

Appendix – Legolas: Deep Leg-Inertial Odometry

In this Appendix, we include additional details about the following:

- A Further details regarding the training of Legolas such as improvements to the chosen architecture and details of data collection.
- B Elaborations of the loss used to train Legolas.
- C Explanation of the steps taken to improve deployment of the baseline methods and details of the deployment of Legolas.
- D Additional trajectories visualizations on the validation dataset and in real-world deployments.

A Further Training and Data Collection Details of Legolas

Training and data collection details are provided to improve the reproducibility of Legolas.

A.1 Training Details

During deployment, it was found that the default 1D ResNet architecture [37] had too many parameters, causing slower than desired model prediction on real-world hardware. Fine-tuning of the model, by reducing the number of residual layers from 3 to 2 decreased the number of parameters in the network from 4.7M to 1.5M allowing for successful deployment. As shown in Tab. 3, this led to no substantial change in the relevant metrics.

#	Model Parameter Sized	RPE@1m ↓	ATE _u ↓
1	4.7M	0.045	0.017
2	1.5M (Ours)	0.050	0.016

Table 3: **Effect of Model Size on Validation Performance.** While decreasing the size of the model reduced model performance in RPE@1m, ATE_u improved slightly when using the smaller model. Due to the changes not substantially changing the relevant metrics, but boosting on-board speed, the smaller model was utilized for deployment.

A.2 Data Collection

The only difference in Legolas for training the pose prediction model \mathcal{P} for the Go2 and Lite3 occurs during dataset collection, the ‘urdf’ file that describes the robot’s kinematics is changed for the respective robots.

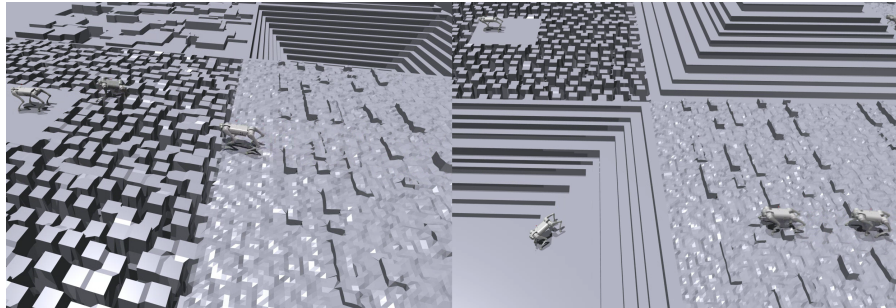


Figure 5: Snapshot of robots collecting trajectories in simulation as performed in Sec. 3.1.

To collect a given trajectory, a robot follows its policy $\hat{\pi}$ with a random velocity command with a target linear velocity uniformly sampled from $[-2.0, 2.0] \frac{m}{s}$. A target linear velocity between $[-0.15, 0.15] \frac{m}{s}$ is instead set to 0 in order to add standing and rotating in-place behaviors into our

dataset. A target angular velocity in yaw is then uniformly sampled from $[-\pi, \pi] \frac{rad}{s}$. If an angular velocity is chosen between $[-0.05, 0.05] \frac{rad}{s}$, the angular velocity is set to 0 to create more walking forward behaviors in our dataset as this is a common motion in the real world. This large range of linear and angular velocities covers the full range of motions possible with $\hat{\pi}$. A visualization of the training environment can be found in Fig. 5.

B Training Loss Details

#	Rotation Representation	RPE@1m ↓	ATE _u ↓
1	Yaw-Pitch-Roll	0.233	0.023
2	6D [45] (Ours)	0.050	0.016

Table 4: **6D rotation improves validation metrics.** Predicting changes in the 6D rotation produced better odometry prediction (RPE@1m decreased by 78.5%) when compared to predicting the changes in yaw, pitch, and roll.

During training and deployment, different rotation representations were utilized. Using a 6D representation [45] during training improved performance on relevant metrics on the validation split as demonstrated in Tab. 4. More specifically, during training, ΔP_R is a 6D representation, and the ground-truth incremental motion of the robot is converted into a 6D representation. Otherwise, during deployment and data collection, a matrix representation is utilized.

C Robotics Experiment Details

Legolas is deployed on real-world hardware with a frequency of 50 Hz to match the frequency used in the simulator. However, with the optimizations provided in Sec. A.1, the model is capable of being run at up to 600 Hz on the Jetson Orin Nano.

During deployment of Legolas, the incremental motion of the robot is predicted at every time step. To estimate the trajectory of the robot, these predictions are accumulated. After converting the output of the models to an $SE(3)$ matrix at some arbitrary step i , S_i , the state at the current step, j , can be computer as $S = S_j S_{j-1} \dots S_0$.

C.1 Visual-Inertial Odometry Deployment Details



Figure 6: **Quick motions and large changes in rotation occurring over one second.** These quick motions cause degraded image captures on the quadraped affecting the deployment of the visual baseline. Changes in pitch (Fig. 6a,6b,6c,6d,6e) and roll (Fig. 6a and Fig. 6e) are notable.

VINS-Fusion found successful deployment when carefully tuned using tools including ‘imu_utils’ [39, 46] and ‘Kalibr’ [40]. imu_tools is utilized to retrieve relevant IMU statistics used by both the EKF and VINS-Fusion baselines such as the gyroscope’s and accelerometer’s bias and noise. Kalibr was used to find the extrinsic matrix of the camera frame with respect to the IMU frame and to find the timing offset between the camera and the IMU. With these steps, more manual fine-tuning of the parameters was required for deployment.

However, direct deployment to outdoor environments of the previously tuned and measured parameters failed. We found that quick motions and rotations such as the one visualized in Fig. 6 became

434 exaggerated on a quadruped robot. Furthermore, interactions between the sun and the onboard cam-
435 era caused degraded depth prediction and tracking. Using a polarized film with a metal shield around
436 the camera reduced this error mode, but still didn't lead to consistent results as shown in Fig. 7.



(a) Stereo camera without shield (b) Stereo camera with shield and (c) Stereo camera with shield and
and full cloud coverage. full cloud coverage. no cloud coverage.

Figure 7: A camera shield is necessary for VINS-Fusion deployment. Without the camera shield, the stereo camera equipped on the robot fails to capture the scene due to interference. However, we still find issues with outdoor deployment of VINS-Fusion if the environment is sunny even when equipped with shielding.

437 D Additional Trajectory Rollouts

438 **Additional validation rollouts:** Fig. 8 demonstrates the superior trajectory reconstruction of using
 439 the full sensing suite rather than just the IMU. The use of the full sensing suite allows Legolas to
 440 directly predict the incremental motions of the robot rather than rely on analytical models of the
 441 robot.

442 **Additional real-world rollouts:** Supplementing the rollout given in Fig. 3a, additional rollouts are
 443 visualized. Fig. 9 demonstrates the robot moving in a straight line in an indoor scene. In this
 444 scenario Legolas tracks the straight line while VINS-Fusion suffers from losing visual tracking due
 445 to reflections on the floor. Fig. 10 demonstrates the robot moving in a double circular pattern
 446 in an indoor environment. In this scenario, Legolas is able to track the ground-truth. The EKF
 447 baseline is capable of reproducing a similar shape up until the 1st completed loop where its trajectory
 448 reconstruction degrades.

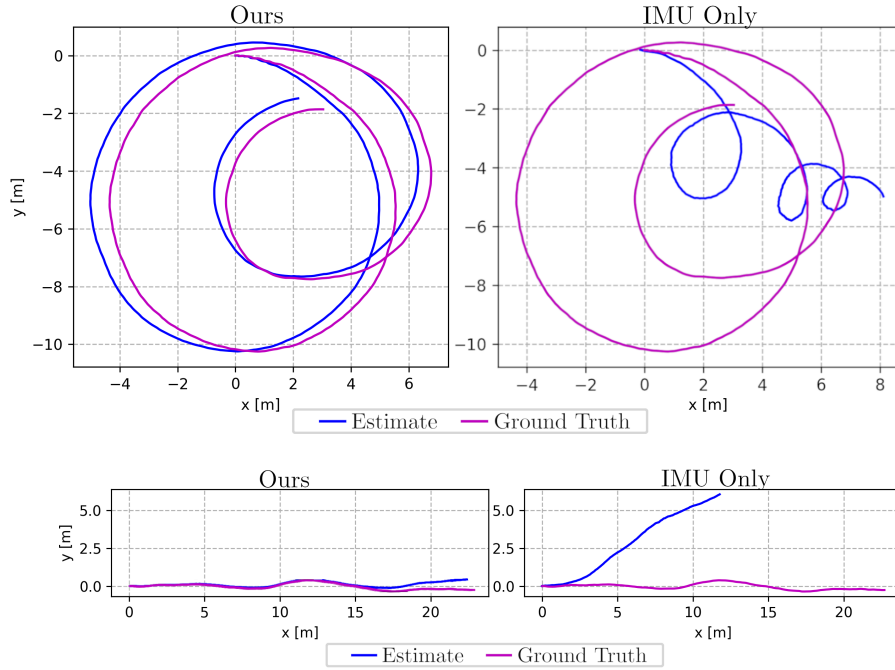


Figure 8: **Top-Down visualization of rollouts of Legolas trained with different sensing modalities.** Legolas demonstrates improved odometry prediction through its use of the full sensing suite on the robot. Previous work has attempted to estimate displacements through learning with only the IMU sensor, this produces inferior predictions.

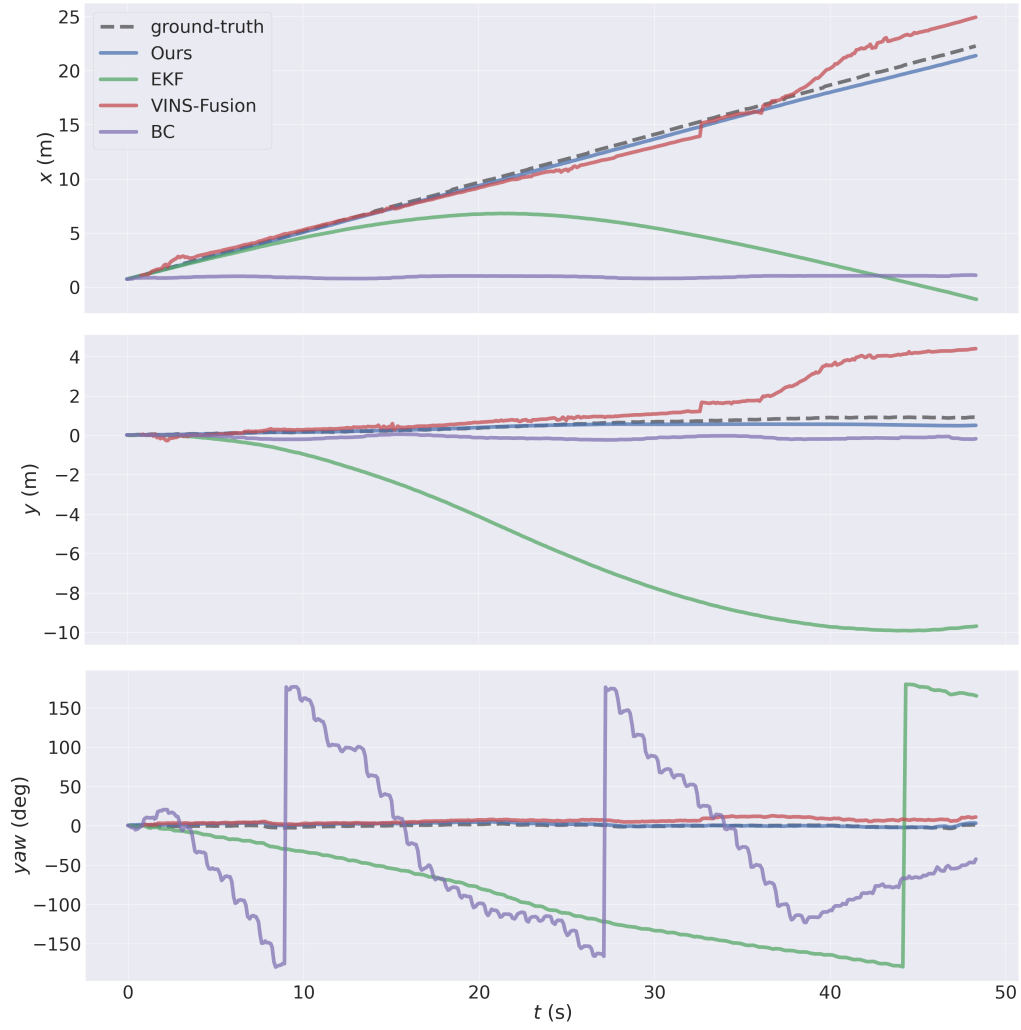


Figure 9: **Straight line rollout.** In this scenario the robot moves in a straight line for approximately 23 meters. Legolas tracks the trajectory, while VINS-Fusion suffers from losing visual tracking around $t(s) = 37$ and the estimated position of the robot jumps.

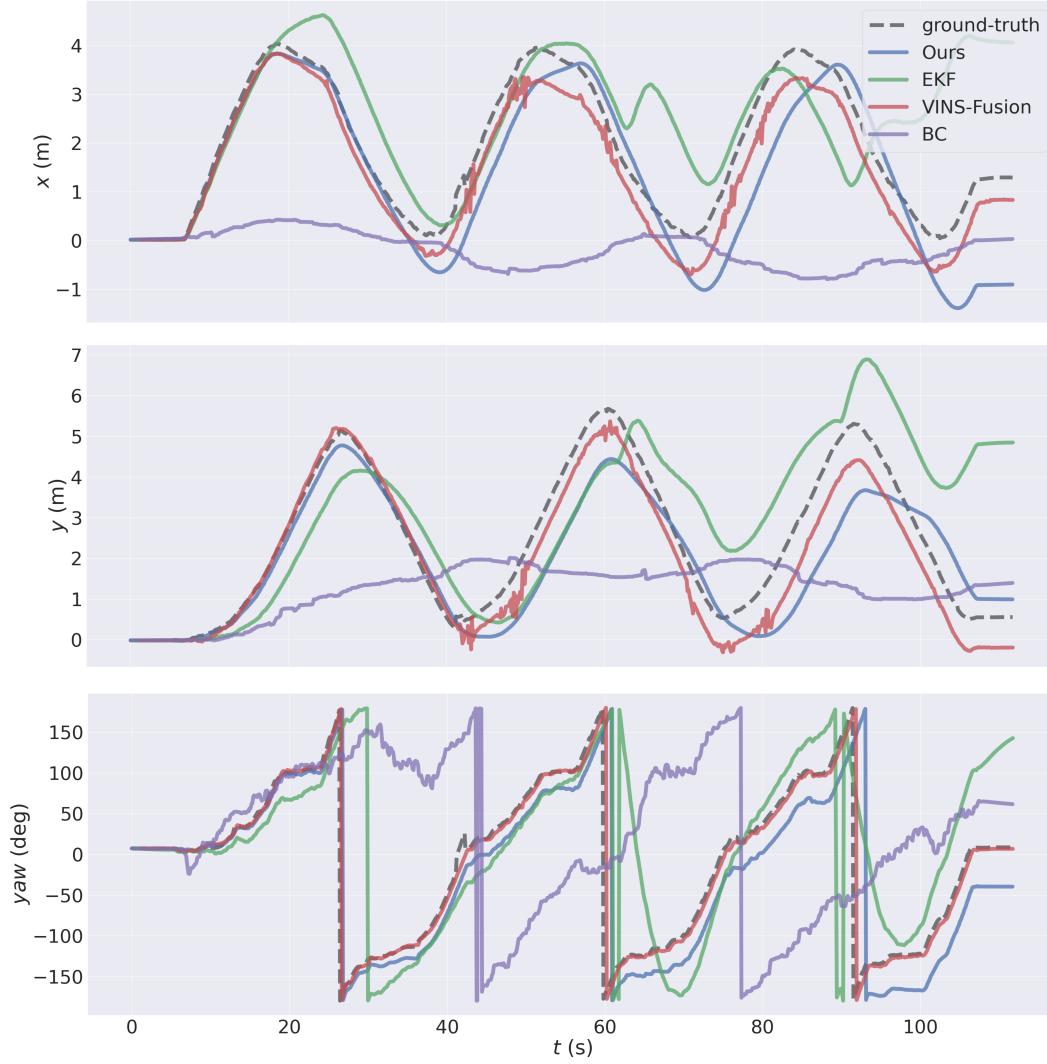


Figure 10: **Circular rollout.** In this scenario the robot moves in a circle two times, both VINS-Fusion and Legolas are capable of tracking the ground-truth trajectory. EKF tracks closely until the first loop is completed. BC is able to track the heading, but poorly track x and y positions.

References

- [1] F. Fraundorfer and D. Scaramuzza. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robotics & Automation Magazine*, 2012.
- [2] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza. Pampe: Perception-aware model predictive control for quadrotors. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018.
- [3] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000.
- [4] S. Herath, H. Yan, and Y. Furukawa. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods. In *2020 IEEE international conference on robotics and automation (ICRA)*, pages 3146–3152. IEEE, 2020.
- [5] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and N. Trigoni. Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference. *IEEE Internet of Things Journal*, 7(5):4431–4441, 2020.
- [6] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.
- [7] T. Qin, P. Li, and S. Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.
- [8] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry for ground vehicle applications. *Journal of Field Robotics*, 23(1):3–20, 2006.
- [9] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.
- [10] T. Shan, B. Englot, C. Ratti, and R. Daniela. Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5692–5698. IEEE, 2021.
- [11] D. Youm, H. Oh, S. Choi, H. Kim, and J. Hwangbo. Legged robot state estimation with invariant extended kalman filter using neural measurement network. *arXiv preprint arXiv:2402.00366*, 2024.
- [12] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart. State estimation for legged robots on unstable and slippery terrain. In *IROS*, 2013.
- [13] D. Wisth, M. Camurri, and M. Fallon. Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots. *IEEE Transactions on Robotics*, 39(1):309–326, 2022.
- [14] D. Wisth, M. Camurri, and M. Fallon. Robust legged robot state estimation using factor graph optimization. *IEEE Robotics and Automation Letters*, 4(4):4507–4514, 2019.
- [15] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis, K. Daniilidis, V. Kumar, and J. Engel. Tlio: Tight learned inertial odometry. *IEEE Robotics and Automation Letters*, 2020.
- [16] C. Chen, X. Lu, A. Markham, and N. Trigoni. Ionet: Learning to cure the curse of drift in inertial odometry. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
- [17] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon. Learning inertial odometry for dynamic legged robot state estimation. In *CoRL*, 2022.
- [18] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged locomotion in challenging terrains using egocentric vision. In *CoRL*, 2023.

- [19] A. Kumar, Z. Fu, D. Pathak, and J. Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.
- [20] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter. Learning a state representation and navigation in cluttered and dynamic environments. *RAL*, 2021.
- [21] X. Cheng, K. Shi, A. Agarwal, and D. Pathak. Extreme parkour with legged robots. *arXiv preprint arXiv:2309.14341*, 2023.
- [22] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. on Robotics*, 2016.
- [23] G. Welch, G. Bishop, et al. An introduction to the kalman filter. 1995.
- [24] V. Dhédin, H. Li, S. Khorshidi, L. Mack, A. K. C. Ravi, A. Meduri, P. Shah, F. Grimmering, L. Righetti, M. Khadiv, et al. Visual-inertial and leg odometry fusion for dynamic locomotion. In *ICRA*, 2023.
- [25] S. Yang, Z. Zhang, Z. Fu, and Z. Manchester. Cerberus: Low-drift visual-inertial-leg odometry for agile locomotion. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4193–4199. IEEE, 2023.
- [26] 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*, 17:17–24, 2013.
- [27] N. Rotella, M. Bloesch, L. Righetti, and S. Schaal. State estimation for a humanoid robot. In *IROS*, 2014.
- [28] M. Maravagakis, D.-E. Argiropoulos, S. Piperakis, and P. Trahanias. Probabilistic contact state estimation for legged robots using inertial information. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12163–12169. IEEE, 2023.
- [29] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim. Contact model fusion for event-based locomotion in unstructured terrains. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4399–4406. IEEE, 2018.
- [30] J. Hwangbo, C. D. Bellicoso, P. Fankhauser, and M. Hutter. Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3872–3878. IEEE, 2016.
- [31] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and N. Trigoni. Oxiod: The dataset for deep inertial odometry. *arXiv preprint arXiv:1809.07491*, 2018.
- [32] H. Yan, Q. Shan, and Y. Furukawa. Ridi: Robust imu double integration. In *Proceedings of the European conference on computer vision (ECCV)*, pages 621–636, 2018.
- [33] G. Cioffi, L. Bauersfeld, E. Kaufmann, and D. Scaramuzza. Learned inertial odometry for autonomous drone racing. *IEEE Robotics and Automation Letters*, 8(5):2684–2691, 2023.
- [34] R. Buchanan, V. Agrawal, M. Camurri, F. Dellaert, and M. Fallon. Deep imu bias inference for robust visual-inertial odometry with factor graphs. *IEEE Robotics and Automation Letters*, 8(1):41–48, 2022.
- [35] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot parkour learning. *arXiv preprint arXiv:2309.05665*, 2023.

- 358 [36] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin,
359 A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for
360 robot learning. *arXiv preprint arXiv:2108.10470*, 2021.
- 361 [37] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*,
362 2016.
- 363 [38] S. Macenski and I. Jambrecic. Slam toolbox: Slam for the dynamic world. *Journal of Open*
364 *Source Software*, 2021.
- 365 [39] O. J. Woodman. An introduction to inertial navigation. Technical report, University of Cam-
366 bridge, Computer Laboratory, 2007.
- 367 [40] P. Furgale, J. Rehder, and R. Siegwart. Unified temporal and spatial calibration for multi-sensor
368 systems. In *IROS*. IEEE, 2013.
- 369 [41] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evalua-
370 tion of rgb-d slam systems. In *IROS*, 2012.
- 371 [42] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evalu-
372 ation of rgb-d slam systems. *2012 IEEE/RSJ International Conference on Intelligent Robots*
373 *and Systems*, pages 573–580, 2012.
- 374 [43] S. Umeyama. Least-squares estimation of transformation parameters between two point pat-
375 terns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 1991.
- 376 [44] M. Labbé and F. Michaud. Rtab-map as an open-source lidar and visual simultaneous local-
377 ization and mapping library for large-scale and long-term online operation. *JFR*, 2019.
- 378 [45] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in
379 neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern*
380 *recognition*, 2019.
- 381 [46] G. Wenliang. imu_utils. https://github.com/gaowenliang/imu_utils, 2018.