DNN-based SLAM Tracking Error Online Estimation

Tianyi Hu
tianyi.hu@duke.edu
Duke University
Durham, NC, USA

Tim Scargill
ts352@duke.edu
Duke University
Durham, NC, USA

Ying Chen
ying.chen151@duke.edu
Duke University
Durham, NC, USA

Guohao Lan
g.lan@tudelft.nl
Delft University of Technology
Delft, the Netherlands

Maria Gorlatova
maria.gorlatova@duke.edu
Duke University
Durham, NC, USA

ABSTRACT

Simultaneous localization and mapping (SLAM) takes in sensor data, e.g., camera frames, and estimates the user's trajectory while creating a map of the surrounding environment. However, existing SLAM evaluation methods are not reference-free, requiring ground-truth trajectories collected from external systems that are infeasible for most scenarios. In this demo, we present Deep SLAM Error Estimator (DeepSEE), a framework that collects features from a standard visual SLAM pipeline as multivariate time series and uses an attention-based neural network to estimate the tracking error at run time. We evaluate DeepSEE in a game engine-based virtual environment, which generates the visual input for DeepSEE and provides the ground-truth trajectory. Demo participants can navigate the virtual environment to create their own trajectories and view the online pose error estimation. This demo showcases how DeepSEE can act as a quality-of-service indicator for downstream applications.

CCS CONCEPTS

• **Human-centered computing** → *Ubiquitous and mobile computing systems and tools.*

KEYWORDS

SLAM, pose tracking, tracking error, error estimate

ACM Reference Format:
Tianyi Hu, Tim Scargill, Ying Chen, Guohao Lan, and Maria Gorlatova. 2023. DNN-based SLAM Tracking Error Online Estimation. In *The 29th Annual International Conference on Mobile Computing and Networking (ACM MobiCom '23), October 2–6, 2023, Madrid, Spain.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3570361.3614082

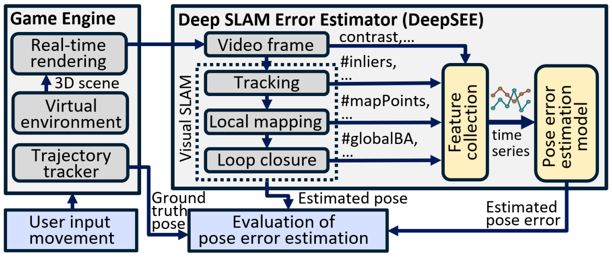

Figure 1: **Our demo system architecture, which showcases DeepSEE evaluated in a virtual environment to calculate the estimated pose and corresponding pose error at run time.**

1 INTRODUCTION

Simultaneous localization and mapping (SLAM), the process of mapping an environment while concurrently tracking the pose of a device within that environment, is employed across various mobile systems, including augmented reality headsets, unmanned aerial vehicles, and autonomous cars [1, 4, 13]. However, even state-of-the-art SLAM algorithms exhibit pose-tracking errors due to challenging input data, e.g., textureless environments. The magnitude of these errors can vary dramatically, ranging from as low as a few millimeters to greater than one meter [3, 4, 7, 9].

Existing evaluation methods require external pose measurements as ground-truth references to analyze the performance of SLAM pose tracking. Traditionally, tracking evaluations [3, 7, 14] are performed by comparing estimated trajectories to the ground-truth poses obtained from optical tracking systems (e.g., in [8, 12]), but considerable setup and calibration time for each new environment makes them impractical for most scenarios. Recent work [2] has shown promise in estimating the absolute trajectory error (ATE) from input data characteristics. However, this work does not implement online estimations. In addition, compared with relative error (RE), which evaluates sub-trajectories and thus provides a fine-grained evaluation of pose tracking, ATE is a coarse metric for the overall trajectory and thus cannot act as a timely evaluation at run time [14].

ACM MobiCom '23, October 2–6, 2023, Madrid, Spain

ACM ISBN 978-1-4503-9990-6/23/10.
https://doi.org/10.1145/3570361.3614082

We present Deep SLAM Error Estimator (DeepSEE), a DNN-based solution which is *the first to provide real-time, online, reference-free estimates of short-term SLAM tracking error* (in terms of the sub-trajectories used to calculate RE). We implement DeepSEE on top of a state-of-the-art visual SLAM algorithm, ORB-SLAM3 [4]. Figure 1 illustrates our demo pipeline, showing that DeepSEE can be seamlessly integrated into existing mobile systems with minimal overhead, and provides estimated pose error as a quality-of-service indicator for downstream applications.

2 SYSTEM DESIGN

Figure 1 shows our demo system architecture. We build a virtual environment in a game engine, which renders camera frames based on user input. Visual SLAM takes in camera frames to estimate the user's pose; meanwhile, DeepSEE collects features from the SLAM pipeline and uses a pose error estimation model to estimate the pose error.

Virtual environment: We evaluate DeepSEE in a game engine-based virtual environment. The game engine monitors user inputs from a keyboard that allows users to explore virtual environments and renders camera frames in first-person perspective as the visual input for DeepSEE. Compared with a real-world evaluation, the virtual environment makes the ground-truth trajectory more easily accessible and allows us to quickly adapt to various environments by loading different game engine settings [5, 6, 10].

Visual SLAM: This module estimates the user's trajectory and builds a map of the surrounding environment based on camera frames. It has three main modules, tracking, local mapping, and loop closure, which conduct different levels of optimization for pose estimation and mapping [1, 4].

Feature collection: We implement this module on top of a visual SLAM pipeline to collect features from the visual input and the internal status of the three main SLAM modules, including image contrast, matched inliers, local bundle adjustment error, and global bundle adjustment error, from the visual input, the tracking, local mapping, and loop closure modules, respectively. The features are sampled at the camera frame rate and recorded over a fixed period, forming multivariate time series for pose error estimation. To reduce the data collection overhead and latency, we implement this module using multi-threading, allowing concurrent feature collection through shared memory.

Pose error estimation model: We formulate the pose error estimation task as a supervised multivariate time series regression problem, whose features are the multivariate time series we collect from the visual SLAM pipeline and its visual input. We assign the pose errors between the estimated trajectory and the ground-truth trajectory to the time series as their labels. The model architecture uses multi-head attention layers to extract features and uses fully connected layers for regression. To train our model, we generate our training set from both virtual environments and real-world SLAM benchmarks, including the SenseTime SLAM benchmark [7].

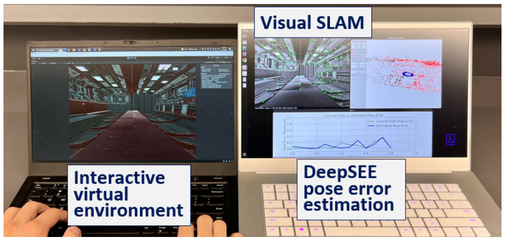
Visual SLAM
Interactive virtual environment
DeepSEE pose error estimation

Figure 2: **DeepSEE in action. Demo participants interact with a virtual environment; DeepSEE estimates the pose error of visual SLAM running in the virtual environment.**

3 INTERACTIVE DEMONSTRATION

The demonstration follows the pipeline shown in Figure 1. It allows participants to gain insights into 1) how their movements in the virtual environment and visual input characteristics impact the SLAM tracking error, and 2) how the proposed multivariate time series regression model estimates the pose error. We provide an annotated demo video online.[1]

The virtual environment is built in a game engine, Unity 2021.3.8f1 [11]. We run DeepSEE in Ubuntu 22.04 on a Razer laptop (CPU: Intel i7-12800H; GPU: Nvidia RTX3080Ti). As shown in Figure 2, participants use a keyboard to move around the virtual environment, in the first-person perspective rendered by the game engine. DeepSEE takes in the camera frames generated during a participant's interaction with the virtual environment to run visual SLAM and collect features to estimate pose error. It takes the ground-truth trajectory and the trajectory estimated by the visual SLAM algorithm to compute and visualize ground-truth pose error.

Participants can compare their movements in the virtual environment with pose error results to understand how motion and visual input data characteristics impact SLAM performance, and how the proposed pose error estimation model performs compared with the ground-truth pose error. In our evaluations, we measure our model's performance on trajectories $A0 \sim A7$ of the SenseTime SLAM benchmark [7], with leave-one-out cross-validation. DeepSEE estimates the relative pose error with a root mean square error (RMSE) of 8.45 cm, which outperforms the baseline [2], a random forest regression method with RMSE of 13.81 cm, by 38.8%.

ACKNOWLEDGMENTS

This work was supported in part by NSF grants CSR-1903136, CNS-1908051, and CNS-2112562, NSF CAREER Award IIS-2046072, a Meta Research Award, and a CISCO Research Award.

[1] https://sites.duke.edu/tianyihu/publications/demo/mobicom23/

REFERENCES

[1] Ali J Ali, Zakieh Sadat Hashemifar, and Karthik Dantu. 2020. Edge-SLAM: Edge-assisted visual simultaneous localization and mapping. In *Proc. ACM MobiSys*.

[2] Islam Ali, Bingqing Wan, and Hong Zhang. 2023. Prediction of SLAM ATE using an ensemble learning regression model and 1-D global pooling of data characterization. *arXiv preprint arXiv:2303.00616* (2023).

[3] Michael Burri, Janosch Nikolic, Pascal Gohl, Thomas Schneider, Joern Rehder, Sammy Omari, Markus W Achtelik, and Roland Siegwart. 2016. The EuRoC micro aerial vehicle datasets. *The International Journal of Robotics Research* 35, 10 (2016), 1157–1163.

[4] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. 2021. ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM. *IEEE Transactions on Robotics* 37, 6 (2021), 1874–1890.

[5] Ying Chen, John Sarik, Hazer Inaltekin, and Maria Gorlatova. 2023. Demo Abstract: Demonstrating resource-efficient SLAM in virtual spacecraft environments. In *Proc. IEEE INFOCOM*.

[6] James Garforth and Barbara Webb. 2019. Visual appearance analysis of forest scenes for monocular SLAM. In *Proc. IEEE ICRA*.

[7] Li Jinyu, Yang Bangbang, Chen Danpeng, Wang Nan, Zhang Guofeng, and Bao Hujun. 2019. Survey and evaluation of monocular visual-inertial SLAM algorithms for augmented reality. *Virtual Reality & Intelligent Hardware* 1, 4 (2019), 386–410.

[8] NaturalPoint Inc. 2023. OptiTrack - Motion capture systems. https://optitrack.com/.

[9] Tong Qin, Peiliang Li, and Shaojie Shen. 2018. VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics* 34, 4 (2018), 1004–1020.

[10] Tim Scargill, Ying Chen, Nathan Marzen, and Maria Gorlatova. 2022. Integrated design of augmented reality spaces using virtual environments. In *Proc. IEEE ISMAR*.

[11] Unity Technologies. 2023. Unity Real-Time Development Platform - 3D, 2D, VR & AR Engine. https://unity.com/.

[12] Vicon Motion Systems Ltd. 2023. Vicon - Motion capture systems. https://www.vicon.com/.

[13] Jingao Xu, Hao Cao, Zheng Yang, Longfei Shangguan, Jialin Zhang, Xiaowu He, and Yunhao Liu. 2022. SwarmMap: Scaling up real-time collaborative visual SLAM at the edge. In *USENIX NSDI*.

[14] Zichao Zhang and Davide Scaramuzza. 2018. A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In *Proc. IEEE/RSJ IROS*.