoor positioning," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 150–157, 2015.
[107] A. H. Lashkari, B. Parhizkar, and M. N. A. Ngan, "Wifi-based indoor positioning system," in *International Conference on Computer and Network Technology*. IEEE, 2010, pp. 76–78.
[108] M. Ghaffari Jadidi, M. Patel, and J. V. Miro, "Gaussian processes online observation classification for RSSI-based low-cost indoor positioning systems," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2017, pp. 6269–6275.
[109] M. Ghaffari Jadidi, M. Patel, J. V. Miro, G. Dissanayake, J. Biehl, and A. Girgensohn, "A radio-inertial localization and tracking system with BLE beacons prior maps," in *Proc. IEEE Int. Conf. Indoor Positioning and Indoor Navigation*. IEEE, 2018, pp. 206–212.
[110] C. Zhang, M. Kuhn, B. Merkl, A. E. Fathy, and M. Mahfouz, "Accurate uwb indoor localization system utilizing time difference of arrival approach," in *IEEE Radio and Wireless Symposium*. IEEE, 2006, pp. 515–518.
[111] M. J. Kuhn, J. Turnmire, M. R. Mahfouz, and A. E. Fathy, "Adaptive leading-edge detection in uwb indoor localization," in *IEEE Radio and Wireless Symposium*. IEEE, 2010, pp. 268–271.
[112] L. Imsland, T. A. Johansen, T. I. Fossen, H. F. Grip, J. C. Kalkkuhl, and A. Suissa, "Vehicle velocity estimation using nonlinear observers," *Automatica*, vol. 42, no. 12, pp. 2091–2103, 2006.
[113] L. Imsland, H. F. Grip, T. A. Johansen, T. I. Fossen, J. C. Kalkkuhl, and A. Suissa, "Nonlinear observer for vehicle velocity with friction and road bank angle adaptation-validation and comparison with an extended Kalman filter," SAE Technical Paper, Tech. Rep., 2007.
[114] L. Imsland, T. A. Johansen, H. F. Grip, and T. I. Fossen, "On non-linear unknown input observers–applied to lateral vehicle velocity estimation on banked roads," *International Journal of Control*, vol. 80, no. 11, pp. 1741–1750, 2007.
[115] H. Guo, H. Chen, D. Cao, and W. Jin, "Design of a reduced-order nonlinear observer for vehicle velocities estimation," *IET Control Theory & Applications*, vol. 7, no. 17, pp. 2056–2068, 2013.
[116] T. A. Wenzel, K. Burnham, M. Blundell, and R. Williams, "Dual extended Kalman filter for vehicle state and parameter estimation," *Vehicle system dynamics*, vol. 44, no. 2, pp. 153–171, 2006.
[117] H. F. Durrant-Whyte, "An autonomous guided vehicle for cargo handling applications," *Int. J. Robot. Res.*, vol. 15, no. 5, pp. 407–440, 1996.
[118] S. Kwon, K. Yang, and S. Park, "An effective Kalman filter localization method for mobile robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2006, pp. 1524–1529.
[119] A. Welte, P. Xu, and P. Bonnifait, "Four-wheeled dead-reckoning model calibration using rts smoothing," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2019, pp. 312–318.
[120] L. Paull, S. Saeedi, M. Seto, and H. Li, "AUV navigation and localization: A review," *IEEE J. Ocean. Eng.*, vol. 39, no. 1, pp. 131–149, 2013.
[121] J. C. Kinsey, R. M. Eustice, and L. L. Whitcomb, "A survey of underwater vehicle navigation: Recent advances and new challenges," in *IFAC conference of manoeuvering and control of marine craft*, vol. 88. Lisbon, 2006, pp. 1–12.
[122] F. Maurelli, S. Krupiński, X. Xiang, and Y. Petillot, "AUV localisation: a review of passive and active techniques," *International Journal of Intelligent Robotics and Applications*, pp. 1–24, 2021.
[123] D. Rudolph and T. A. Wilson, "Doppler velocity log theory and preliminary considerations for design and construction," in *2012 Proceedings of IEEE Southeastcon*. IEEE, 2012, pp. 1–7.
[124] P. H. Milne, *Underwater acoustic positioning systems*. Gulf Publishing Co., Houston, TX, 1983.
[125] L. Huang, B. He, and T. Zhang, "An autonomous navigation algorithm for underwater vehicles based on inertial measurement units and sonar," in *2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR 2010)*, vol. 1. IEEE, 2010, pp. 311–314.
[126] A. Alcocer, P. Oliveira, and A. Pascoal, "Study and implementation of an ekf gib-based underwater positioning system," *Control engineering practice*, vol. 15, no. 6, pp. 689–701, 2007.
[127] G. C. Karras, S. G. Loizou, and K. J. Kyriakopoulos, "On-line state and parameter estimation of an under-actuated underwater vehicle using a modified dual unscented Kalman filter," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2010, pp. 4868–4873.

[128] Ø. Hegrenaes and O. Hallingstad, "Model-aided INS with sea current estimation for robust underwater navigation," *IEEE J. Ocean. Eng.*, vol. 36, no. 2, pp. 316–337, 2011.
[129] P. A. Miller, J. A. Farrell, Y. Zhao, and V. Djapic, "Autonomous underwater vehicle navigation," *IEEE J. Ocean. Eng.*, vol. 35, no. 3, pp. 663–678, 2010.
[130] G. S. Chirikjian, *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Springer Science & Business Media, 2011.
[131] B. Hall, *Lie groups, Lie algebras, and representations: an elementary introduction*. Springer, 2015, vol. 222.
[132] A. W. Long, K. C. Wolfe, M. J. Mashner, and G. S. Chirikjian, "The banana distribution is Gaussian: A localization study with exponential coordinates," in *Proc. Robot.: Sci. Syst. Conf.*, vol. 265, 2013.
[133] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 679–693, 2014.
[134] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, 2016.
[135] J. G. Mangelson, M. Ghaffari, R. Vasudevan, and R. M. Eustice, "Characterizing the uncertainty of jointly distributed poses in the Lie algebra," *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1371–1388, 2020.
[136] M. Brossard, A. Barrau, P. Chauchat, and S. Bonnabel, "Associating uncertainty to extended poses for on Lie group IMU preintegration with rotating earth," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 998–1015, 2022.
[137] B. Katz, J. Di Carlo, and S. Kim, "Mini Cheetah: A platform for pushing the limits of dynamic quadruped control," in *Proc. IEEE Int. Conf. Robot. and Automation*. IEEE, 2019, pp. 6295–6301.
[138] D. Kim, J. Di Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *arXiv preprint arXiv:1909.06586*, 2019.
[139] M. Grupp, "evo: Python package for the evaluation of odometry and SLAM." https://github.com/MichaelGrupp/evo, 2017.
[140] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, Oct. 2012.
[141] P. Cieślak, "Stonefish: An advanced open-source simulation tool designed for marine robotics, with a ros interface," in *OCEANS 2019 - Marseille*, jun 2019.
[142] M. Brossard, A. Barrau, and S. Bonnabel, "RINS-W: Robust inertial navigation system on wheels," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots and Syst.* IEEE, 2019, pp. 2068–2075.
[143] ——, "AI-IMU dead-reckoning," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 4, pp. 585–595, 2020.
[144] J. J. Leonard and A. Bahr, "Autonomous underwater vehicle navigation," *Springer handbook of ocean engineering*, pp. 341–358, 2016.

Tzu-Yuan (Justin) Lin received the B.S. degree in Mechanical Engineering from National Taiwan University, Taipei, Taiwan, in 2017 and the M.S. degree in Robotics from the University of Michigan, Ann Arbor, MI, USA, in 2020. He is currently a Ph.D. candidate in Robotics under the supervision of Professor Maani Ghaffari at the University of Michigan, Ann Arbor. His research interests include robot perception, simultaneous localization and mapping (SLAM), and computer vision.

Tingjun Li received the Master's degree in Robotics from the University of Michigan, Ann Arbor, MI, USA, in 2022. He was working as a Research Assistant in the CURLY Lab under the supervision of Professor Maani Ghaffari. His interests include simultaneous localization and mapping (SLAM) and the implementation of related algorithms. He is now working at Amazon.com, Inc. as a Software Dev Engineer.

Wenzhe Tong received the M.S.E. degree in Electrical and Computer Engineering from University of Michigan, Ann Arbor, MI, USA, in 2023. He was working in the CURLY Lab under the supervision of Professor Maani Ghaffari. He is currently a first-year Ph.D. pre-candidate in Robotics under the supervision of Professor Xiaonan Huang at the HDR Lab of the University of Michigan, Ann Arbor. His research interests include robot autonomy, simultaneous localization and mapping (SLAM), and state estimation.

Maani Ghaffari received the Ph.D. degree from the Centre for Autonomous Systems (CAS), University of Technology Sydney, NSW, Australia, in 2017. He is currently an Assistant Professor at the Department of Naval Architecture and Marine Engineering and the Department of Robotics, University of Michigan, Ann Arbor, MI, USA, where he directs the Computational Autonomy and Robotics Laboratory (CURLY). His work on sparse, globally optimal kinodynamic motion planning on Lie groups received the best paper award finalist title at the 2023 Robotics: Science and Systems conference. He is the recipient of the 2021 Amazon Research Awards. His research interests lie in the theory and applications of robotics and autonomous systems.