Towards a Robustness/Resilience-Aware Simultaneous Localization and Mapping (SLAM)

by

Islam Alaaeldin Mustafa Ali

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

University of Alberta

Abstract

In this thesis, we investigate the problem of robustness and resilience in simultaneous localization and mapping systems (SLAM). With the vast adoption of robotics in many industries and disciplines, robustness and resilience are becoming of immense importance for the reliable and safe deployment of robotics in real-world settings. The current literature in SLAM treats accuracy as a synonym for both robustness and resilience. In this thesis, we start by providing a rigorous and formal definition for robustness and resilience as two major objectives in modern robotics, which reveals the following requirements for achieving them: (1) quantitative characterization of operating conditions, (2) objective evaluation of SLAM, (3) predictability of performance, and (4) real-time fault detection and performance monitoring. Thus, the thesis is divided into four parts addressing each of the aforementioned requirements.

Robustness and resilience are tightly coupled to measurable operating conditions in which a robot is deployed. Due to the difficulty of SLAM evaluation in the real-world, researchers utilize datasets for that purpose. These datasets implicitly include the operating condition at which the reported performance is guaranteed. Thus, our first proposal addresses this problem by introducing a generic and extensible framework for the quantitative characterization of SLAM datasets and benchmarks. The proposed system automatically characterizes datasets based on defined metrics that collectively represent the operating conditions imposed by an environment or robot motion. Additionally, the proposed system automatically and systematically provides analysis of

datasets on the measurement, sequence, and dataset level, which can be used to push the boundaries of SLAM evaluation. Moreover, it analyzes the correlation between metrics and SLAM performance, revealing the sensitivity of a given algorithm to many environmental conditions.

Current evaluation methodologies in SLAM are very redundant in nature and do not take into consideration the characteristics of the deployment environment. To that end, our second proposal tackles this point by proposing a tabulation-based dynamic programming algorithm that utilizes the quantitative characterization of SLAM datasets to achieve two goals, which are: elimination of redundancy in the evaluation pool of data sequences, and objective selection of the most proper subset of sequences to ensure coverage of the deployment conditions.

Moreover, we address the problem of performance predictability in SLAM in our third proposal by employing a supervised ensemble learning regression model to predict SLAM errors on the pose level and the trajectory level. We utilize the concept of sub-trajectories for diversifying the number of training samples and apply a 1-D global average pooling function for dimensionality reduction. Additionally, we ensure the efficacy of the method with limited training data and analyzes out-of-distribution predictions.

Fault detection is critical for resilience in SLAM since it highlights the need to engage recovery mechanisms to converge when divergence is detected. Therefore, our fourth proposal uses a decoupled calibrated IMU-based kinematics model as a high-accuracy supervisory monitoring signal for SLAM. The method relies on a modified version of DUET, a deep learning IMU calibration method, to provide reliable IMU measurements. Then, consistency is measured between IMU-based and SLAM pose streams and is then used as an indicator of faults. Moreover, the proposed method is non-invasive and algorithm agnostic since it has no assumptions on the SLAM system itself.

Preface

Research in this thesis was conducted under the supervision of Prof. Hong Zhang at the Department of Computing Science, University of Alberta. Some extracts from this thesis appear in the following publications and preprints.

- **Chapter 3:** Islam Ali and Hong Zhang. "Are we ready for robust and resilient slam? A framework for quantitative characterization of slam datasets". In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 2810–2816. IEEE, 2022.

- **Chapter 4:** Islam Ali and Hong Zhang. "Optimizing slam evaluation footprint through dynamic range coverage analysis of datasets". In 2023 Seventh IEEE International Conference on Robotic Computing (IRC), pages 127–134, 2023.

- **Chapter 5:** Islam Ali, Selina Wan, and Hong Zhang. "Prediction of slam ate using an ensemble learning regression model and 1-D global pooling of data characterization". In 21st Conference on Robots and Vision (CRV), 2024. **(Best paper award in Robotics at CRV 2024)**

- **Chapter 6:** Islam Ali and Hong Zhang. "A Decoupled Calibrated IMU-based Approach For Performance Degradation Monitoring in VI-SLAM". *(Preprint, In Progress)*

To my dad, my mom, and my wife

For the unwavering support and love. I wouldn't have done this without you.

Acknowledgements

I am deeply grateful to my supervisor, Prof. Hong Zhang, for his unwavering support, guidance, and mentorship throughout my PhD journey. His expert insights, constructive feedback, and encouragement have been invaluable in shaping my research and enhancing its quality. This work would not have been possible without his dedicated supervision and continuous support.

I am also grateful to Prof. Martin Jagersand and Prof. Nilanjan Ray, members of my supervisory committee, for their invaluable support and scholarly guidance. Their comments and discussions have played a pivotal role in shaping the direction of this research.

I extend my heartfelt thanks to my friends and colleagues who have supported me throughout my PhD. I deeply appreciate their contributions, which have enriched this research endeavor.

Last but not least, I would like to express my profound gratitude to my family, who have always been there for me through thick and thin. I thank my father, Mr. Alaaeldin, and mother, Mrs. Azza, for their infinite prayers and belief in my abilities to reach greater heights. To my wife, Yara, for her unwavering support, patience, and belief in my dreams – she has been my greatest motivation. I am forever grateful for her steadfast allyship, standing beside me with unwavering strength and resilience. My gratitude also goes to my brother, Khaled, and my sister, Nada, for their encouragement, understanding, and companionship. Your presence in my life has made this journey richer and more meaningful. And above all, I thank Allah for all the blessings in my life.

Contents

List of Tables

List of Figures

Glossary

2D Two Dimensional

3D Three Dimensional

APE Absolute Pose Error

ATE Absolute Trajectory Error

BA Bundle Adjustment

CNN Convolutional Neural Network

CR Contrast Ratio

DP Dynamic Programming

EKF Extended Kalman Filter

FAST Features from Accelerated Segment Test

FOV Field of View

GAP Global Average Pooling

H Entropy

ICP Iterative Closest Point

IMU Inertial Measurement Unit

KF KeyFrame

LC Least Cost

LE Least Experiments

LiDAR Light Detection and Ranging

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

NeRF Neural Radiance Fields

ORB Oriented FAST and Rotated BRIEF

PF Particle Filter

PMCC Pearson's Product Moment Correlation Coefficient

RANSAC Random Sample Consensus

RGB-D Red, Green, Blue, Depth

RMSE Root Mean Square Error

ROS Robot Operating System

RPE Relative Pose Error

SDI Simpson Diversity Index

SFM Structure from Motion

SIFT Scale-Invariant Feature Transform

SLAM Simultaneous Localization and Mapping

UAV Unmanned Aerial Vehicle

VI-SLAM Visual-Inertial Simultaneous Localization and Mapping

VO Visual Odometry

V-SLAM Visual Simultaneous Localization and Mapping

Chapter 1

Introduction

1.1 Motivation and Problem Statement

Simultaneous localization and mapping (SLAM) is considered a standard and fundamental building block in many modern robotic systems due to its ability to achieve two critical goals, which are accurate localization of the robot in the observed environment while building a map of that environment at the same time [192]. Currently, robotics is being used in a wide spectrum of applications such as autonomous driving, material handling, search and rescue missions, among many others. Due to the increase in the adoption of robotics in these applications and much more, the reliability of SLAM becomes more critical due to its responsibility to provide subsequent control or navigation tasks with accurate estimates for both the localization and mapping tasks. The accuracy and reliability of these estimates when a robot is faced with a challenging situation are crucial to ensure successful and safe deployment of robotics in the real world.

The literature divides the history of SLAM into three major ages where each one witnessed a theme to the work and effort done to solve the SLAM problem [24]. The first was *the classical age* (1986-2004), where the problem was formulated and filtering-based methods were introduced. This was followed by *algorithm analysis age* (2004-2015), where the focus was directed to analyze the formerly introduced SLAM algorithms and the study of observability, convergence, and consistency of SLAM estimates. Afterwards, we entered *the robust perception age* (2004-now), where SLAM became task-driven, which

highlighted the need for robust and resilient SLAM and also the need for long-term operation in real-world settings.

Despite claims that the SLAM problem is solved, many argue that it evolves into other problems that are more fundamental and critical at the same time [42] with the introduction of new deployment requirements, new supporting technologies, and more advanced computational power. As outlined in [24], robustness and resilience are among the major fundamental requirements in modern robotic systems due to their deployment in a wide spectrum of settings. Emphasis on this idea was also provided in [50], where both robustness and resilience were recognized as some of the main challenges facing today's robotic field and one of the current open problems that we shall direct our research efforts towards. In order to achieve robustness/resilience or measure them in SLAM, a proper definition for them must be achieved first by, for example, exploring how they are defined in other disciplines in science and engineering. For instance, biological systems [173][54], systems engineering [179], and control engineering [172][175] define robustness as the ability of the system to maintain performance under measured perturbations. On the other hand, psychology and ecology [68] [108], mechanical and physical robotic systems [124] [87], and system engineering [193] [27] define resilience as the convergence of a system after divergence while operating outside of its nominal perturbation limits. Perturbations are defined as any external conditions causing the system to deviate from its equilibrium state.

In the SLAM literature, accuracy is used as an indication of robustness and resilience. However, the aforementioned definitions disagree with this line of thinking and suggest that accuracy cannot be used to measure robustness or resilience without the quantitative characterization of operational conditions in which the system will be deployed as well as the performance of the system is guaranteed.

Thus, a distinction between the three definitions is an essential first step that shall precede any discussion of how to make SLAM systems robust or resilient. Based on the aforementioned definitions in closely related fields in science and engineering, one can deduce the following definitions that accurately

distinguish each term. **Accuracy in SLAM** is the measurable performance of the SLAM system under perturbations, which is not a direct indication of either robustness or resilience if such perturbations are not defined, and if the relation between the performance and these perturbations is not identified.

On the other hand, **robustness in SLAM** is defined to be the ability of a SLAM system to maintain its guaranteed performance under defined perturbations. It is typically concerned with the performance of SLAM *within* its nominal operating conditions or perturbations levels. While **Resilience in SLAM** can be defined as the ability to detect faults and to converge to a defined performance level after divergence due to the operation *outside* of its nominal operating conditions or perturbation levels. As shown, both objectives (robustness and resilience) are tightly coupled to defined and measured perturbation levels to which a SLAM system is subject. In SLAM, accuracy is usually measured on benchmark datasets which implicitly contain these defined perturbation levels. Thus, the following requirements can be drawn from the two definitions to address both robustness and resilience in SLAM.

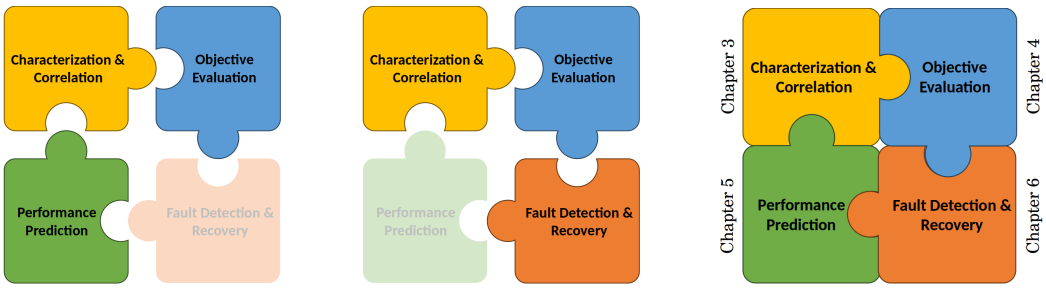
1. Characterization of perturbations and their correlation to SLAM performance highlighting SLAM sensitivity to certain conditions.

2. Objective evaluation of SLAM where deployment conditions are incorporated in evaluation procedures so that we can safely deploy SLAM systems.

3. Predictability of performance to allow the judgment of the suitability of a system to a given deployment setting that was not included in the evaluation process but characterized and measured sufficiently.

4. Fault detection to engage recovery mechanism to ensure convergence of SLAM to nominal defined performance boundaries when faced with out-of-boundary conditions in the real world.

Although numerous SLAM algorithms and benchmark datasets are proposed in the literature, a big gap is yet to be filled to address the aforementioned requirements. This is due to the lack of a formal and rigorous definition



(a) (b) (c)

Figure 1.1: Requirements for (a) robustness and (b) resilience in SLAM, which form our contributions in this thesis as depicted in (c) linked to their respected chapters in this thesis

for robustness and resilience. In this thesis, we seek to bridge this gap by systematically addressing the problem of robustness and resilience in SLAM with solutions to the major requirements for robustness and resilience, where these requirements are directly extracted from a formal and rigorous definition of both terms in the SLAM context.

1.2 Contributions

The contributions of this thesis are as follows:

- We propose a generic and extendable framework for the quantitative characterization of SLAM datasets and benchmarks, which is essential for measuring robustness and resilience in SLAM systems. Utilizing RGB cameras and IMUs, our methodology derives various characterization metrics, creating a modular framework that easily accommodates new metrics, sensors, or datasets.

- We address the problem of objective evaluation and dataset redundancy reduction in SLAM by proposing a tabulation-based dynamic programming (DP) algorithm to select the optimal subset of sequences for testing and evaluating SLAM based on objective criteria. Our method, evaluated on both single-objective and multi-objective criteria, demonstrates superiority in optimality and time complexity compared to baseline methods such as random search and greedy-based algorithms.

- We tackle the problem of performance predictability in SLAM as an integrity measure, essential for determining the suitability of a SLAM algorithm for deployment based on its performance in previously evaluated environments. We propose an ensemble learning regression model to predict the absolute trajectory error (ATE) and the absolute pose error (APE) respectively. Additionally, we study the reliability of this method when limited training data is available.

- We discuss fault detection in SLAM using a decoupled IMU-based kinematics model as an external supervisory signal for real-time consistency checks and early fault detection. Employing a modified DUET method for IMU calibration, we minimize measurement errors and drift. Evaluation covers four fault indicators, shows the IMU model's reliability in the short-term, and includes a comparison with traditional SLAM fault detection techniques, as well as an analysis of sensitivity to history size and error thresholds.

1.3 Thesis Outline

The remainder of this thesis is organized as follows. In Chapter 2, we provide an overview of relevant background and a detailed discussion of the work related to this thesis. We highlight prior work done to address the aforementioned requirements as well as the gap in the literature that we seek to fill. After that, in Chapter 3, we propose our framework for quantitative characterization of SLAM datasets, which addresses the problem of robustness and resilience evaluation and measurement. Then, in Chapter 4, we discuss our proposal for objective evaluation of SLAM using a tabulation-based dynamic programming algorithm. In Chapter 5, we address the problem of predicting different SLAM errors at both the pose level and the trajectory level using an ensemble learning regression methodology. This is followed by the proposal of an algorithm-agnostic method for fault detection in SLAM using a decoupled calibrated IMU-based kinematics model that is used to measure the consistency of SLAM estimates in Chapter 6. The conclusion of this thesis and

directions for future work are explained in Chapter 7.

Chapter 2

Background and Related Work

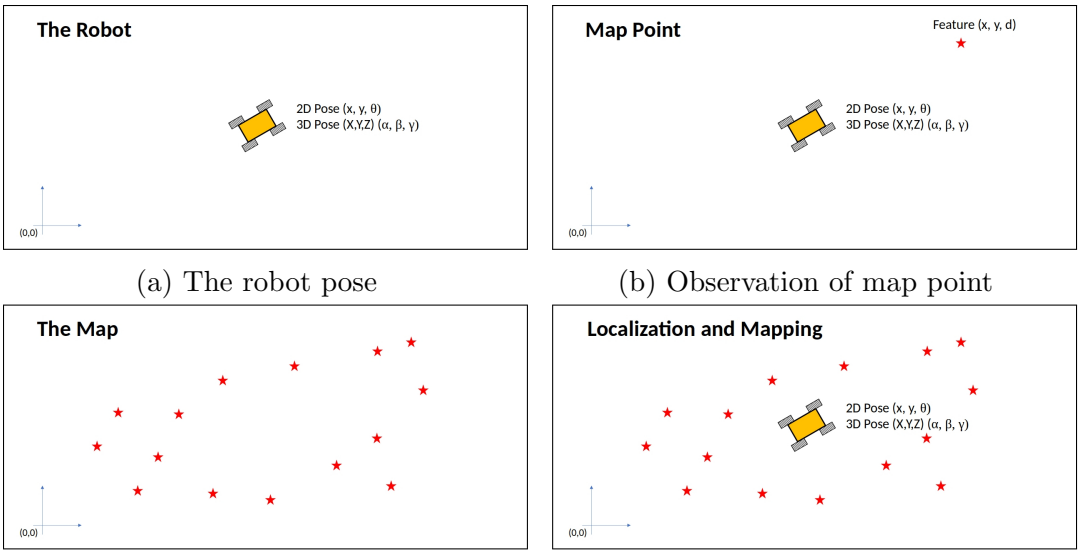
In Chapter 1, we provided a definition for robustness and resilience in SLAM and extracted some fundamental requirements to achieve both robustness and resilience. As highlighted, despite the availability of some previous efforts to address these important requirements, these efforts fall short when it comes to either generality, completeness, or efficacy.

In this chapter, we provide a background of related concepts in SLAM followed by a detailed literature review of related work. We highlight the gaps in the literature for each of the aforementioned requirements mentioned in Chapter 1 which are, operating conditions characterization as a way to measure robustness and resilience and their correlation to the SLAM performance, objective evaluation of SLAM and coverage analysis, predictability of SLAM performance, and finally fault detection and recovery in SLAM.

2.1 The SLAM Problem

As mentioned earlier, Simultaneous Localization and Mapping (SLAM) is the process of building a map of the environment and localizing the robot in this environment at the same time [24]. The requirement of the map is crucial for accurate localization of the robot, as it allows the robot to reset its error when a location in the map is revisited through loop closure [195]. This process is illustrated in Figure 2.1, which shows the dual objective of SLAM compared to localization-only or mapping-only systems.

Given a robot traversing a trajectory in an environment, we define the



(a) The robot pose

(b) Observation of map point

(c) Map is a collection of observations

(d) SLAM

Figure 2.1: Steps to achieve SLAM where a robot in 2D or in 3D seeks to estimate its location inside a map that is being built at the same time compared to single-task systems

following [192] representing a single state of the robot, landmarks, or observations:

- $\mathbf{x}$: defines a robot pose/state in the trajectory traversed.
- u: represents an odometry constraint between two robot poses.
- m_i: represents the location of landmark i.
- z_i: represents the observation for the location of landmark i.

Subsequently, we define the following sets, representing the history of SLAM variables over the traversed trajectory so far.

- $\chi_k = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k\}$: represents the history of robot poses.
- $U_k = \{u_1, u_2, ..., u_k\}$: represents the history of odometry vectors.
- $m = \{m_1, m_2, ..., m_n\}$: represents the locations of all available landmarks.
- $Z_k = \{z_{(i,1)}, z_{(i,2)}, ..., z_{(i,k)}\}$: represents the history of observations of landmark locations at each previously visited robot pose x_i.

Given the aforementioned parameters, we seek to find the correct poses of the robot throughout the trajectory traversed as well as the locations of different landmarks observed at the same time. Through the history of SLAM, multiple solutions has been proposed to solve this problem efficiently. In this section, we give a brief history of the problem, highlighting major milestones and paradigm shifts as well as the different methodologies used to solve the SLAM problem efficiently. After that, we explore emerging trends in SLAM and how the problem is evolving to meet modern requirements in the real world. Finally, we ask the important question of whether SLAM is a solved problem or not, which is a reflection of why more work is still needed in SLAM.

2.1.1 Evolution of SLAM and The Rise of Visual-SLAM (VSLAM)

In the evolution of SLAM, there exist a clear distinction between traditional SLAM methods that relies on range sensing and filtering-based solutions, and Visual SLAM (VSLAM) in their approach to handling dimensionality and environment complexity. As stated, traditional SLAM methods employ filtering-based techniques, such as Extended Kalman Filter (EKF), to handle high dimensional state spaces. This category of methods were originally developed to deal with uncertainties in the filter state, where sensors provided limited information. For instance, in EKF SLAM, the estimator maintains a probabilistic estimates of the robot position as well as each point in the map. Afterwards and iteratively, it updates these estimates as new sensor measurements arrives [192]. Although these methods showed effectiveness in estimating uncertainties, it remains impractical due to its intensive computational requirements and struggles with large-scale environments which suggest that this group of methods are not scalable in nature [117].

A significant paradigm shift occurred with the rise of Visual SLAM (VSLAM), which was driven by advances in multi-view geometry techniques as well as the advances in visual sensing technologies. With the development of scale-invariant feature transform (SIFT) [102], a key technology became available for robust feature detection and matching across multiple views, which

is fundamental to VSLAM. This technique and others provided VSLAM with the ability to leverage visual data more effectively and provided richer information that can be utilized for localization and mapping [159]. Additionally, Structure-from-Motion (SFM) techniques have witnessed significant advances and offered solutions for reconstructing 3D points from 2D images [66]. Early filtering-based approaches to SLAM, such as Mono-SLAM [43], utilized these principles. However, they were limited in terms of depth estimation and scalability due to the inherited limitations of filtering-based methods.

The integration of multi-view geometry and Structure-from-Motion techniques in SLAM provided major advances to the solution of the VSLAM problem. This was exemplified in the introduction of PTAM [83] which incorporates geometric insights in SLAM for more accurate and scalable feature tracking and mapping. This transition from filtering-based methods to geometry-based methods was further validated in [176], where the superior performance of geometry-based techniques over filtering-based methods was proven in the comparative study conducted. This study paved the way to new optimization frameworks such as g2o [90], which builds upon these technologies.

2.1.2 Graph-based SLAM

Traditional SLAM architecture relied on Extended Kalman Filters (EKF) [182] or Particle Filters (PF) [181] to solve the SLAM problem. However, such methods suffered from accuracy, consistency, and scalability issues [44]. Thus, graph-based SLAM was introduced to bridge the gap and overcome the shortcomings of previously introduced methods.

Modern SLAM architectures depend on dividing the SLAM pipeline into two major modules: the front-end and the back-end. The front-end is responsible for feature extraction and both short and long-term data association, while the back-end formulates SLAM estimation as a *Maximum A Posteriori (MAP)* estimation problem that consumes the information produced by the front-end. Figure 2.2 provides an illustration of the SLAM pipeline.

The relation between different SLAM variables (robot states, loop closure, odometry, and landmark positions) is typically modeled in the form of a factor

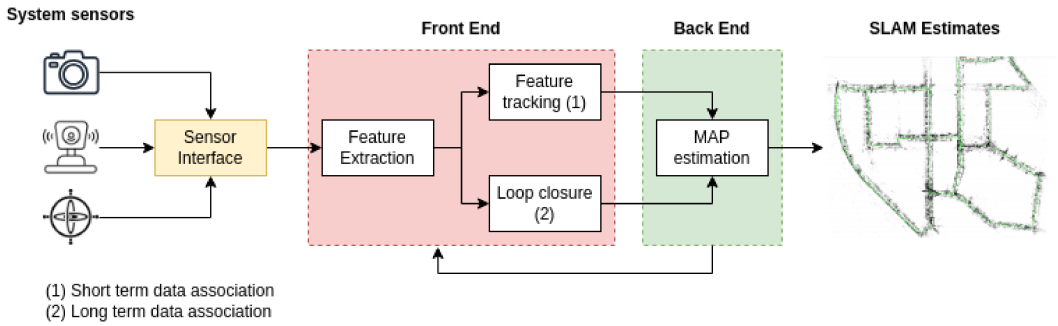


Figure 2.2: An illustration of SLAM constraints as a factor graph where trajectory poses and landmarks are connected via constraints from odometry or loop closure

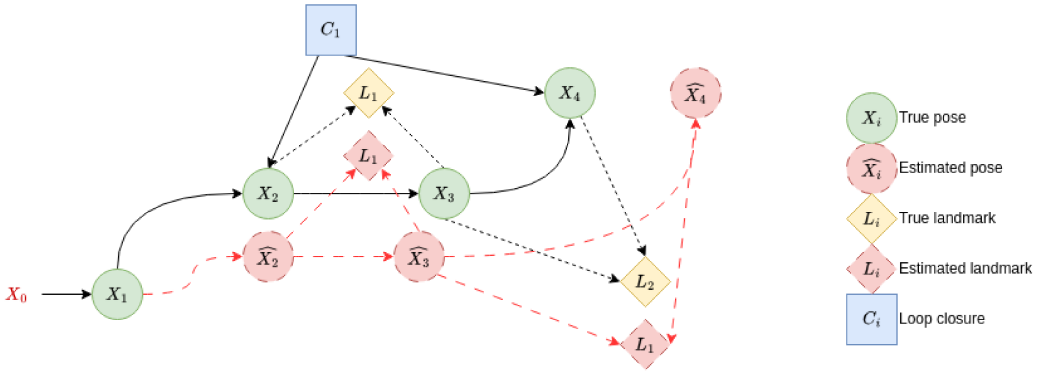


Figure 2.3: An illustration of SLAM constraints as a factor graph where trajectory poses and landmarks are connected via constraints from odometry or loop closure

graph, where the dependency is preserved by graph edges. This formulation exploits sparsity and provides an efficient representation for MAP estimation. Figure 2.3 provides an illustration of the SLAM factor graph and shows how robot states can suffer from drifts with the lack of loop closure constraints optimization.

The MAP back-end seek to estimate the variable χ, which represents a set of variables χ_i that corresponds to a robot pose. Given a set of measurements $Z = \{z_k, k = 1, .., m\}$, where each measurement z_k is given by:

$$z_k = h_k(\chi_k) + \epsilon_k \tag{2.1}$$

where χ_k is a subset of robot poses ($\chi_k \subseteq \chi$), $h_k(.)$ is the measurement model, and ϵ_k is the measurement noise. Thus, we seek to estimate the variable χ^*,

which maximizes the belief over χ given the measurements Z, which can be formulated as:

$$\chi^* = \arg\max_{\chi} P(\chi) \prod_{k=1}^{m} P(z_k|\chi_k) \tag{2.2}$$

following the derivation in [24], we end up with the following formulation of the MAP estimation:

$$\chi^* = \arg\min_{\chi} \sum_{k=1}^{m} ||h_k(\chi_k) - z_k||^2_{\Omega_k} \tag{2.3}$$

2.1.3 Localization and Mapping Representation

The outcome of SLAM is twofold: the pose of the robot and the map of the environment observed. The representation of these two outcomes is governed by the operation of the robot in either 2-dimensional space or 3-dimensional space, and the requirements imposed by either subsequent control algorithms or deployment limitations.

2.1.3.1 Robot Pose

In 2D SLAM, the pose of a robot at time k can be represented using the Special Euclidean Group SE(2). The group SE(2) describes the set of all possible 2D transformations, which include both translation and rotation. Thus, a robot pose $\mathbf{x}$ in 2D SLAM at time k is represented by:

$$\mathbf{x}_k = [x_k \ y_k \ \theta_k]^T \tag{2.4}$$

where x_k and y_k are the Cartesian coordinates of the robot in 2-D space, and θ_k is the yaw angle relative to a reference frame.

On the other hand, in 3D SLAM, the pose of a robot at time k is represented using the Special Euclidean Group SE(3). SE(3) describes the set of all possible 3D transformations, which include translation and rotation in three dimensions. Therefore, a robot pose $\mathbf{x}$ in 3D SLAM at time t can be given by:

$$\mathbf{x}_k = [x_k \ y_k \ z_k \ q_x \ q_y \ q_z \ q_w]^T \tag{2.5}$$

Where x_k, y_k, z_k are the Cartesian coordinates of the robot in 3-D space, and q_x, q_y, q_z, q_w are the quaternion components representing the robot orientation.

2.1.3.2 Estimated Map

From the mapping perspective, numerous representations were utilized to represent the estimated map. These representations differ in their memory requirements and the level of detail achieved. Table 2.1 compares different mapping representations, highlighting key differences and examples of each method. Additionally, Table 2.2 briefly compares the pros and cons of each mapping method.

Table 2.1: A detailed comparison of different mapping method used in SLAM

#	Method	2D/3D	Definition	Example
1	**Grid-based maps**	Both	- Divides the environment into a set of grid cells (2D) or voxels (3D) - Each cell/voxel holds the probability of occupancy by an object	[65]
2	**Feature-based maps**	Both	- Uses features keypoints to represent the environment	[25]
3	**Point cloud maps**	3D	- Uses dense point clouds from LiDAR or depth cameras	[203]
4	**Semantic maps**	Both	- Add semantic labels to traditional maps	[111]
5	**Topological maps**	2D	- Representation of the environment as a graph - Nodes in the graph are robot poses, and edges are navigational path	[114]
6	**Hybrid maps**	Both	- Combines different map types to get the best of both worlds	[69]

Table 2.2: A detailed comparison of the pros and cons of different mapping method used in SLAM

#	Method	Pros	Cons
1	**Grid-based maps** [65]	- Simple representation - Good for obstable avoidance	- Performance is grid resolution-dependent - Not suitable for dynamic envionrment
2	**Feature-based maps** [25]	- Efficient representation	- Environment dependent. - Fails in textureless environment.
3	**Point cloud maps** [203]	- Accurate mapping of the environment - Represents surface geometries	- High computational cost - Hard to manipulate and process
4	**Semantic maps** [111]	- Useful for complex manipulation tasks	- Complex pipeline with object recognition
5	**Topological maps** [114]	- Very compact representation - Suitable for navigation tasks	- Very abstract representation - Challenges in dynamic environments
6	**Hybrid maps** [69]	- More diverse and detailed mapping - Balance pros/cons of different methods	- Complex implementation - Integration challenges

2.1.4 Key Metrics for SLAM Evaluation

The performance of SLAM can be measured by running a SLAM algorithm against a pre-recorded dataset. That is due to the difficulty of having repeatable experiments in the real-world and the lack of resources to achieve that. A dataset consists of sensor measurements as well as ground truth information of the actual robot pose and map structure. The performance of SLAM is achieved by comparing the outcomes of SLAM to available ground truth data to calculate how far SLAM estimates are from target outcomes. Due

to the dual objectives of SLAM (localization and mapping), the evaluation of performance is done on each objective separately.

2.1.4.1 Localization Evaluation

Two metrics are highly accepted and utilized in the community for evaluating the localization performance of SLAM: the *Absolute Trajectory Error (ATE)* and the *Relative Pose Error (RPE)* [205]. Both metrics require the availability of ground-truth localization information and proper alignment and scaling of both the ground-truth data and SLAM outcomes.

1. **Absolute Trajectory Errors (ATE)** is the de facto method for measuring performance, which is the root mean square of the errors between corresponding absolute pose errors (APE). The APE for translation and rotation are given by:

$$APE_{trans}(k) = ||\hat{T}_i - T_i|| \tag{2.6}$$

$$APE_{rot}(k) = \hat{R}_i^{\ T} R_i \tag{2.7}$$

   where $APE_{trans}(k)$ and $APE_{rot}(k)$ are translational and rotational errors at time step k, respectively.

   Consequently, the Absolute trajectory error (ATE) for translation and rotations can be calculated by:

$$ATE_{trans} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} ||\hat{T}_i - T_i||^2} \tag{2.8}$$

$$ATE_{rot} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} ||\log(\hat{R}_i^{\ T} R_i)||^2} \tag{2.9}$$

   where ATE_{trans} and ATE_{rot} are translational and rotational absolute trajectory errors over a trajectory of size N.

2. **Relative Trajectory Errors (RPE)** exploits the effect of the length of the trajectory on localization accuracy by calculating the root mean square error (RMSE) of APEs over a fixed time interval Δt. Varying the

time interval Δt produces multiple error numbers, and thus, statistical operators such as mean, median, and standard deviation are used to report the overall performance of SLAM or visual odometry modules.

2.1.4.2 Mapping Evaluation

Due to the availability of multiple representation methods for mapping in SLAM, a unified method for the evaluation of mapping performance is not available. Rather, each representation requires a different evaluation strategy that stems from the nature of the representation itself. Additionally, SLAM researchers and practitioners tend to use localization accuracy as an indication of mapping accuracy based on the fact that accurate localization is not possible if the mapping is not accurate as well. This assumption may not hold true if the map is being used by subsequent tasks for any control or manipulation missions. Thus, more work is needed in that direction to formally evaluate different mapping representations.

2.1.5 Emerging Technologies in SLAM

The SLAM architecture evolves with the introduction of new technologies, making use of their performance and allowing the extension of its utilization beyond traditional applications. In this section, we discuss two major emerging technologies that are currently being used in SLAM and are transforming its performance and structure.

2.1.5.1 Deep Learning in SLAM

Due to the great success of deep learning (DL) algorithms in a wide spectrum of vision-based applications, SLAM researchers have directed their attention to the study of how to harness this success for SLAM [116]. Thus, the usage of DL in SLAM was divided into two categories. The first category is end-to-end SLAM, where the whole SLAM pipeline is replaced with a deep neural network. The input is typically the unprocessed sensor measurements, while the output is the estimated robot poses and environment map. For instance, DeepSLAM [95] proposed an unsupervised learning pipeline that consists of

a number of DNNs, where each performs a specific task of SLAM, such as tracking, mapping, and loop closure. The algorithm depends on using stereo pairs of images as inputs but operates in monocular mode when in the inference phase. DeepSLAM showed very promising performance compared to traditional SLAM algorithms when evaluated on publicly available datasets.

The second category of contributions is concerned with replacing a single module in the traditional SLAM pipeline with a DL counterpart. This achieves a hybrid pipeline where we can leverage the accuracy of DL modules while keeping the robustness of traditional non-DL methods [116]. These proposals included the usage of deep feature extraction modules [94], deep loop closure modules [10], and vision-based pose estimation [186].

2.1.5.2 Semantic SLAM

Another emerging technology in SLAM is the utilization and/or production of semantic information within the traditional pipeline of SLAM. The availability of rich semantic information can greatly enhance the performance of both localization and mapping, which impacts any subsequent application in which SLAM is used [33]. Additionally, a number of SLAM systems have been proposed where a semantic map is generated in contrast to traditional maps [32]. This opens doors to the deployment of SLAM in a wider space of applications where such capabilities are of great importance.

2.1.5.3 Neural Radiance Fields and Gaussian Splatting in SLAM

Neural Radiance Fields (NeRF) [113] is a novel method in the field of computer vision and 3D reconstruction for representing and rendering 3D scenes from a set of 2D images. It leverages deep neural network models to generate detailed 3D scenes by learning a continuous volumetric scene function from a set of 2D images.

Another representation of 3D scenes is Gaussian Splatting [81], which is also a novel method for rendering and reconstructing 3D scenes. It represents a scene using a set of Gaussian splats rather than traditional polygons or voxel grids.

Integrating Neural Radiance Fields (NeRF) and Gaussian Splatting into SLAM systems offers complementary benefits for 3D mapping and localization. NeRF provides high-quality, detailed reconstructions and novel view synthesis, which enhances scene understanding and visual fidelity but comes with high computational demands and extensive training requirements. In contrast, Gaussian Splatting offers efficient, real-time 3D representation with smooth, continuous surfaces, making it suitable for fast, dynamic SLAM applications [109]. While NeRF excels in generating detailed maps for high-resolution applications, Gaussian Splatting's computational efficiency supports real-time performance and large-scale data handling, making both techniques valuable depending on the specific needs of the SLAM system.

2.2 Related Work

Study of the robustness and resilience problem in SLAM is tightly coupled to the requirements implied in their definitions. Thus, in this section, we thoroughly review the literature to highlight key contributions that are related the problem of robustness and resilience in SLAM. We divide our literature review into three main categories: the characterization of SLAM datasets as a way to determine operating conditions of SLAM, prior work in data analysis and coverage as a way to optimize the evaluation process, and finally the problem of fault detection and recovery as a main requirement for robustness and resilience in modern SLAM systems.

2.2.1 SLAM Datasets and Benchmarks

A wide range of SLAM datasets and benchmarks have been introduced to the SLAM research community in the last decade. These classical set of SLAM datasets that is usually used to evaluate and compare SLAM performance include and is not limited to: the KITTI odometry benchmark [59], TUM Visual-Inertial dataset [163], and EuroC MAV dataset [22], among many others. Recently, a great attention and many trials were directed towards pushing the limits of SLAM through exposing them to very challenging situations and

scenarios. A number of these challenging datasets were gathered using physical systems such as: The UZH-FPV Drone Racing Dataset[45] and the blackbird dataset [9] to name a few, while others were synthetic rendered data produced using modern game engines. The improvements witnessed in photo-realistic simulators such as: CARLA[48] and FlighMare [171], and the existence of sophisticated game engines with advanced physics and graphics capabilities such as: Unreal Engine [52]; both directed the attention of SLAM datasets creators to the usage of these engines/simulators to create more challenging conditions for SLAM that are hard to capture with a physical system. For instance, the TartanAir dataset [187] is a photo-realistic synthetic dataset that was built using the Unreal game engine [52] with the objective of pushing the limits of SLAM systems by exposing SLAM solutions to a number of challenging scenarios in terms of lighting, aggressive motion profiles, texture-less scenes, among many other conditions.

In this subsection, we provide a comparison of an extended number of SLAM datasets (55 SLAM datasets/benchmarks) in terms of the dataset size, data acquisition conditions, and sensory configuration. Additionally, the comparison includes qualitative comparison of the environment conditions and the carrier used for data capturing. The comparisons mentioned in this section are qualitative and are based on the information available in the original publication of each dataset.

2.2.1.1 The Taxonomy of SLAM Datasets and Benchmarks

SLAM datasets and benchmarks differ in a wide spectrum of aspects. For instance, they differ in their sensory configuration, size, availability of ground truth data, data collection methods, environment in which the data was collected, sensory configuration carrier, among many other aspects. Despite the introduction of new datasets each year with a steady trend, one can observe a similarity in terms of the qualitative aspects of most of the introduced datasets. As shown in Figure 2.5 and Figure 2.4, one can clearly see that diversity is only witnessed in the type of carrier used for data acquisition, the type of environment (indoor vs. outdoor), and the sensory configuration. On the contrary,

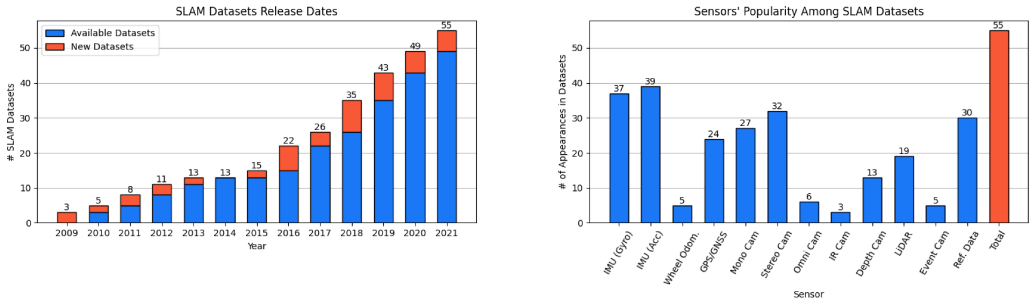


Figure 2.4: Yearly tracking of dataset releases and availability of different SLAM sensors in datasets/benchmarks

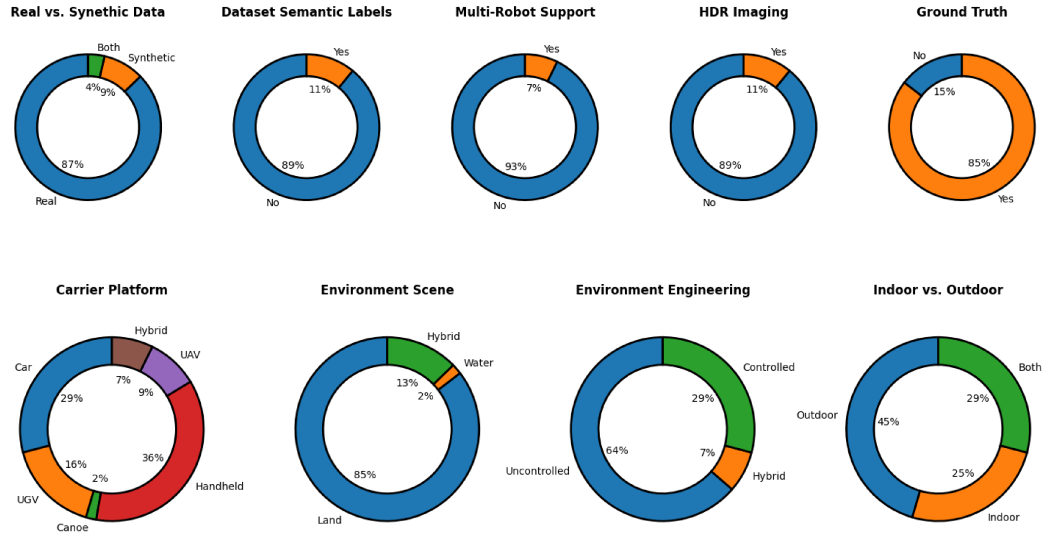


Figure 2.5: A pie chart illustrating the different configurations available in SLAM datasets/benchmarks

other aspects and qualitative parameters did not have the same diversity.

A closer look at the different qualitative aspects of SLAM datasets provides a clear perspective towards the trends in SLAM algorithms themselves. For instance, we can observe a special interest given to visual and visual-inertial datasets due to the increased interest in those sensors and their wide usage in SLAM as standard sensors. The same perspective can be observed with the increase of outdoor dataset with the emergence of SLAM algorithms directed towards autonomous driving and autonomous vehicles. The concept remains applicable as well when we look at single-robot vs. multi-robot autonomy.

To highlight the different aspects of SLAM datasets and benchmarks, we provide a detailed taxonomy of studied datasets in Figure 2.6

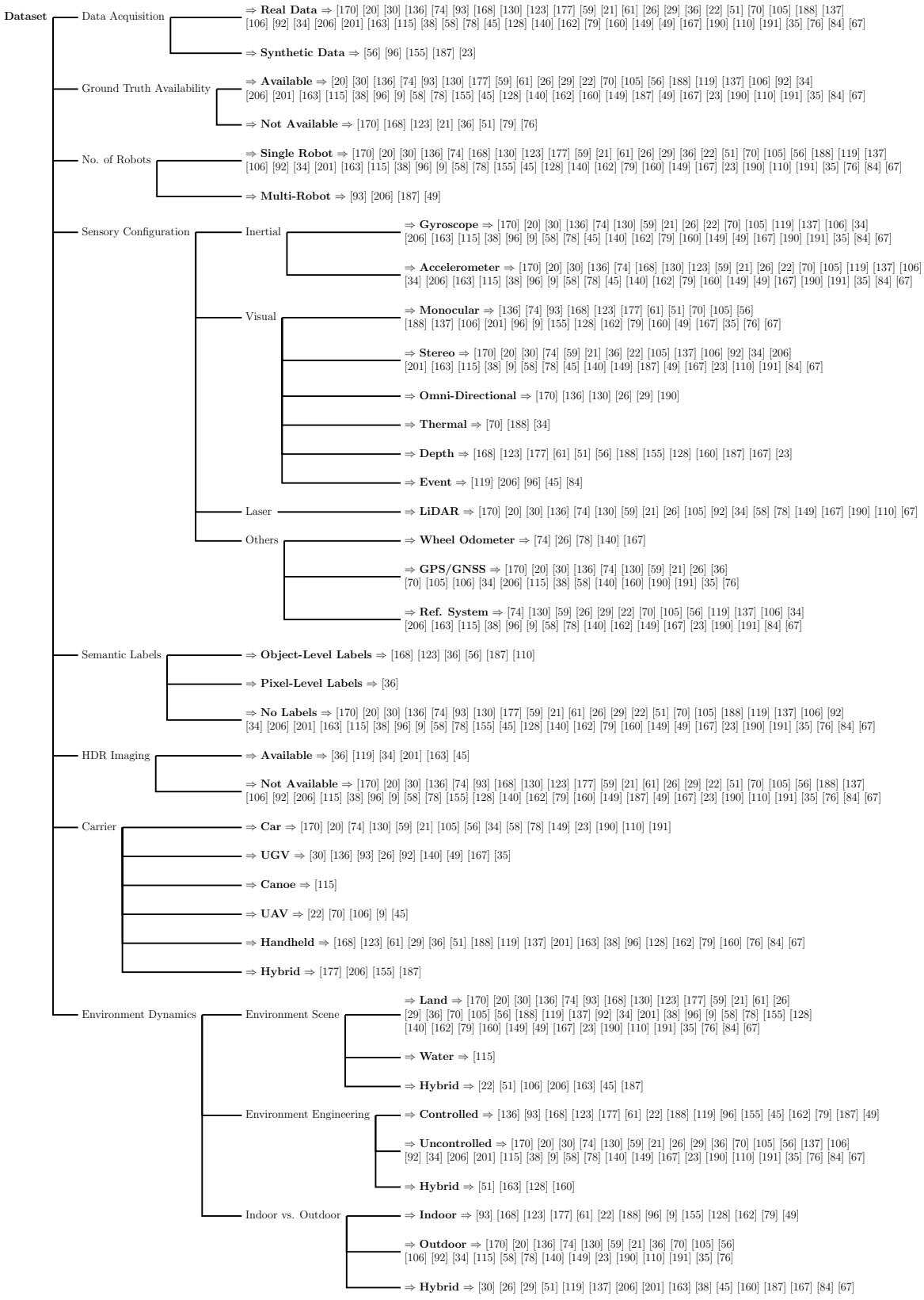


Figure 2.6: A Detailed Taxonomy of SLAM Datasets/Benchmarks Based on Qualitative Properties

2.2.1.2 A Qualitative Comparison of SLAM Datasets

In order to provide an overview of the SLAM dataset landscape, we qualitatively compare an extended number of SLAM datasets and benchmarks which are widely used in the community. We start by comparing the sensory configurations in Table 2.3. Then, we compare diversity aspects of different datasets such as the dataset structure, carrier used, and environment exploited in Table 2.4. Finally, we provide a comparison of the popularity of those studied datasets by comparing the number of citations for each one of them in Table 2.5.

2.2.2 Characterization of SLAM datasets

Quantitative comparison of datasets is typically conducted in the context of introducing a new SLAM algorithm/system with the purpose of justifying the evaluation methodology selected, or introducing a new SLAM dataset to illustrate and elaborate on the differences between the new proposed dataset, and the previously available ones [155]. In the former context, datasets are qualitatively compared with respect to publicly available information. While in the latter context, they are usually compared based on a single metric to provide evidence of how the new dataset is superior compared to its peers. Lately, the focus has started to direct towards SLAM datasets as a discipline of study. For instance, in [101], an extended number of SLAM datasets were reviewed and compared based on a number of qualitative metrics. The paper provided a good overview of the landscape and paved the way to many complementary studies.

On the other hand, quantitative comparison of datasets can provide some interesting and useful metrics for dataset characterization. However, the generalization of these metrics and the aggregation of them under a single framework for dataset characterization have rarely been attempted, although studies exist on narrow aspects of dataset characteristics. For instance, in [155], the difficulty of a sequence of robot sensor measurements has been measured independently from the execution of any SLAM system using the Wasserstein

Table 2.3: Qualitative Comparison of Sensory Configurations of Publicly Available SLAM Datasets

Dataset	Release Date	Data Type	Ground Truth	Multi-Robot	Sensors Setup												Freq.(Hz)	
					Gyroscope	Accelerometer	Wheel Encoder	GPS/GNSS	Mono Cam	Stereo Cam	Omni Cam	Thermal Cam	Depth Cam	Event Cam	LiDAR	Ref. System	Visual	Inertial
New College [170]	2009	Real	x	x	✓	✓	-	✓	-	✓	✓	-	-	✓	-	-	20	28
Malaga 2009 [20]	2009	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	✓	-	-	7.5	100
Rasweeds [30]	2009	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	✓	-	-	15	100
Marulan [136]	2010	Real	✓	x	✓	✓	-	✓	✓	-	✓	-	-	✓	-	-	10-15	50
DARPA Urban [74]	2010	Real	✓	x	✓	✓	✓	✓	✓	✓	-	-	-	✓	-	✓	10	100
UTIAS MRCLAM [93]	2011	Real	✓	✓	-	-	-	-	✓	-	-	-	-	-	-	-	10	x
NYU Depth V1 [168]	2011	Real	x	x	-	✓	-	-	✓	-	-	-	✓	-	-	-	30	30
Ford Campus [130]	2011	Real	✓	x	✓	✓	-	✓	-	-	✓	-	-	✓	-	✓	15	100
NYU Depth V2 [123]	2012	Real	x	x	-	✓	-	-	✓	-	-	-	✓	-	-	-	30	30
TUM RGB-D SLAM [177]	2012	Real	✓	x	-	-	-	-	✓	-	-	-	✓	-	-	-	30	-
KITTI Odometry [59]	2012	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	✓	-	✓	10	100
Malaga 2013 [21]	2013	Real	x	x	✓	✓	-	✓	-	✓	-	-	-	✓	-	-	20	100
MS 7-Scenes [61]	2013	Real	✓	x	-	-	-	-	✓	-	-	-	✓	-	-	-	30	x
UMich NCLT [26]	2015	Real	✓	x	✓	✓	✓	✓	-	-	✓	-	-	✓	-	✓	5	100
TUM Omni LS-SLAM [29]	2015	Real	✓	x	-	-	-	-	-	-	✓	-	-	-	-	✓	30	-
Cityscapes [36]	2016	Real	x	x	-	-	-	✓	-	✓	-	-	-	-	-	-	17	x
EuRoC MAV [22]	2016	Real	✓	x	✓	✓	-	-	-	✓	-	-	-	-	-	✓	20	200
TUM Monocular VO [51]	2016	Real	x	x	-	-	-	-	✓	-	-	-	✓	-	-	-	20-50	x
Hetero.UAV Fleet [70]	2016	Real	✓	x	✓	✓	-	✓	✓	-	-	✓	-	-	-	✓	30	50
Oxford RobotCar [105]	2016	Real	✓	x	✓	✓	-	✓	✓	✓	-	-	-	✓	-	✓	16	50
Virtual KITTI [56]	2016	Synthetic	✓	x	-	-	-	-	✓	-	-	-	✓	-	-	✓	10	-
CoRBS [188]	2016	Real	✓	x	-	-	-	-	✓	-	-	✓	✓	-	-	-	30	-
Event Cam Dataset [119]	2017	Both	✓	x	✓	✓	-	-	-	-	-	-	-	-	✓	✓	24	1k
PennCOSYVIO [137]	2017	Real	✓	x	✓	✓	-	-	✓	✓	-	-	-	-	-	✓	20	200
Zurich Urban MAV [106]	2017	Real	✓	x	✓	✓	-	✓	✓	✓	-	-	-	-	-	✓	30	30
Chilean Mine [92]	2017	Real	✓	x	-	-	-	-	-	✓	-	-	-	✓	-	-	16	-
KAIST MS [34]	2018	Real	✓	x	✓	✓	-	✓	-	✓	-	✓	-	✓	-	✓	30	25
MVSEC [206]	2018	Real	✓	✓	✓	✓	-	✓	-	✓	-	-	-	-	✓	✓	20-50	200
SPVO [201]	2018	Real	✓	x	-	-	-	-	✓	✓	-	-	-	-	-	-	30	x
TUM Visual-Inertial [163]	2018	Real	✓	x	✓	✓	-	-	-	✓	-	-	-	-	-	✓	20	200
VI Canoe [115]	2018	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	-	-	✓	20	200
ADVIO [38]	2018	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	-	-	✓	60	100
InteriorNet [96]	2018	Synthetic	✓	x	✓	✓	-	-	✓	-	-	-	-	-	✓	✓	25	800
Blackbird [9]	2018	Both	✓	x	✓	✓	-	-	✓	✓	-	-	-	-	-	✓	120	100
Urban@CRAS [58]	2018	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	✓	-	✓	15	200
Complex Urban [78]	2019	Real	✓	x	✓	✓	✓	-	-	✓	-	-	-	✓	-	✓	10	200
ICL [155]	2019	Synthetic	✓	x	-	-	-	-	✓	-	-	-	✓	-	-	-	20-30	x
UZH-FPV [45]	2019	Real	✓	x	✓	✓	-	-	-	✓	-	-	-	-	✓	-	30-50	0.5-1k
ReFusion [128]	2019	Real	✓	x	-	-	-	-	✓	-	-	-	✓	-	-	-	30	-
Rosario [140]	2019	Real	✓	x	✓	✓	✓	✓	-	✓	-	-	-	-	-	✓	15	142
TUM RS-VI [162]	2019	Real	✓	x	✓	✓	-	-	✓	-	-	-	-	-	-	✓	20	200
ZJU-SenseTime [79]	2019	Real	x	x	✓	✓	-	-	✓	-	-	-	-	-	-	-	30	100-400
ETH3D [160]	2019	Real	✓	x	✓	✓	-	✓	✓	-	-	-	✓	-	-	-	27	-
Newer College [149]	2020	Real	✓	x	✓	✓	-	-	-	✓	-	-	-	✓	-	✓	30	650
TartanAir [187]	2020	Synthetic	✓	✓	-	-	-	-	-	✓	-	-	✓	-	-	-	-	x
AirMuseum [49]	2020	Real	✓	✓	✓	✓	-	-	✓	✓	-	-	-	-	-	-	20	400k
OpenLORIS [167]	2020	Real	✓	x	✓	✓	✓	-	✓	✓	-	-	✓	✓	-	✓	30	60-400
Virtual KITTI 2 [23]	2020	Synthetic	✓	x	-	-	-	-	-	✓	-	-	✓	-	-	✓	10	-
UrbanLoco [190]	2020	Real	✓	x	✓	✓	-	✓	-	-	✓	-	-	✓	-	✓	10	100
YCOR [110]	2021	Real	✓	x	-	-	-	-	-	✓	-	-	-	✓	-	-	10	x
4Seasons [191]	2021	Real	✓	x	✓	✓	-	✓	-	✓	-	-	-	-	-	✓	30	2k
ROOAD [35]	2021	Real	✓	x	✓	✓	-	✓	✓	-	-	-	-	-	-	-	30	400
CrowdDriven [76]	2021	Real	x	x	-	-	-	✓	✓	-	-	-	-	-	-	-	-	-
TUM VIE [84]	2021	Real	✓	x	✓	✓	-	-	-	✓	-	-	-	-	✓	✓	20	200
HILTI SLAM Challenge [67]	2021	Real	✓	x	✓	✓	-	-	✓	✓	-	-	-	✓	-	✓	10	800

Table 2.4: Qualitative Comparison of SLAM datasets Diversity and Capabilities

Dataset	Distance (km)	#Frames	#Scenes	#Seq's	Labels	HDR	Carrier	Environment		
								Scene	Ctrl'd	In/Out
New College [170]	2.2	51,000	-	9	-	-	Car	Land	x	Outdoor
Malaga 2009 [20]	6	38,000	2	6	-	-	Car	Land	x	Outdoor
Rasweeds [30]	1.9	-	-	11	-	-	UGV	Land	x	Both
Marulan [136]	-	-	40	40	-	-	UGV	Land	✓	Outdoor
DARPA Urban [74]	92.7	-	3	3	-	-	Car	Land	x	Outdoor
UTIAS MRCLAM [93]	-	-	1	9	-	-	UGV	Land	✓	Indoor
NYU Depth V1 [168]	-	108,617	7	64	✓	-	Handheld	Land	✓	Indoor
Ford Campus [130]	5.1	7,000	2	2	-	-	Car	Land	x	Outdoor
NYU Depth V2 [123]	-	435,103	26	464	✓	-	Handheld	Land	✓	Indoor
TUM RGB-D SLAM [177]	0.4	65,000	27	27	-	-	Hybrid	Land	✓	Indoor
KITTI Odometry [59]	39.2	41,000	22	22	-	-	Car	Land	x	Outdoor
Malaga 2013 [21]	36.8	113,082	15	15	-	-	Car	Land	x	Outdoor
MS 7-Scenes [61]	-	42,660	7	50	-	-	Handheld	Land	✓	Indoor
UMich NCLT [26]	147.4	-	2	27	-	-	UGV	Land	x	Both
TUM Omni LS-SLAM [29]	-	33634	5	5	-	-	Handheld	Land	x	Both
Cityscapes [36]	-	25,000	8	50	✓	✓	Handheld	Land	x	Outdoor
EuRoC MAV [22]	0.8936	27,049	2	11	-	-	UAV	Hybrid	✓	Indoor
TUM Monocular VO [51]	-	190,573	50	50	-	-	Handheld	Hybrid	✓	Both
Hetero.UAV Fleet [70]	-	-	-	17	-	-	UAV	Land	x	Outdoor
Oxford RobotCar [105]	1010	11,070,651	-	133	-	-	Car	Land	x	Outdoor
Virtual KITTI [56]	-	17000	5	35	✓	-	Car	Land	x	Outdoor
CoRBS [188]	0.407	-	20	4	-	-	Handheld	Land	✓	Indoor
Event Cam Dataset [119]	-	N/A	11	27	-	✓	Handheld	Land	✓	Both
PennCOSYVIO [137]	0.6	-	2	1	-	-	Handheld	Land	x	Both
Zurich Urban MAV [106]	2	-	-	-	-	-	UAV	Hybrid	x	Outdoor
Chilean Mine [92]	2	-	7	58	-	-	UGV	Land	x	Outdoor
KAIST MS [34]	42	105,000	4	27	-	✓	Car	Land	x	Outdoor
MVSEC [206]	-	-	5	11	-	-	Hybrid	Hybrid	x	Both
SPVO [201]	-	-	-	11	-	✓	Handheld	Land	x	Both
TUM Visual-Inertial [163]	20	-	5	28	-	✓	Handheld	Hybrid	✓	Both
VI Canoe [115]	2.7	53,275	1	10	-	-	Canoe	Water	x	Outdoor
ADVIO [38]	4.5	-	5	23	-	-	Handheld	Land	x	Both
InteriorNet [96]	-	15000	15	15	-	-	Handheld	Land	✓	Indoor
Blackbird [9]	-	-	5	186	-	-	UAV	Land	x	Indoor
Urban@CRAS [58]	14.2	-	5	5	-	-	Car	Land	x	Outdoor
Complex Urban [78]	190.989	-	12	18	-	-	Car	Land	x	Outdoor
ICL [155]	0.5848	-	4	8	-	-	Hybrid	Land	✓	Indoor
UZH-FPV [45]	10	-	4	27	-	✓	UAV	Hybrid	✓	Both
ReFusion [128]	-	-	8	24	-	-	Handheld	Land	✓	Indoor
Rosario [140]	2.296	-	6	6	-	-	UGV	Land	x	Outdoor
TUM RS-VI [162]	0.446	-	1	10	-	-	Handheld	Land	✓	Indoor
ZJU-SenseTime [79]	-	-	9	16	-	-	Handheld	Land	✓	Indoor
ETH3D [160]	-	100170	19	98	-	-	Handheld	Land	✓	Both
Newer College [149]	2.2	-	2	7	-	-	Car	Land	x	Outdoor
TartanAir [187]	-	1,000,000	30	1037	✓	-	Hybrid	Hybrid	✓	Both
AirMuseum [49]	-	12,232	1	5	-	-	UGV	Land	✓	Indoor
OpenLORIS [167]	-	-	5	22	-	-	UGV	Land	x	Both
Virtual KITTI 2 [23]	-	17000	5	35	-	-	Car	Land	x	Outdoor
UrbanLoco [190]	13.8	-	12	12	-	-	Car	Land	x	Outdoor
YCOR [110]	-	1,076	4	4	✓	-	Car	Land	x	Outdoor
4Seasons [191]	-	-	9	33	-	-	Car	Land	x	Outdoor
ROOAD [35]	0.835	20566	6	6	-	-	UGV	Land	x	Outdoor
CrowdDriven [76]	-	2847	26	40	-	-	Handheld	Land	x	Outdoor
TUM VIE [84]	-	-	-	21	-	-	Handheld	Land	x	Both
HILTI SLAM Challenge [67]	-	-	8	12	-	-	Handheld	Land	x	Both

Table 2.5: Online availability of SLAM datasets and benchmarks and their popularity measured by number of citations

Dataset	Year	Citations	Link
New College [170]	2009	331	N/A
Malaga 2009 [20]	2009	191	https://www.mrpt.org/malaga_dataset_2009
Rasweeds [30]	2009	161	http://www.rawseeds.org/home/
Marulan [136]	2010	82	http://sdi.acfr.usyd.edu.au/
DARPA Urban [74]	2010	61	N/A
UTIAS MRCLAM [93]	2011	66	http://asrl.utias.utoronto.ca/datasets/mrclam/
NYU Depth V1 [168]	2011	546	https://cs.nyu.edu/ silberman/datasets/nyu_depth_v1.html
Ford Campus [130]	2011	287	http://robots.engin.umich.edu/SoftwareData/Ford
NYU Depth V2 [123]	2012	3690	https://cs.nyu.edu/ silberman/datasets/nyu_depth_v2.html
TUM RGB-D SLAM [177]	2012	2475	https://vision.in.tum.de/data/datasets/rgbd-dataset
KITTI Odometry [59]	2012	7735	http://www.cvlibs.net/datasets/kitti/eval_odometry.php
Malaga 2013 [21]	2013	206	https://www.mrpt.org/MalagaUrbanDataset
MS 7-Scenes [61]	2013	126	https://www.microsoft.com/en-us/research/project/rgb-d-dataset-7-scenes/
UMich NCLT [26]	2015	209	http://robots.engin.umich.edu/nclt/
TUM Omni LS-SLAM [29]	2015	238	https://vision.in.tum.de/data/datasets/omni-lsdslam
Cityscapes [36]	2016	5934	https://www.cityscapes-dataset.com/
EuRoC MAV [22]	2016	863	https://projects.asl.ethz.ch/datasets/doku.phpN/Aid=kmavvisualinertialdatasets
TUM Monocular VO [51]	2016	153	https://vision.in.tum.de/data/datasets/mono-dataset
Hetero.UAV Fleet [70]	2016	20	https://projects.asl.ethz.ch/datasets/doku.phpN/Aid=iser2016
Oxford RobotCar [105]	2016	809	https://robotcar-dataset.robots.ox.ac.uk/
Virtual KITTI [56]	2016	775	N/A
CoRBS [188]	2016	88	http://corbs.dfki.uni-kl.de/
Event Cam Dataset [119]	2017	284	http://rpg.ifi.uzh.ch/davis_data.html
PennCOSYVIO [137]	2017	68	https://daniilidis-group.github.io/penncosyvio/
Zurich Urban MAV [106]	2017	84	http://rpg.ifi.uzh.ch/zurichmavdataset.html
Chilean Mine [92]	2017	19	N/A
KAIST MS [34]	2018	97	https://sites.google.com/view/multispectral/home
MVSEC [206]	2018	153	https://daniilidis-group.github.io/mvsec/
SPVO [201]	2018	6	https://www.hs-karlsruhe.de/odometry-data/
TUM Visual-Inertial [163]	2018	123	https://vision.in.tum.de/data/datasets/visual-inertial-dataset
VI Canoe [115]	2018	16	https://databank.illinois.edu/datasets/IDB-9342111
ADVIO [38]	2018	40	https://github.com/AaltoVision/ADVIO
InteriorNet [96]	2018	119	https://interiornet.org/
Blackbird [9]	2018	33	N/A
Urban@CRAS [58]	2018	19	https://rdm.inesctec.pt/dataset/nis-2018-001
Complex Urban [78]	2019	78	https://irap.kaist.ac.kr/dataset/
ICL [155]	2019	9	https://peringlab.org/lmdata/
UZH-FPV [45]	2019	82	https://fpv.ifi.uzh.ch/
ReFusion [128]	2019	28	N/A
Rosario [140]	2019	22	https://www.cifasis-conicet.gov.ar/robot/doku.php
TUM RS-VI [162]	2019	17	https://vision.in.tum.de/data/datasets/rolling-shutter-dataset
ZJU-SenseTime [79]	2019	22	http://www.zjucvg.net/eval-vislam/
ETH3D [160]	2019	76	https://www.eth3d.net/slam_datasets
Newer College [149]	2020	18	https://ori-drs.github.io/newer-college-dataset/
TartanAir [187]	2020	31	https://theairlab.org/tartanair-dataset/
AirMuseum [49]	2020	1	http://ieee-dataport.org/2683
OpenLORIS [167]	2020	38	https://lifelong-robotic-vision.github.io/dataset/scene.html
Virtual KITTI 2 [23]	2020	45	https://europe.naverlabs.com/research/computer-vision/proxy-virtual-worlds-vkitti-2/
UrbanLoco [190]	2020	27	https://advdataset2019.wixsite.com/urbanloco
YCOR [110]	2021	44	https://theairlab.org/yamaha-offroad-dataset/
4Seasons [191]	2021	7	https://www.4seasons-dataset.com/
ROOAD [35]	2021	0	https://github.com/unmannedlab/ROOAD
CrowdDriven [76]	2021	0	N/A
TUM VIE [84]	2021	0	https://vision.in.tum.de/data/datasets/visual-inertial-event-dataset
HILTI SLAM Challenge [67]	2021	0	N/A

distance, which is a statistical method to measure the distance between random variables. By treating each pose as a random variable, the metric can be computed, and thus, is used to measure the level of difficulty of the given sequence. Additionally, the aggressiveness of the robot motion was measured in [45] using the magnitude of optical flow. Finally, the motion composition and its variability among system axes were measured using principal component analysis (PCA) in [187] in the analysis of the new challenging TartanAir dataset. A study of the relation between the dataset properties and the SLAM performance was presented in [198]. In this study, qualitative characteristics of datasets were used as categorical features to build a decision tree to characterize the difficulty of a dataset. Then, the study was extended to explore the relationship between the SLAM performance and the categorical properties. These efforts were directed towards the introduction of characterization metrics, rather than a framework for dataset characterization.

2.2.3 Dataset analysis and coverage

Analysis of dataset properties is a crucial topic in a wide range of disciplines in science and engineering especially in learning problems [107] and in pure data analysis problems [11]. The analysis of dataset bias [183] and dataset shift [185] in the aforementioned disciplines led to methods and techniques for correcting the dataset bias [82] particularly in the context of deep neural network models [88]. Although the aforementioned research is directed to other computer vision tasks, the same concepts can be adopted in SLAM research especially in learning-based SLAM [104]. The concept of coverage pattern was introduced in [11], which uses characterization parameters as a feature vector. Moreover, the relationship between different patterns was modelled by edges in a directed graph. The approach proposed is suitable for situations where the characterization metrics are discrete and not continuous. In SLAM, however, the characterization of measurements is often a continuous variable. Thus, to utilize the same concepts mentioned in [11], quantization in the continuous space is required, leading to quantization errors and potential sub-optimal analysis of the coverage.

2.2.4 Fault detection in SLAM

Fault detection in SLAM is crucial to achieve robustness and resilience due to the fact that both rely on the ability of the system to self assess its performance and take any corrective actions required. However, this topic was not the core focus of the research community. Recently, a modest attention is given to this direction in SLAM and in robotics in general. In this section, we review the effort exerted in that field and discuss the relevance of the work done to robustness and resilience of SLAM in detail. Additionally, we provide a taxonomy of the different methods mentioned to provide a graphical representation of the available landscape in that topic.

There exist a wide spectrum of methods for fault detection associated with the specifics of the SLAM paradigm, internal structure, sensors used, observability, and evaluation environment. The following subsections discuss each category of fault detection and briefly discuss its principle of operation. As known, modern graph-based SLAM pipelines consist of three major submodules: Sensors and sensor acquisition/pre-processing, front-end where feature extraction and data association is conducted, and back-end where MAP estimation is conducted.

Prior to graph-based SLAM, different filtering techniques were used to solve the SLAM problem. Filtering-based SLAM depends the utilization of either kalman filters and their variants or Particle filters for the estimation of the SLAM state [24]. Fault detection in Filtering-based method was relatively straight forward due to the nature of the filters themselves that has an internal indicator for state quality. For instance, in kalman filters, monitoring of state covariance was used in [73] to detect faults in the system as it provides an indication of the quality of the estimates. On the other hand, anomalies in the weights of different particles in particle filters can be used as an indicator for estimation faults [47].

2.2.4.1 Fault detection in sensory inputs

SLAM relies on sensory inputs to percept the environment, and among the most famous sensors used are: IMUs, RGB cameras, and LiDARs. It is worth mentioning that, faults in SLAM due to sensors can be divided into two categories: faults in the sensor hardware itself resulting in input signals that are not properly representing the environment, and faults due to the nature of the environment and the motion profiles of the carrier.

The former category is usually detected by means of *fault detection and isolation (FDI)* techniques [180], where nominal sensor conditions are monitored and a fault sensor is isolated from any estimation process. A wide spectrum of faults can happen in sensors due to manufacturing defects and variability, operation outside nominal conditions, and bad calibration. A comprehensive review of such faults are provided in [18] for IMUs, [164] for RGB camera, and in [31] for LiDARs.

On the other hand, the latter category relies on the detection of out-of-distribution measurements, consistency among sensor readings, and statistical modeling of sensor errors. This is under the assumption that the sensor itself is not faulty and that, sensor readings are a reflection of the environment it was deployed in.

Statistical modeling of IMU biases has been discussed in the context of filtering-based SLAM, where the EKF state is defined to include both gyroscope biases and accelerometer biases [197]. This statistical modeling seeks to predict the non-deterministic bias instability in both gyroscopes and accelerometers along side the covariance of such estimations. This covariance is usually used as an indication of the quality of the input, and thus, an in-direct indication of a potential fault in subsequent modules. Additionally, saturation conditions in IMUs (specifically accelerometers) were used to detect failures in aggressive manoeuvres of drones in [53], while detection of zero-rate states was used for detection of stationary carriers which was used an initiator for error correction in navigation state in [7].

Additionally, the monitoring of image quality and consistency over a tra-

jectory is one key to detect failure in SLAM systems due to the fact that such conditions can constitute a big challenge for SLAM to overcome. This can be done by characterization of RGB images and studying the relation between such characteristics and the overall performance of SLAM, which was introduced and discussed in our prior work [5].

Moreover, LiDARs play a vital role in modern perception systems, and range measurements can be impacted by adverse conditions in the environment such as fog, rain, among many others. These conditions result in anomalies in the point clouds retrieved and can indicate a challenge on the SLAM system, which is also an indirect indication of failure. Detection of such anomalies was provided in [202] where point clouds were analyzed and a metric for measuring the quality of point clouds retrieved from LiDARs. Studying the relation between such metric and SLAM performance can provide an early indicator of global SLAM failure.

Finally, consensus of sensor readings was also considered as a method for detecting reliable sensors to use in estimation. For instance, in [80], a number of parallel EKFs were used with different mixes of sensory inputs. Then, a majority vote-like decision process was conducted to detect and isolate problematic sensory inputs.

2.2.4.2 Fault detection in SLAM front-end

SLAM front-end is responsible for both short and long term data association. For that to happen, a wide spectrum of tasks are performed such as feature extraction, feature matching, odometry, and registration. Faults can happen in any or all of these modules. Thus, there exist many categories of fault detection methods that differ in terms of their indication of either local or global fault, their dependency on a ground-truth for operation, and their knowledge of the structure of the SLAM algorithm architecture. In this section we discuss these techniques and provide a synthesis of their usage in SLAM pipelines.

Local fault detection is the detection of faults in a specific SLAM component. This may not necessitate the failure of the whole SLAM system as correction can happen in later modules. Such techniques can be classified

based on their reliance on ground truth data or the local module/component in SLAM they test.

The need for ground truth data was evident in a number of studies. For instance, the amount of clutter in the scene (scene complexity) was modeled using an α parameter in [122], which is then used to estimate the uncertainty in pose estimation. Weak-supervision used inertial data in [3] to train a predictor that can predict failures in odometry modules. Labels were generated using radar readings, and was assumed to be inaccurate in the learning process. Ground truth data was also used in [6], where an model was trained to predict failures in LiDAR SLAM. The method crafts a descriptor of the input data (laser scans) and pass this through two a supervised model to predict pass/fail conditions. Point cloud alignment score was also used to predict SLAM failures by training a classifier with score-pass/fail data in [199].

However, other methods and techniques were able to detect faults without the reliance on ground truth data. For instance, detection of failure in features tracking was presented in ORB-SLAM3 [25] where the number of tracked points is monitored, and a failure is declared when this number falls below a threshold of 15 points. Key points covariance was predicted in [112] and was utilized to judge keypoints and guide their association process in object-based SLAM. Moreover, uncertainty in visual odometry estimates was introduced in [2], where it was then used to verify loop closure estimates. The system keeps track of all loop closure opportunities and perform late verification on them to avoid rejecting good loop closure candidates. RTAB-Map [91] also relied on odometry failures as an indicator of SLAM faults, where it detects constant odometry estimates that are equal to identity transforms to initiate its sub-mapping module. Additionally, point cloud alignment and scan matching scores were used to predict failures in a self-supervised setup in [1] where the concept of entropy was used to produce a quality metric that is highly sensitive to the performance of LiDAR odometry. Scan matching was utilized as a fault detection technique in [77], results of scan matching using ICP are converted to images and then feed to a CNN to detect matching errors. Furthermore, residuals monitoring was also utilized to estimate faults in vision

sub-module in GPS/SLAM setup in [17] by using the concept of super-pixel. The detect faults were used to reject bad estimates coming from vision sub-module. Finally, a model of the error residual is learned on-line in [148]. This error is used to determine the important portions of the image, where more features are likely to be reliable. The predicted residuals are then fed-back to the feature extraction module to extract more features from important and interesting parts of the image.

On the other hand, global fault detection is concerned with the detection of situations where SLAM will either not produces estimates or provide unreliable estimates that cannot be consumed by subsequent control algorithms. Most of the global methods for global fault detection require ground-truth data. For instance, GT is required to properly model SLAM errors as discussed in [204] where a model of the vision system is learned in order to predict failures. Additionally, prediction of SLAM performance cannot be achieved without GT data as proposed in [103] where a regression model is trained using an extended number of simulations to predict the performance of SLAM by modeling the travelled path traversed by the robot by Voronoi Graphs. Although the system is relying on ground-truth data to perform, this GT data is acquired from simulations, which lowers the level of effort required to provide it. In our work [4], we rely on ground truth to train a random forest given the characterization of the input to predict the ATE of the trajectory travelled so far. Moreover, a metric was defined to classify pass/fail conditions of SLAM estimates using a supervised learning setup in [40], where features were extracted from sensor inputs using a neural network. In [89], another model was trained to detect pass/fail conditions as well using sensor inputs where the model embraces early detection of failures before they happen given a sequence of events. Finally, to determine map consistency, a new metric was developed in [118] that requires ground truth maps. However, the system cannot run on-line in real-time which is an obstacle in the way of using it for real-world applications.

Finally, model-agnostic fault detection systems can be obtained by means of simulation-based modeling of errors given only the input characteristics.

The independence of the model in this context means independence from evaluation setup, and from an input from the SLAM system. An example of this system is the aforementioned predictor of SLAM performance [103].

2.2.4.3 Built-in fault detection mechanism in SLAM

A wide range of SLAM systems do not have an internal mechanism for detecting faults due to their reliance on loop closure for the corrections of errors if they happen. However, this assumption may not hold true in many real-world scenarios where loop closure opportunities may be very scarce or when they are not reliable enough to be included in the SLAM optimization problem.

For that reason, a limited number of SLAM systems introduced an extra module to the conventional SLAM pipeline, which is responsible for the detection of faults, and the triggering of any recovery mechanisms if available. For instance, ORB-SLAM3 [25] monitors the number of matches between two keyframes, when this number falls below a defined threshold, the system is declared to be visually lost. To recover, the system enters an inertial-only mode where only measurements from the IMU are used for localization till the vision pipeline recovers. Additionally, VINS-Mono [146] relies on multiple fault indicators which are the number of tracked features across keyframes, sudden jump in the estimated pose (translation or rotation), and sudden change in IMU bias values optimized by the backend. In RTAB-Map [91], the system relies on the detection of identity transformation from odometry sources for a defined time window. When this behaviour is detected, a failure is declared and recovery mechanism is engaged using sub-mapping.

2.2.4.4 Fault detection in multi-robot systems

Fault detection in multi-agent/robot system can be achieved by collective crowd-sending in the sense that, different agents/robots share their perception information and enforce different constraints to maintain both local and global consistency. The existence of more sensing and constraints sources makes it easier for rejecting outlier that this where only one robot is used for sensing. For instance, this concept was utilized in [100], where multiple robots were used

to provide globally consistent maps through sharing of experience. Then, the globally consistent information where used to calculate the local reprojection error residuals, which guides the process of feature matching. Consequently, they were able to maintain local consistency through global consistency.

2.2.5 Fault tolerance and recovery in SLAM

Fault tolerance and recovery play a critical role to achieve resilience in SLAM due to the fact that they equip SLAM algorithms with the ability to re-converge after a divergence is witnessed due to the in ability to produce estimates or when estimates are not reliable enough for subsequent tasks. In this section, we discuss how fault recovery was introduced in the SLAM pipeline.

2.2.5.1 Fault tolerance in SLAM back-end

Most of the mentioned approaches so far either deal with the sensory input or the SLAM front end. However, others directed the attention to the SLAM backend and proposed the usage of certifiable algorithms [196] where the back-end (MAP optimization) is modified to provide robustness against huge amount of outlier constraints (70% to 90% of random outlier constraints) in the pose graph.

2.2.5.2 Fault recovery after detection

A limited number of general purpose SLAM algorithms are equipped with modules to monitor the existence of a fault and thus provides a mechanism to overcome such faults. In the literature, two paradigms in fault recovery were witnessed. The first paradigm relies on the existence of multiple odometry sources that differ in their failure modes. For instance, visual-inertial SLAM is a typical example of a multi-sensor SLAM system where inertial odometry and visual odometry failure modes do not intersect. In that case, when a fault is detected in the visual pipeline, it is suspended from producing estimates, and the inertial odometry pipeline overtakes the system until a recovery in the visual pipeline is observed. An example of such behaviour was introduced in LVI-SAM [165] and R^3-LIVE [97], where a tightly-coupled visual-inertial

architecture was utilized. Once a fault is detected by monitoring the number of matched points, the coupling is disengaged and the systems runs in inertial odometry-only mode.

The second paradigm relies on the fact the initialization phase of SLAM is more rigorous and stable compared to normal operation. For that reason, re-initialization of the system was conducted whenever a fault is detected. This is evident in VINS-Mono [146] where re-initialization is invoked whenever a fault is detected by monitoring the number of matched points, the consistency in the estimated trajectories, and the sudden changes in carrier direction or pose.

Modern SLAM solutions depends on a mix of the two paradigms to achieve higher reliability. Once a fault is detected, the visual pipeline is suspended and the inertial odometry is engaged as the sole estimates producer in the system. If the error is sustained over a long period of time, re-initialization is engaged as well to provide a new stable and reliable state of the system. In order to avoid the shortcomings of re-initialization, the concept of sub-mapping is utilized where an initialization of a new map is conducted and then those sub-maps are merged in parallel whenever a loop closure is detected. This technique was introduced in RTAB-Map [91] and ORB-SLAM3 [25] to provide a more rigorous fault recovery mechanism for SLAM.

2.2.5.3 Fault correction without detection

A wide spectrum of SLAM algorithms depend on the correction of errors without detection by solely relying on loop closure constraints to the factor graph and initiate a new optimization iteration. This is evident in google cartographer [69], ORB-SLAM3 [121], and S-PTAM [139] to name a few. This category of systems assumes correct detection of loop closure and the availability of loop closure opportunities in the trajectory traversed, which are strong assumptions specially for general-purpose SLAM algorithms.

Chapter 3

A Framework for Quantitative Characterization of Datasets

3.1 Introduction and Motivation

The last few decades have witnessed a number of advancements in the field of Simultaneous Localization and Mapping (SLAM). This has been manifested in the introduction of a number of SLAM solutions targeting accuracy and efficiency such as: RTAB-Map [91], ORB-SLAM 1,2, and 3 [120][121][25], and VINS-Mono [146], among many others. However, another important consideration of performance, robustness/resilience, has rarely been formally addressed or measured due to the lack of a rigorous definition for it in the SLAM literature. With the increasing need for reliable SLAM solutions in a wide range of critical applications, such as autonomous driving, search and rescue mission, social robotics etc., one may ask whether currently available datasets are able to properly test robustness and resilience of a SLAM system, and whether they can provide us with enough confidence in the system performance both in a challenging situation outside of its tested operating range, and for operating for an extended period of time.

As outlined previously, robustness and resilience are tightly coupled with the pertrubations subjected to a robotic system. Perturbations are defined to be any external conditions causing the system to deviate from its equilibrium state. In SLAM, perturbations are usually implicitly contained in benchmark

datasets with different sensor measurements and used as inputs to SLAM algorithms. Quantitative characterization of those perturbations leads to the identification of the operating ranges/conditions of SLAM systems and thus, provides a measurement for robustness and resilience of SLAM.

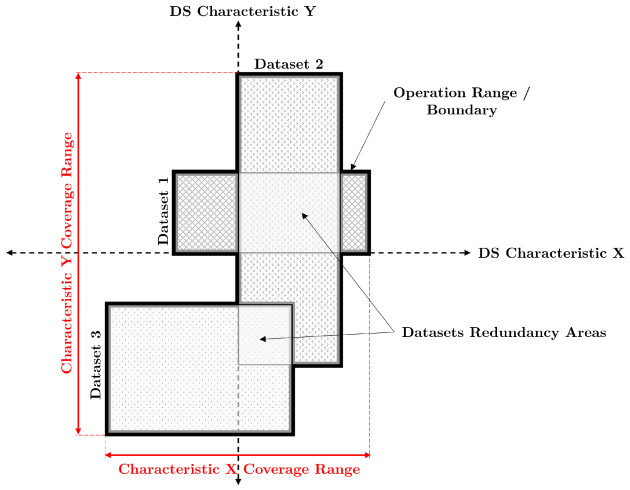


Figure 3.1: An illustration of how the characterization of datasets defines the operating range of SLAM

Subsequently, there is an undeniable need to systematically evaluate and compare SLAM datasets based on specified quantifiable metrics. A quantitative analysis of different characteristics present in datasets will not only help identify the suitability of a certain dataset to evaluate robustness or resilience, but also will help in standardizing the process of evaluation of SLAM systems against specifications in general. Figure 3.1 illustrates an example of how the characteristics of datasets define the operating ranges of a SLAM system and the limits in which the performance is evaluated and guaranteed.

In order to meet the above need, we introduce a generic and extendable framework for automatic characterization and analysis of SLAM datasets. The framework is designed to provide an efficient way for extension to additional sensors, characteristics, or datasets with limited development efforts. The current version supports visual-inertial SLAM datasets due to their popularity in the SLAM literature.

Developing a SLAM system typically focuses on two aspects: the design of the algorithm itself, and the methodology used for testing and evaluation. The former aspect has received a lot of attention from the SLAM community and led to the development of many advanced and complex systems with different characteristics, sensors support, and architectures. On the other hand, the evaluation of SLAM has focused on the quantification of the localization and mapping quality [55], and on the introduction of benchmarks and datasets to be used for off-line testing and evaluation. The characterization of the introduced benchmarks/datasets, and the comparison between them has been mostly done qualitatively [101]. In this work, we direct the attention of the SLAM community to the importance of the characterization of the datasets themselves rather than the algorithms with the purpose of defining the operating conditions of systems. We provide a measurement of robustness and an evaluation procedure for resilience.

3.1.1 Measurement of Robustness and Resilience

As outlines in Chapter 1, the definitions for robustness and resilience rely on the knowledge of the operating conditions in which a SLAM system will be deployed. Current practices report accuracy when a given SLAM system is run against a pre-recorded dataset. The reported performance cannot be used as an indication for robustness or resilience unless the operation conditions this dataset imposes are defined and measured. Thus, the proposed framework bridges the gap by provided a systematic way to define operating conditions (characterization metrics) as well as measuring them (applying them on datasets). Therefore, accuracy is now tied to measurable environment conditions, and performance can only guaranteed if the deployment environment is exhibiting the same characteristics as the evaluation and testing environment.

3.2 Problem Formulation

Let $\mathcal{D}$ be a set of sequences Q_i of a SLAM dataset where:

$$\mathcal{D} = \{Q_i,\ i = 1, ..., n\} \tag{3.1}$$

As such, a sequence Q_i is a vector of m_i measurements (e.g. images/inertial measurements), i.e. $m_i = |Q_i|$, where i corresponds to a sequence inside the dataset $\mathcal{D}$.

Let $\mathcal{F}$ be a set of c characterization metrics $f_k(.)$, where:

$$\mathcal{F} = \{f_k,\ k = 1, ..., c\} \tag{3.2}$$

Each characterization metric $f_k(.)$ is a function which response when applied is denoted by q_i. Thus, we characterize each sequence, and a corresponding characterization vector q_i is generated for each characterization metric $f_k(.)$ applied so that:

$$q_i = \{q_{i,1}, ..., q_{i,m_i}\} \tag{3.3}$$

$q_{i,j,k}$ is a feature representing the result of applying a characterization metric $f_k(.)$ on sample (e.g. image) I_j in sequence i, and is given by:

$$q_{i,j,k} = f_k(I_{i,j}) \tag{3.4}$$

For each characterization metric $f_k(.)$ in $\mathcal{F}$, we seek to obtain a characterization vector q_i for each sequence Q_i in $\mathcal{D}$.

3.3 Proposed Method

To address the aforementioned need, we introduce a generic and extendable framework for the automatic characterization and analysis of SLAM datasets. The framework is designed to offer an efficient way for extension to additional sensors, characteristics, or datasets with limited development efforts. The current version supports visual-inertial SLAM datasets, given their popularity in the SLAM literature.

The framework is divided into a number of sub-modules that either work on-line (for data characterization) or off-line (for data analysis and visualization). The system includes a number of built-in configuration files to control different tunable hyperparameters present in the system. In Figure 3.2, a block diagram of all sub-modules of the framework and their interactions are provided in detail. These components are detailed as follows:

1. **Dataset Adaptor** Datasets differ in terms of the way the data is organized, which requires unification in order for the system to operate seamlessly. Thus, adaptation of datasets to match a certain format is done off-line before the start of the characterization process. This block changes based on the selected dataset and generates a dataset description file which is used by the framework for dataset reading.

2. **Dataset Handler** This module consists of a database that holds the set of sequences Q_i, and their enclosed sensor data. The acquisition timestamps are saved alongside the data for synchronization purposes. Additionally, it includes a scoreboard to store the characterization results once calculated from processing elements. Utility functions are also available to facilitate the process of data exchange with other modules.

3. **Processing Elements Handler** This module consists of a number of sample/sequence-level processing engines $f_k(.)$, which perform the characterization on dataset data. The characterization results q_i are propagated back to the dataset handler for recording in the dataset scoreboard.

4. **Dataset Scoreboard** The dataset scoreboard is a 2-D vector of smaller scoreboards/databases, where each element represents the characterization results of a sequence q_i when run through one of the processing elements $f_k(.)$.

5. **UI Handler and System Controller** The main responsibility of this sub-module is to handle the communication with the user, and control the flow of actions and data among system modules and components.

6. **Help Function and System-Wide Utilities** A number of system utilities are available such as: statistical utilities, calculus utilities, and data conversion/manipulation functions. Additionally, a number of import/export functions are provided to facilitate the later phase of results visualization.

7. **Data Analysis and Post-Processing** This module is considered an off-line parallel sub-system, where the characterization results are analyzed and visualized after exporting.

3.3.1 Characterization Stages

The characterization process is conducted on three different levels: Sample-level, Sequence-level, and Dataset-level. The system starts by applying different processing elements on applicable dataset samples in each sequence. Then, statistical analysis of the extracted sample-level data is conducted. The objective of this stage is to provide insights on the similarities and the level of diversity of sequences within the same dataset. Finally, statistical analysis of the aggregated data of the whole dataset is done with the objective of comparing datasets and providing rigorous measurement of their implied operating conditions.

3.3.2 Characterization Assumptions

The framework uses the default values of the configuration parameters of any underlying engines such as feature extraction, disparity calculations, etc. Tunable parameters such as threshold values can be controlled from within the framework. Both default and tunable parameters used for the characterization process are kept exactly the same to ensure fairness of comparison.

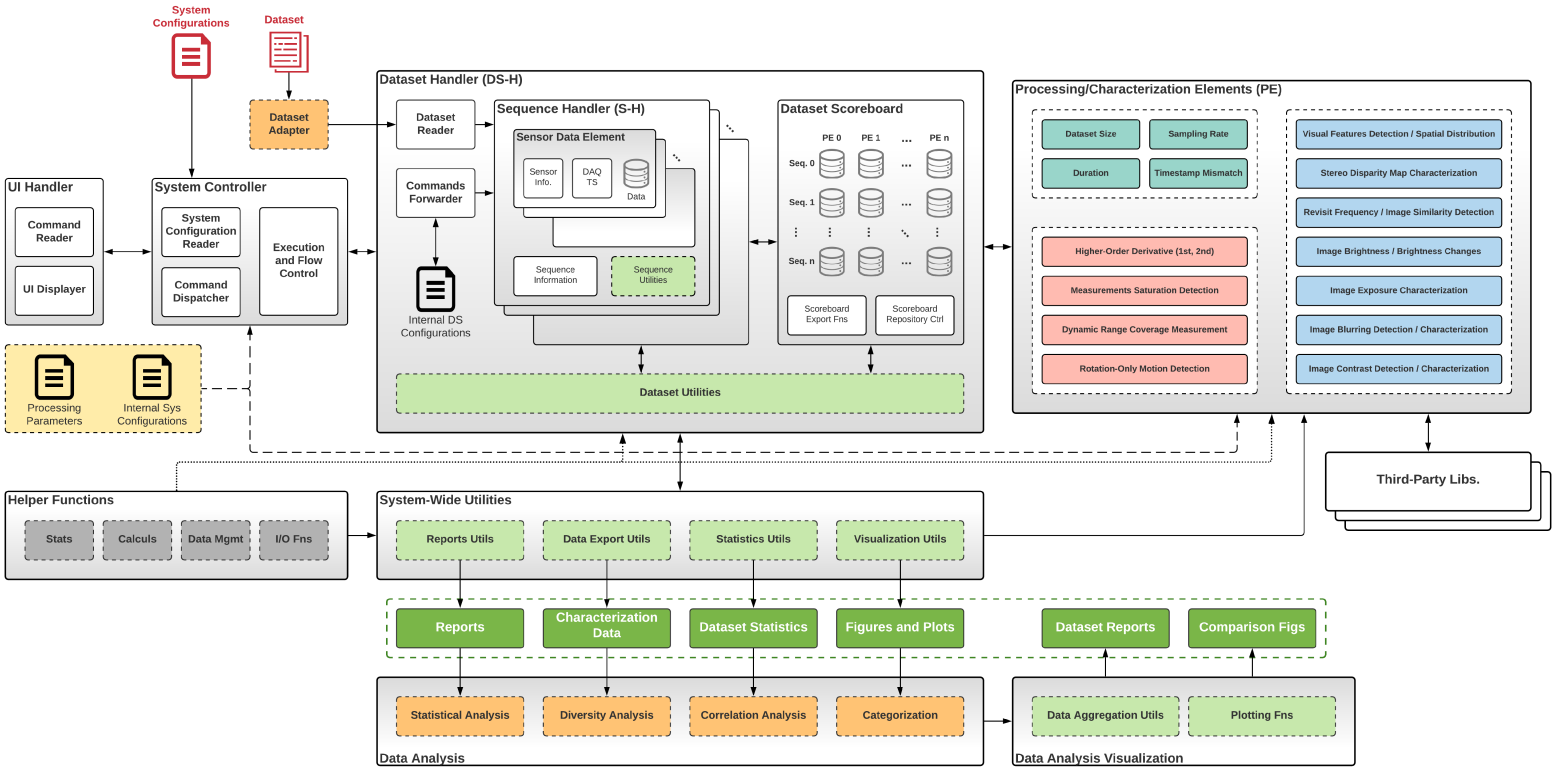


Figure 3.2: A detailed block diagram of the system, illustrating different modules, internal components and the flow of both commands and data throughout the framework

3.3.3 SLAM Dataset Characterization Metrics

A wide spectrum of characterization metrics can be selected for the purpose of determining the operating conditions of a SLAM dataset. Thus, we select a sub-set of metrics that are directly impacting SLAM performance and are of interest to the SLAM community. This section provides a description of a selection of characterization metrics which are both measurable and challenging to SLAM systems. However, the framework can be extended to more characterization parameters easily by defining new processing elements. The characterization parameters are categorized into three different groups, according to the aforementioned characterization levels of the framework. The different characterization parameters are summarized in Table 3.1.

3.3.3.1 General Characterization Metrics

General characterizations of datasets refer to the characteristics which do not depend on the physical output of a certain sensor. Rather, they depend on the conditions of data acquisition such as the size of the data, its total duration, changes in sampling rates, and the mismatch between timestamps of different sensors. These characteristics are usually reported as part of the qualitative description of SLAM datasets. However, not all datasets report them. Thus, the need to automatically report these quantities becomes important in our context.

A. Dataset Size

Dataset size refers to the number of sequences available $|Q|$ and the number of sensor measurements available $m_i = |Q_i|$ for each sequence. This can be extended to detect the number of scenes available as well, however, this semantic analysis is beyond the current scope of the paper.

B. Total Duration

The total duration is calculated by accumulating the timestamp difference of the sequence, which is equivalent to the difference between the end and start

Table 3.1: A Concise Summary of Datasets' Characterization Parameters

Parameter	Unit	SM-L	SQ-L	DS-L	Definition
General Parameters					
Measurements Size	Samples		✓	✓	Total number of measured samples / sensor
Total Duration	Sec.		✓	✓	$t_{total} = \sum_{i=2}^{N}(t_i - t_{i-1})$
Sampling Time	Sec.	✓	✓	✓	$t_{sensor} = (\sum_{i=2}^{N}(t_i - t_{i-1}))/N$
Timestamps Mismatch	Sec.	✓	✓	✓	Timestamps difference between corresponding sensors' samples
Higher-Order Derivatives [161]					
Jerk (j)	m/sec^3	✓	✓	✓	1^{st}-order time-derivative of acceleration data $[A_x, A_y, A_z]^T$.
Snap (S)	m/sec^4	✓	✓	✓	2^{nd}-order time-derivative of acceleration data $[A_x, A_y, A_z]^T$.
Angular acc. (α)	$^\circ/sec^2$	✓	✓	✓	1^{st}-order time-derivative of angular velocity data $[G_x, G_y, G_z]^T$.
Angular jerk (φ)	$^\circ/sec^4$	✓	✓	✓	2^{nd}-order time-derivative of angular velocity data $[G_x, G_y, G_z]^T$.
Sensor Saturation					
Dynamic Range Coverage	%		✓	✓	$(Max_{sensor} - Min_{sensor}/DR_{sensor})\%$
Dynamic Range Crossing [125]	%		✓	✓	$(\sum_{i=1}^{n} \mathbb{1}\lvert x_{(i,sensor)} - DR_{(sensor)}\rvert < DR_{(crossing-ratio)})$ %
Rotation-Only Motion					
Acceleration Magnitude [125]	m/sec^2	✓	✓	✓	$f_{mag} = \sqrt{f_{x,b}^2 + f_{y,b}^2 + f_{z,b}^2}$
Rotation-Only Samples	%	✓	✓	✓	$(\sum_{i}^{n} \mathbb{1}(f_{magnitude} \geq 9.81 \pm 10\%))$ %
Image Brightness					
Avg. Brightness ($\overline{Br}$) [16]	DL	✓	✓	✓	$\overline{Br} = \sum_{i}^{n}(0.299 * R + 0.587 * G + 0.114 * B)$
Zero-Mean Avg. Brightness Derivative (β_{Br})	DL		✓	✓	$\beta_{Br} = dBr/dt - \mu_{Br}$
Ratio of Thresholding (β_{Br})	%		✓	✓	$(\sum_{i}^{n} \mathbb{1}(\lvert\beta_{Br}\rvert < \sigma_T))\%, \quad \sigma_T \in \{\sigma_{Br}, 2\sigma_{Br}, 3\sigma_{Br}\}$
Image Exposure					
Trimmed Image Mean (μ_α) [150]	DL	✓	✓	✓	$\mu_\alpha = (1)/(n - 2\lceil n\alpha\rceil) * \sum_{i=n-\lceil n\alpha\rceil}^{\lceil n\alpha+1\rceil} I_i$
Trimmed Image Skewness (S_α) [150]	DL	✓	✓	✓	$S_\alpha = (\sum_{i=n-\lceil n\alpha\rceil}^{\lceil n\alpha+1\rceil}(I_i - \mu_\alpha)^2)/((n - 2\lceil n\alpha\rceil - 1) * \sigma_\alpha^3)$
Exposure Zone [60]	DL	✓	✓	✓	Detection of black, white, under-, over-, or properly exposed images
Image Contrast [134]					
Contrast Ratio (C_{CR})	DL	✓	✓	✓	$C_{CR} = (L_{target}/L_{background}) * 100$
Weber Contrast (C_W)	DL	✓	✓	✓	$C_W = ((L_{target} - L_{background})/L_{background}) * 100$
Michelson Contrast (C_M)	DL	✓	✓	✓	$C_M = (L_{max} - L_{min})/(L_{max} + L_{min})$
RMS Contrast (C_{RMS})	DL	✓	✓	✓	$C_{RMS} = \sqrt{1/n \sum_{i=1}^{n}(l_i - \overline{l})^2}$
Image Blurring [135]					
Blurring Score (σ_{∇^2})	DL	✓	✓	✓	Variance of the laplacian $(\sigma_{\nabla^2}^2) = variance(\partial^2 I/\partial x^2 * \partial^2 I/\partial y^2)$
Blurring Percentage/Image	%	✓	✓	✓	$(\sum_{i}^{n} \mathbb{1}\sigma_{\nabla^2,sub-img}^2 > Blurring_{Threshold})\%$
Blurred Images Percentage	%		✓	✓	$(\sum_{i}^{n} \mathbb{1}\sigma_{\nabla^2,img}^2 > Blurring_{Threshold})\%$
Detectable V-Features(SIFT[102], ORB[153], FAST[152])					
Avg. # Feature/sub-image (F_{avg})	$Features$	✓	✓	✓	$F_{avg} = (F_{total})/(\lceil I_H/b_{dim}\rceil * \lceil I_W/b_{dim}\rceil)$
Avg. Spatial distribution Ratio ($F_{dist-avg}$)	%	✓	✓	✓	$F_{dist-avg} = (\sum_{i=1}^{n} \mathbb{1}(F_i >= F_{avg}))/(b_{total})\%$
Abs. Spatial distribution Ratio ($F_{dist-abs}$)	%	✓	✓	✓	$F_{dist-abs} = (\sum_{i=1}^{n} \mathbb{1}(F_i >= 1))/(b_{total})\%$
Image Disparity					
Avg. Disp.(StereoBM) ($\mu_{D,BM}$) [86]	DL	✓	✓	✓	Average of Disparity Map using StereoBM Method
Std. Dev. Disp.(StereoBM) ($\sigma_{D,BM}$) [86]	DL	✓	✓	✓	Standard dev. of Disparity Map using StereoBM Method
Avg. Disp.(StereoSGBM) ($\mu_{D,SGBM}$) [71]	DL	✓	✓	✓	Average of Disparity Map using StereoSGBM Method
Std. Dev. Disp.(StereoSGBM) ($\sigma_{D,SGBM}$) [71]	DL	✓	✓	✓	Standard dev. of Disparity Map using StereoSGBM Method
Image Similarity [57]					
DBoW2 Similarity Score	DL	✓	✓	✓	DBoW2 score to closet match in the same sequence
Distance to Closest Match	$Frames$	✓	✓	✓	The proximity distance between an image and its closest match in a sequence

* SM-L, SQ-L, and DS-L refers to sample-, sequence-, and dataset level characterization

* DR refers to inertial sensor's dynamic range

* DL refers to dimensionless quantity

sample timestamps, which is given by:

$$Duration_{total} = \sum_{i=2}^{N} (t_i - t_{i-1}) = t_N - t_1 \tag{3.5}$$

The duration is reported for each available sensor in the data acquisition set-up as an indication of the level of synchronization between different dataset sensors.

C. Sample Time of Dataset sensors

The sample time is defined as the time step which a certain data sample represents and is reported in seconds. This can be calculated by measuring the difference between timestamps. The average sample time is an approximation that is usually used to represent the sensors' sampling rates, which can be calculated the definition of total duration mentioned above as:

$$t_{sensor} = \frac{Duration_{total}}{N} \tag{3.6}$$

Instead of using the average sample time for different calculus operations, we use the exact sample time for our differentiation/integration operations throughout the framework. This leads to a better and a more accurate calculation of higher order derivatives and avoids the production of any residual errors as a result of the sample time approximations.

D. Mismatch Between Dataset Sensors

Most of the available SLAM datasets provide their sensor data after some level of post-processing, which usually includes removing a number of data anomalies such as timestamps mismatch. However, this is not the general case when raw data are used directly. However, for any multi-sensor system (such as SLAM) to operate correctly, input information must be synchronized and must be sampled at the exact same time to avoid any data integrity issues which can lead to erroneous estimations. In order to detect these challenges, and with the help of existing timestamps of all dataset sensors, the mismatch between timestamps of different input sensors can be detected. The process is simple, as it relies on finding the closest match for a certain sensor sample in other available sensors. Then the difference in timestamps is calculated and reported as an indication of the level of timestamps mismatch.

3.3.3.2 Inertial Characterization Metrics

Inertial Measurement Units (IMUs) are currently considered a basic building block in industrial/commercial navigation systems. The reason for that, is the considerable gain in terms of accuracy that can be achieved with the integration of visual and inertial sensors either in a loosely-coupled or a tightly-coupled schemes. This is reflected on datasets since the introduction of the New College Dataset [170] and until our current day with tens of introduced datasets in-between. Inertial characteristics of a dataset can provide information on the motion profile, the aggressiveness of motion, and the composition of motion with respect to different axes in a coordinate system.

Inertial measurements can be referenced to either the body frame or the inertial global frame [125]. Commonly, inertial data in SLAM datasets are presented in body frame [163][22]. This is, usually, coupled with alignment information of the the system sensors with respect to each other. The existence of a either a INS/GNSS system or a reference system is needed if body frame-referenced inertial data are needed to be referenced in the local-level frame. The relation between the two systems of axes is determined by a homogeneous rotation matrix R_b^l, which can be used for executing the conversion of system axes when needed.

A. Higher-Order Time Derivatives of Inertial Data

Statistical analysis of inertial data is commonly used as an indicator of the boundaries of motion profiles the dataset are experiencing. Moreover, more information can be deduced from the statistical analysis of higher order time derivatives of inertial measurements in the body frame space. For instance, *Jerk (j)* and *Snap S* [161], given in Equ. 3.7 and 3.8, are defined to be the first and second order time-derivatives of acceleration, which are a commonly used quantities to measure the smoothness of motion and the level of sudden/abrupt changes a certain rigid body is experiencing. Similarly, first and second order time-derivatives of angular velocity data known as the *angular acceleration* α and *angular jerk* φ, provided in Equ. 3.9 and 3.10, are indicators of the aggressiveness and smoothness of the rotation performed by the carrier holding

the system sensors.

$$\vec{j^b} = \frac{d\vec{f^b}}{dt} \tag{3.7}$$

$$\vec{S^b} = \frac{d\vec{j^b}}{dt} = \frac{d^2\vec{f^b}}{dt^2} \tag{3.8}$$

$$\vec{\alpha^b} = \frac{d\vec{\omega^b}}{dt} \tag{3.9}$$

$$\vec{\varphi^b} = \frac{d\vec{\alpha^b}}{dt} = \frac{d^2\vec{\omega^b}}{dt^2} \tag{3.10}$$

Where: $\vec{j^b}$, $\vec{S^b}$, $\vec{\alpha^b}$, and $\vec{\varphi^b}$ are the jerk, snap, angular acceleration, and angular jerk. The quantities provided are 3-dimensional vectors representing the three sensing axes of accelerometer and gyroscope referenced to the body frame.

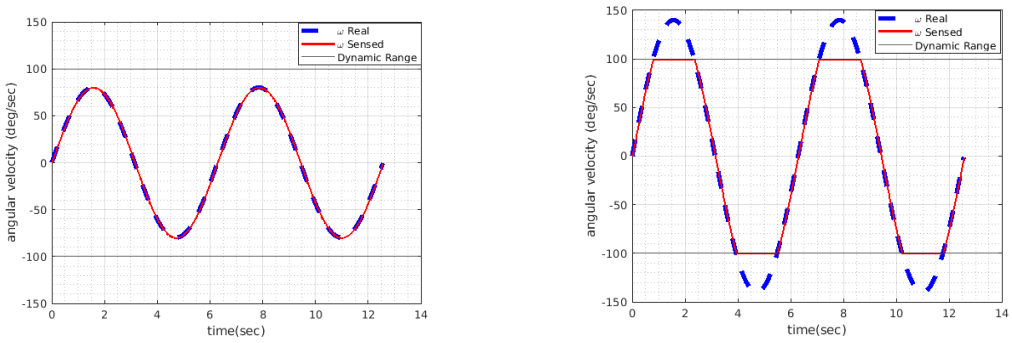
Furthermore, the relative location of the IMU and the camera and the accuracy of alignment and referencing to a single unified system of axes can differentiate the angular motion from being a spinning motion or an orbiting motion. The existence of one type of motion while having physical constraints from the carrier itself can cause more uncertainty in the measurements and will lead to erroneous motion estimation by the SLAM algorithm. For instance, a UAV can have a spinning motion, while a car cannot due to its structure and principles of motion imposed by its mechanical structure.

Higher-order derivative of motion profiles are of great importance in modern control systems due to the reliance of control systems on them for controlling the motion of the complete physical plant. Controlling higher-order derivatives is used in order to provide more stability and robustness [126] to the controlled plant.

B. Sensors Saturation

Examination of the inertial sensor measurements against the dynamic range can be used to detect the occurrence of sensor saturation, which is an indication of possible data loss by the data clipping experienced. It is also an indication

that the motion is beyond the measurements limits of the IMU, and thus an indication of the wrong choice of the sensors configuration in the dataset collection set-up.



(a) (b)

Figure 3.3: An example of inertial sensor saturation and data loss anomaly. (a) Represents a physical signal within the sensor dynamic range. (b) Represents a physical signal beyond the sensor's dynamic range and illustrating the data clipping/loss witnessed

Fig. 3.3 illustrates the importance of data saturation checking. It shows two different angular velocity measurements. The first one is not experiencing any data clipping or wrapping as it is within the sensor's dynamic range, while the second one is experiencing data loss due to information clipping when the physical quantity measured is exceeding the sensor's dynamic range. In the later case, IMU measurements are unreliable due to the data clipping witnessed. This data loss can cause odometry modules to produce unreliable odometry measurements and can degrade the performance of the whole SLAM solution if not corrected by aiding odometry sources or detected by implicit measurements reliability/integrity checks.

C. Detection of Rotation-Only Motion Profiles

Rotation-only motion profiles is the scenario where the carrier is only performing rotation around its center-of-mass with no translation in any axis. This motion profile imposes a challenge on traditional SLAM systems that rely on keyframes for motion estimation. The rotation-only motion profile forces the keyframe selection engine to drop a huge number of frames, resulting in a very sparse keyframe vector which causes both the mapping and localization

to fail [138]. Due to this fact, SLAM datasets need to be characterized for the detection of such motion profiles.

The detection of rotation-only motion profiles can be done either by analyzing the ground-truth localization data provided with datasets or by analyzing the acceleration data provided as part of the raw data of the dataset. The former method is straightforward and is done by extracting the translation vector of the carrier and detecting any non-zero components present. The process can be enhanced by setting a threshold to absorb the small noise margin the data acquisition system may be subject to.

The later method of analyzing the accelerometer data depends on the principle of operation of the accelerometer itself. The accelerometer is capable of measuring any linear acceleration (in case of linear motion) and the gravity vector as well [125]. Although the acceleration measurements are provided in the body frame, the existence of linear acceleration can be detected by measuring the total acceleration magnitude, and comparing it to the gravity value. The comparison is augmented by an uncertainty range of 10% to avoid any acceleration noises present. This ratio is added to accommodate for non-ideality of inertial sensors, as well as for the compensation of the Corlios effect. Thus, The acceleration magnitude can be calculated by:

$$f_{magnitude} = \sqrt{f_{x,b}^2 + f_{y,b}^2 + f_{z,b}^2} \tag{3.11}$$

Where $[f_{x,b}, f_{y,b}, f_{z,b}]^T$ is the acceleration vector in the body frame. The acceleration magnitude $f_{magnitude}$ is then tested to detect linear translation as follows:

$$T_{status} = \begin{cases} 1, & \text{if } f_{magnitude} \leq 9.81 \pm 10\% \\ 0, & \text{otherwise} \end{cases} \tag{3.12}$$

The former method is useful when the ground-truth localization information are provided with the dataset, which is the case for most of the released SLAM datasets. While the later method is specifically useful in the situation where the ground-truth data are not available.

3.3.3.3 Visual Characterization Metrics

Visual sensors is the most commonly used sensors in the SLAM datasets and benchmarks literature. This is a direct indication of the important of such sensors and the implicit indication of the utilization of vision as a basic and fundamental building block in both legacy and modern SLAM systems. Thus, the characterization of the visual data provided in datasets is essential and critical to stand on the operating boundaries of SLAM systems. Visual information is rich in nature and provides a wide range of insights into the scene a SLAM system is deployed in. However, this implies a huge challenge in the process of characterization of such information due to its diversity and richness. Thus, in the proposed framework, a number of characterization metrics were picked based on their direct impact on the front-end sub-system of SLAM solution, and also based on the claimed qualitative characteristics of any newly proposed SLAM dataset. The generic nature of our proposed framework is a key feature of it, and is a key enabler for future extension and alteration of characterization metric with limited development efforts.

A. Detectable Visual Features

A fundamental building block in visual SLAM systems is the feature extraction and matching block. The process starts with feature extraction from subsequent images using one of the famous feature extractors such as SIFT [102], SURF [13], ORB [153], among many others. Features are then matched between each two consecutive images and the matches are then used in order to estimate both the rotation and the displacement which was performed by the camera. This process is known to be very critical in the SLAM pipeline [24], as the accuracy of the matching governs the accuracy of the estimated motion that, in turn, affects the accuracy of both localization and mapping of the SLAM solution. Motion estimation can be performed with a limited number of keypoint matches, however, with an extended number of feature matches, the motion estimation process becomes over-constrained, resulting in more accurate estimations. On the other hand, the less detectable features, the more challenging the SLAM process becomes. This highlights the need

to have guarantees in the systems to deal with such situations is crucial to achieve a robust performance under these challenging conditions.

Subsequently, the measurement of the number of detectable features in datasets as well as their spatial distribution across the image are needed to indicate the level of challenge a certain dataset imposes on the SLAM system. This measurement shall be done on the image level, sequence level, and finally the whole dataset level.

In the proposed framework, we start from the image level, where we utilize three visual feature extractors which are: SIFT [102], FAST [152], and ORB [153] to report the total number of detectable visual features in an image. Then, we divide the image into square-shaped bins with configurable dimensions. After that we calculate the average number of feature per bin, which is given as:

$$F_{avg} = \frac{F_{total}}{\lceil I_H / b_{dim} \rceil * \lceil I_W / b_{dim} \rceil} \tag{3.13}$$

where: F_{avg} is the average features per bin, F_{total} is the total number of detectable features in an image, I_H and I_W are the height and width of the image, and b_{dim} is the configurable dimension of the image bin.

After that, we compare the number of features in each bin F_b with the average number of features per bin F_{avg} to decide on whether the features are uniformly distributed or spatially biased in the image. If a the number of features in a certain bin is greater than or equal to the average number of features per bin, then, the bin is marked to have enough a proper number of features, and vice versa otherwise.

The number of bins with features that are greater than or equal to the average number of features per bin F_{avg} are then counted to calculate the percentage of bins with above average detectable features count using the following formula:

$$F_{dist-avg} = \frac{\sum_{i=1}^{n} \mathbb{1}(F_i >= F_{avg})}{b_{total}} \% \tag{3.14}$$

Additionally, the number of bins with any number of features presented

are also counted as anther indication of how well the features are spatially distributed

$$F_{dist-abs} = \frac{\sum_{i=1}^{n} \mathbb{1}(F_i >= 1)}{b_{total}}\% \tag{3.15}$$

where: $F_{dist-avg}$ is the percentage of image with a number of detectable features greater than or equal to the average number of features per pin, $F_{dist-avg}$ is the percentage of detectable features greater than or equal to 1, F_i is the number of features in a certain bin, F_{avg} is the calculated average number of features per block from equ. 3.13, and b_{total} is the total number of bins available in an image.

Subsequently, we represent an image with the total number of detectable features, and the percentage of the image which is occupied with these features. These values are then propagated to the sequence level, where other images values are also computed and then statistical values are extracted indicating the average, maximum, minimum, and standard deviation of these values in a certain sequence. They are also propagated to the dataset level, and the same statistical values are computed. The main objective is to represent the whole dataset with quantitative measurements to allow proper objective comparison among different datasets.

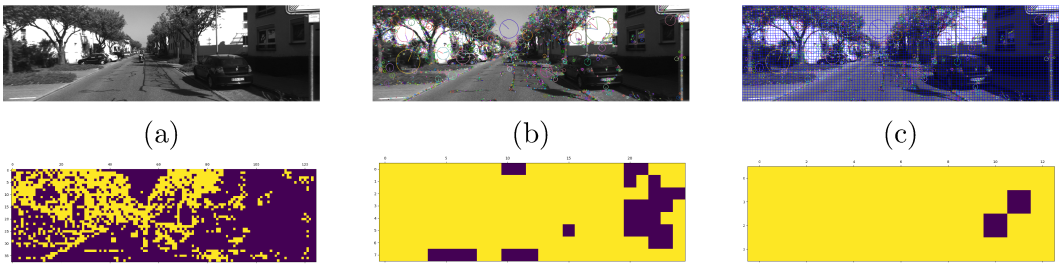


Figure 3.4: Detectable Features Analysis on A Sample Image from KITTI Seq.00 where (a), (b), and (c) represent the original image, detected SIFT features, and the image bins. While (d), (e), and (f) shows the spatial distribution of features when bin=10,50, and 100

In Fig. 3.4, where an image from sequence no. 0 in KITTI odometry dataset [59] and the SIFT [102] features were used to illustrate the process. In this example, we used three different square patch sizes which are are 10,

50, and 100 pixel. As shown, we can observe that with the increased bin size, the spatial distribution bias of visual features fades away. It also shows how the spatial distribution of the features in the image can provide insights of reliability of detected visual features.

B. Image Disparity Levels

A wide spectrum of modern SLAM solutions support stereo vision due to its usefulness in the detection of image depth and the lack of problem of scale ambiguity that is present in monocular vision solutions [158]. Consequently, most of the available SLAM datasets provide stereo vision data. One of the challenges a SLAM solution faces is the range of disparity values, which is directly related to the depth by the following equation:

$$Disparity = x - x' = \frac{Bf}{Z} \tag{3.16}$$

where x and x' are the 2D projections of a 3D point in two stereo image pair, B is the baseline between the two stereo camera, f is the focal length of the camera, and finally Z is the depth of the 3D point.

In order to characterize the range of disparity values, we calculate the disparity image of the input image pairs and report the average image disparity $\mu_{Disparity}$ and the standard deviation $\sigma_{Disparity}$ of each resulting disparity image. Two different algorithms were used in order to diversity the characterization process which are: the Block Matching algorithm (StereoBM in OpenCV) [86] and the Semi-Global Block Matching algorithm (StereoSGBM in OpenCV)[71]. Similar parameters were provided to both algorithms in order to ensure the fairness of comparison.

C. Revisit Frequency and Image Similarity

A wide range of visual SLAM algorithms depend on the existence of loop-closure opportunities to correct the accumulated errors, which is not guaranteed to exist when the system is deployed in an environment or tested on a dataset different from what it was trained to tackle. In fact, the lack of loop-closure opportunities adds to the challenge a SLAM system has to tackle, and impact the system robust when these opportunities are rare or non-existing.

Thus, the revisit frequency in SLAM datasets has to be quantitatively measured and reported to expose the dependency of the SLAM solution accuracy and loop-closure.

A well-known and widely-used technique for loop-closure detection is the bag-of-visual-features method, where images are described by a histogram of visual features' frequencies [169]. These visual features are called the visual dictionary of the vocabulary and are the result of clustering the visual features space [8]. In the proposed framework, we utilize DBoW2 [57] which is used by a number of modern state-of-the-art SLAM solutions for loop-closure detection such as RTAB-Map [91], ORB-SLAM 1,2, and 3 [120][121][25], and VINS-Mono [146], among many others. Moreover, we utilize the visual vocabulary provided with the ORB-SLAM series (ORBVoc) as the base vocabulary for image descriptors due to its diversity.

A number of metrics can be extracted to indicate the level of image similarity in the sequence in hand. The DBoW2 [57] provides a similarity score between the query image and the closest match in the sequence loaded to the database. This score, denoted as LC_S, is recorded. Additionally, the distance to the closest match, denoted by LC_D and represented in frames, is recorded as an indicator of the level of dynamics in the sequence. The calculated score LC_S is checked for three thresholds which are $S = 1.0, 0.9, 0.5$ and the number of matches above the aforementioned scores is recorded. In the original DBoW2 [57] work, a score of $S = 0.3$ was considered a successful loop closure opportunity.

D. Motion Blurring

Motion blurring can be a result of the motion of either the camera or the objects inside the camera's field of view. In the former case, motion blur is expected to be global and its impact shall be manifested on the whole image, while in the latter case, we can witness local blurring of specific objects in the image. Consequently, the detection of each type of image blurring is different and both are important characterization metrics of SLAM datasets as they pose two different types of challenges on the SLAM algorithms and

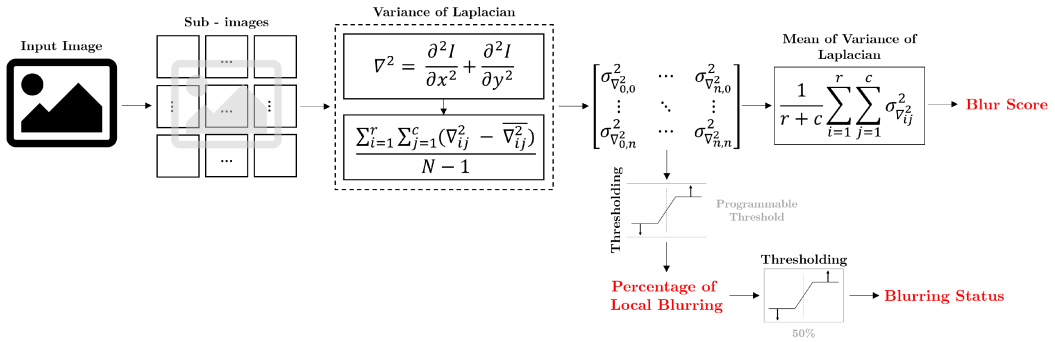


Figure 3.5: Detection and Characterization of Image Blurring

systems when faced with such situations. A wide range of blur detection algorithms and techniques are available in the literature and are summarized in [135]. The techniques mentioned in this work differ in terms of the limitations, the suitability for both local and global blur detection, and in terms of the tunable hyper-parameters impacting the performance. One of the simplest, yet effective, methods for blur detection is measuring the statistical variance of the image after applying a high-pass filter on it. This depends on the assumption that blurring detection is synonymous to the lack of edges, which are manifested in the high-frequency spectrum of the image. In openCV, applying the Laplacian operator ∇^2 is equivalent to applying a high-pass filter on the image. The Laplacian of an image I can be given by:

$$\nabla^2 = \frac{\partial^2 I}{\partial x^2} . \frac{\partial^2 I}{\partial y^2} \tag{3.17}$$

Subsequently, the variance of the Laplacian operator can be given by:

$$\sigma_{\nabla^2} = \frac{\sum_{i=1}^{N}(\nabla_i^2 - \bar{\nabla}^2}{N-1}) \tag{3.18}$$

where N is the number of pixels of the image.

In order to detect whether a given image is considered blurred or not, the variance of Laplcian score must be compared to a programmable threshold $threshold_{blur}$ to decide on whether the image is blurred or not. Similarly, the same concept can be applied to detect local blurring in visual images by dividing a given image I into a number of sub-images with a user-defined dimension d_{si}. The same operator and thresholding mechanism can be applied

on sub-images and thus detect localized blurring.

The resulting matrix of blurring scores of sub-images can further be utilized to detect the percentage of blurring in images. Moreover, the declaration of an image being blurred or not is now subject to the number of blurred sub-images. In the proposed framework, an image is considered blurred if more than 50% of its sub-images are blurred. The process is illustrated in Fig. 3.5.

In order to test the technique proposed, four images were selected from KITTI Odometry [59] and EuroC MAV [22] datasets. Then, an induced blurring was super-imposed on the selected images, and the algorithm was applied to determine whether it will successfully detect the blurring percentage induced or not. Table 3.2 provides a summarization of the results of applying the segmented variance of Laplacian method on selected sample images.

Table 3.2: Blurring Detection Results using Segmented Variance of Laplacian Method with bin size = 50px and blur threshold = 50

		Blur Ratio		
Dataset_Sequence - Image	**Score**	**Induced**	**Detected**	**Blurred?**
EUROC_MH_01_easy - SI#1	401.21	25 %	23.8 %	No
	280.54	50 %	53.1 %	Yes
	176.45	75 %	73.1 %	Yes
	5.87	100 %	100 %	Yes
EUROC_MH_05_difficult - SI#2	244.88	25 %	30 %	No
	185.19	50 %	55 %	Yes
	94.83	75 %	77.5 %	Yes
	5.99	100 %	99.4 %	Yes
KITTI_00 - SI#3	557.24	25 %	36.5 %	No
	503.24	50 %	49.5 %	No
	134.87	75 %	82 %	Yes
	4.2194	100 %	100 %	Yes
KITTI_08 - SI#4	924.72	25 %	35 %	No
	590.79	50 %	57.5 %	Yes
	160.77	75 %	81 %	Yes
	3.75	100 %	100 %	Yes

It can be shown that, the algorithm was able to detect the percentage of induced blurring in all of the introduced images with a small error. The error is a result of the existence of blurring in the sample images before inducing the simulated blurring and due to the empirical tuning of the blurring threshold on which a sub-image is declared blurred or not.

The utilized algorithm has some limitation due to its sensitivity to the value of the threshold $threshold_{blur}$. Another limitation stems from the nature of

the scene we wish to characterize. For instance feature-less scenes such as clear skies cannot be characterized by the aforementioned measure. However, with the combination with other metrics (e.g. detectable visual features metrics), one can identify the reliability of the blurring score and results provided.

E. Image Brightness

Image brightness is one of the metrics that can pose a challenge on SLAM algorithms, specially when the image brightness is biased or skewed towards being either too dark or too shiny. However, these terms need to be quantified for the sake of comparison. A number of operators are being used the academia and the industry to calculate the brightness of each pixel in an image. Some of these operators, their differences, and their limitations were mentioned and discussed in [16] such as arithmetic mean, HSV color scheme, BCH color scheme, and YUV color scheme.

In our proposed framework, we depend on the openCV library's definition of brightness which depends on the definition of Luma used in image compression algorithms (e.g. MPEG ... etc), which is given by:

$$B = 0.299 * R + 0.587 * G + 0.114 * B \tag{3.19}$$

To have a single brightness characterization metric for the whole image brightness, we calculate the geometric mean of all brightness values (grayscale image intensities) and report this value to be the average image brightness.

On the sequence level, rapid and abrupt changes in image brightness put more challenges into the path of SLAM algorithms, and thus, the characterization of such phenomena is of great importance. The change of brightness can be characterized by calculating the first-order time derivative of image brightness values $\frac{dB}{dt}$. A new variable β_B is defined to be the zero-mean first-order time-derivative of brightness and is given by:

$$\beta_B = \frac{dB}{dt} - \mu_B \tag{3.20}$$

This zero-mean rate of change of image brightness given by the variable β_B

is then examined against whether it exceeds the values of $|\sigma|$, $|2\sigma|$, and $|3\sigma|$. The number of samples exceeding each threshold is counted and the ratios of the changes w.r.t. the size of the sequence are calculated.

F. Image Exposure

A closely-related metric to image brightness is image exposure, which is an indication of the quality of the lighting condition of the image. In order to measure image exposure and to detect whether an image is either properly-exposed, over-exposed, or under-exposed, the image intensity histogram is to be constructed. Additionally, the mean μ, variance σ, and skewness S values are to be calculated. Due to the fact that images may have some pixels with extreme intensity values, we utilize the concept of *trimmed mean* [194] instead of the *arithmetic mean* that is usually used, which mandates dropping a percentage of pixels α with extreme intensity values present on the two ends of the histogram. The *trimmed mean* is defined by[150]:

$$\mu_\alpha = \frac{1}{n - 2[n\alpha]} \sum_{i=n-[n\alpha]}^{[n\alpha+1]} I_i \tag{3.21}$$

Subsequently, the standard deviation and the skewness can be defined as:

$$\sigma_\alpha = \sqrt{\frac{\sum_{i=n-[n\alpha]}^{[n\alpha+1]} (I_i - \mu_\alpha)}{n - 2[n\alpha]}} \tag{3.22}$$

$$S_\alpha = \frac{\sum_{i=n-[n\alpha]}^{[n\alpha+1]} (I_i - \mu_\alpha)^2}{(n - 2[n\alpha] - 1) * \sigma_\alpha^3} \tag{3.23}$$

As shown in Fig. 3.6, a bias in the location of the trimmed mean μ_α can be one indication of the improper exposure of the image. Also, the selectivity of the trimmed standard deviation σ_{alpha} and the signal skewness adds more emphasis on the detected anomaly. The objective is to detect whether an image is properly exposed, over-exposed, or underexposed. For that purpose, the image intensity histogram is divided into seven regions: two discarded regions, each representing 5% of the dynamic range, that are of extreme intensity and

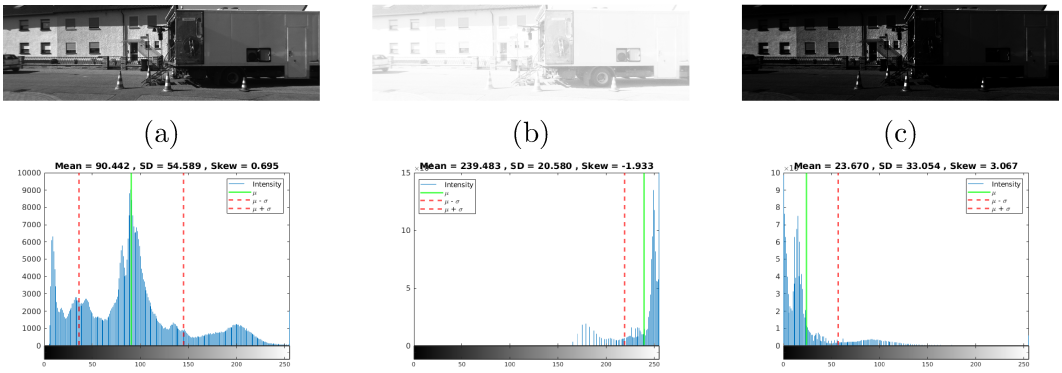


Figure 3.6: The relation between the image exposure and the intensity histogram. (a), (b), and (c) show the same image with different exposure levels: original, over-exposure, under-exposure. (d), (e), and (f) show the corresponding intensity histograms, intensity mean, and intensity variance.

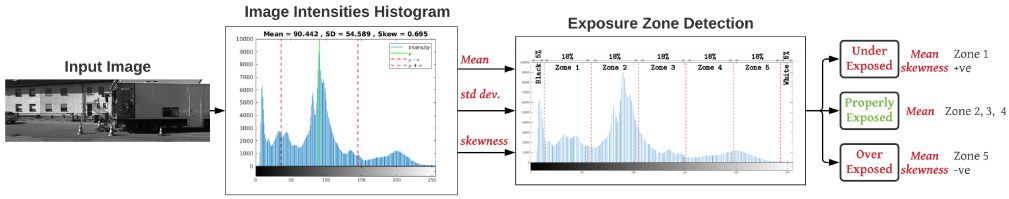


Figure 3.7: Image Exposure Level Detection

five intensity regions, each representing 18% of the intensity dynamic range. In Fig. 3.7, the exposure level detection mechanism is described. For each image, we examine the values of the trimmed mean μ_α and the skewness S_α. An under-exposed image is defined to have a positive S_α and a zone #1 μ_α. On the other hand, an over-exposed image is defined to have a negative S_α and a zone #5 μ_α. Otherwise, the image is considered to be properly exposed. The process can be summarized as follows:

$$Exposure_{Status} = \begin{cases} BL, & \mu_\alpha \in Zone\#0 \\ UE, & S_\alpha > 0 \,\&\, \mu_\alpha \in Zone\#1 \\ OE, & S_\alpha < 0 \,\&\, \mu_\alpha \in Zone\#5 \\ WL, & \mu_\alpha \in Zone\#6 \\ PE, & \text{otherwise} \end{cases} \tag{3.24}$$

where BL, UE, OE, WL, and PE refer to black image, under-exposed, over-

exposed, white image, and properly-exposed image status.

G. Image Contrast

Intuitively, image contrast C is a property of that represents the differences in pixel colors and brightness. These differences play a vital role in making these images both identifiable and distinguishable. Quantitative measurement of image contrast is a long discussed problem in the literature, however, there exist a number of definitions, outlined and discussed in [134], that are practically used, which we briefly discuss.

The first method, which is considered the simplest, is the *Contrast Ratio* and is given by:

$$CR = \frac{L_{target}}{L_{background}} * 100 \tag{3.25}$$

The second method is given by the *Weber Formula* and is given by:

$$C_{Weber} = \frac{L_{target} - L_{background}}{L_{background}} * 100 \tag{3.26}$$

The third method is based on the *Michelson Formula* and is given by:

$$C_{Michelson} = \frac{L_{max} - L_{min}}{L_{max} + L_{min}} \tag{3.27}$$

While the last method is measuring the *root-mean square (RMS)* of the image pixels' intensities, and can be given by:

$$C_{RMS} = [\frac{1}{n}\sum_{i=1}^{n}(l_i - \bar{l})^2]^{\frac{1}{2}} \tag{3.28}$$

where: L_{max}, L_{min}, and $L_{background}$ are the maximum, minimum, and background luminance values in the image extracted from the image after the conversion to the LAB colorspace. l_i is the pixel intensity at pixel i, and $\bar{l}$ is the normalized mean of the image pixels' intensities. The contrast ratio, Weber contrast, and Michelson contrast values are scaled by 100 for visualization purposes.

The methods used measure the ratio of luminance change in the image w.r.t. the background luminance. However, they differ in the dynamic range

in which the outputs are provided and in the corner cases where they provide unexpected behaviour. The suitability of the aforementioned methods for measuring contrast is also on the long discussed issues in the literatures. For instance, having one pixel with extreme intensity values can bias the contrast measurement, which is not a proper representation of image contrast. However, we believe that the contrast measurements shall be evaluated w.r.t. the application. In the SLAM case, having pixels with extreme values can impact the reliability of some of the SLAM building blocks such as feature matching and tracking unless some pre-processing is done. Thus, using the contrast measurement to indicate such anomalies may have its usefulness in the SLAM context.

3.4 Experimental Setup

To systematically analyze and compare measured characteristics, dataset characterization techniques from the field of data science [132] are used. The analysis includes: *Statistical Analysis*, *Diversity And Interestingness Analysis* where Shannon Entropy (H) and Simpson Diversity Index (SDI) are calculated, and *Correlation Analysis* where Pearson Correlation Coefficient ($PPMC$) is measured between the characterization metrics and performance metrics of SLAM algorithms. The analysis metrics are provided in Table. 3.3.

Table 3.3: Dataset Characterization Results' Analysis Metrics

Analysis Metric	**Formula**	**Parameters**
Statistical Analysis	μ, mid, σ, σ^2, S	N total number of samples x input vector
Entropy (H)	$H = -\sum_{i=0}^{k} p_i \, log_2 \, (p_i)$	k no. of unique values p_i prob. associated
Simpson Diversity Index (SDI)	$SDI = 1 \; - \; \frac{\sum_{i=1}^{k} n_i(n_i-1)}{N(N-1)}$	N total number of samples k no. of unique values n_i no. of samples for a unique value
Pearson Correlation Coefficient (PPMC)	$PPMC(x,y) = \frac{N\sum(x.y)-\sum(x)\sum(y)}{\sqrt{[N\sum(x^2)-(\sum(x)^2)].[N\sum(y^2)-(\sum(y)^2)]}}$	N total number of samples x first characterization metric vector y second characterization metric vector

The framework has been used to characterize three datasets which, together, are considered a classical standard for benchmarking SLAM. These datasets are KITTI Odometry [59], EuroC MAV [22], and TUM VI [163]. Due to the extendability and generality of the framework, support for additional

characterization parameters, sensors, or datasets is possible with minimal development efforts.

The importance of the characterization becomes evident when one takes a deeper look at the results as many insights can be extracted and measured. The following points highlight some of the benefits for the SLAM research and development community. The framework provides a foundation for a complete ecosystem for a systematic and a reliable methodology for SLAM algorithms development, testing, and performance evaluation.

3.5 Results and Discussion

In this section, we present the characterization results of the three datasets previously mentioned. As we will show, the framework proposed provides a plethora of tools for characterization and analysis of datasets on the measurement level, the sequence level, and the dataset level. Those tools enables in depth understanding of the boundaries of situations implied by a dataset and paves the way to systematic measurement of robustness and resilience in SLAM.

3.5.1 Dataset Diversity and Interestingness

In Figure 3.8 and Figure 3.9, entropy and SDI analysis results are presented. It can be shown that although the three datasets are very diverse, the amount of information measured by the entropy and the SDI vary due to the nature of the dataset formulation. Moreover, one can observe that the TUM-VI dataset is superior in diversity when compared to EURO-C MAV and KITTI in terms of both general and visual characterization. However, the diversity of TUM-VI and EURO-C MAV is almost the same when it comes to inertial characteristics.

3.5.2 Dataset Anomalies

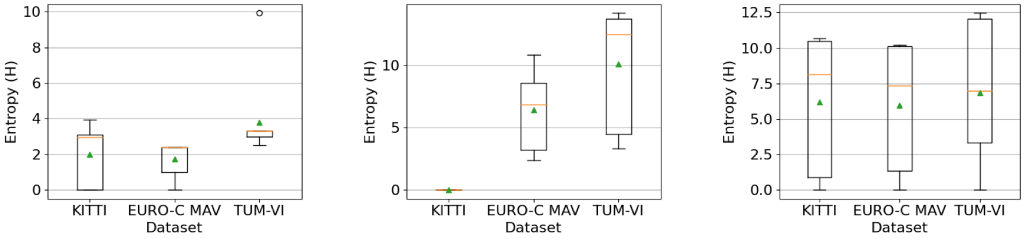
Dataset anomalies can be detected using the proposed framework. For instance, sensor timestamp mismatch has been detected in the TUM-VI dataset,

Table 3.4: Statistical analysis of Entropy Results

	mean±std. dev		
	KITTI	EURO-C	TUM-VI
General Metrics	2.0±(1.67)	1.73±(0.86)	3.81±(2.05)
Inertial Metrics	0.0±(0.0)	6.42±(2.83)	10.12±(4.39)
Visual Metrics	6.17±(4.33)	5.95±(4.03)	6.86±(4.46)

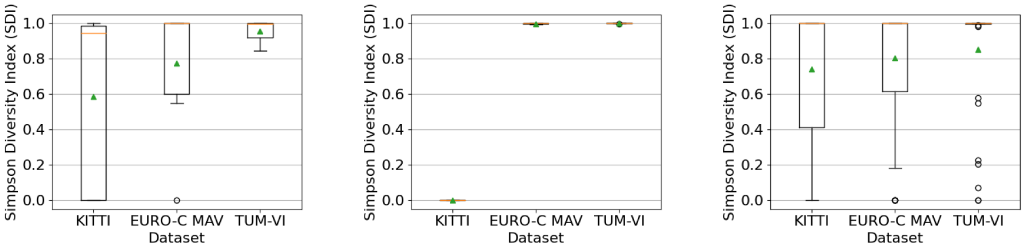
Table 3.5: Statistical analysis of SDI Results

	mean±std. dev		
	KITTI	EURO-C	TUM-VI
General Metrics	0.59±(0.48)	0.78±(0.32)	0.96±(0.06)
Inertial Metrics	0.0±(0.0)	1.0±(0.0)	1.0±(0.0)
Visual Metrics	0.74±(0.42)	0.8±(0.35)	0.85±(0.32)



(a) (b) (c)

Figure 3.8: Entropy analysis of (a)general, (b)inertial, and (c)visual characterization results respectively

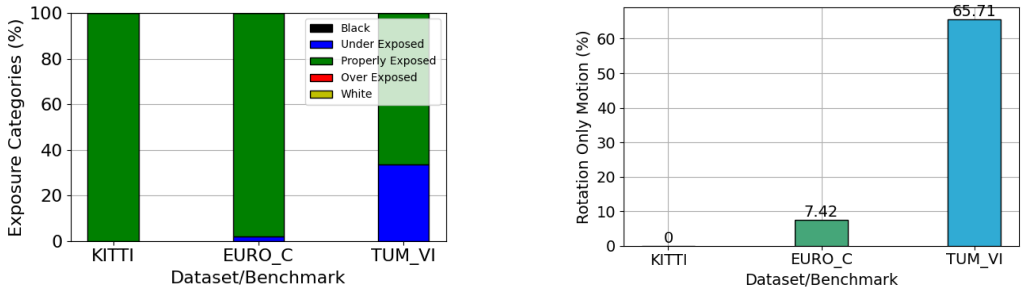


(a) (b) (c)

Figure 3.9: Simpson's Diversity Index (SDI) analysis of (a)general, (b)inertial, and (c)visual characterization results respectively

and this imposes a data integrity issue for SLAM. Additionally, over-exposed and under-exposed images have also been detected and localized in both Eu-

roC and TUM-VI datasets as shown in Figure 3.10a. Moreover, rotation-only motion profiles have been detected in EuroC and TUM-VI as shown in Figure 3.10b, and this implies a challenge on conventional SLAM solutions. The existence of these anomalies proposes that the TUM-VI dataset is more diverse in terms of existence of data anomalies compared to the other two datasets.



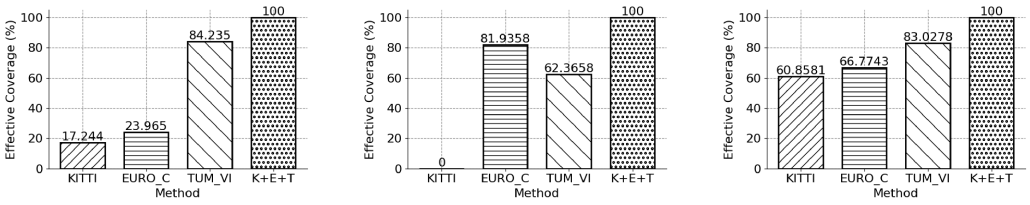
(a) Exposure Zones

(b) Rotation-only Motion

Figure 3.10: Example dataset anomalies detected using the analysis of the characterization results

3.5.3 Operating Conditions and Measurement of SLAM Robustness/Resilience

The operating conditions and dynamic ranges of a SLAM system can finally be defined quantitatively using the proposed framework. In Figure 3.11, datasets are compared based on the level of dynamic range coverage achieved compared to the usage of the three of them combined. From the perspective of general and visual characterization metrics, TUM-VI is superior in terms of the achieved coverage. However, the EuroC-MAV dataset achieved a higher coverage when it comes to inertial characterization metrics. Claimed performance of SLAM is now tied to an operating range which is a direct indication of how robust the system is. Subjecting a SLAM system to perturbation beyond its operating conditions until divergence is also possible and is a measure of the system rating (operating limits) in which a system can survive. With the knowledge of the operating limits, a measurement of resilience can be realized. This can be achieved by subjecting SLAM to conditions beyond the operating limits and measuring the time to convergence and the associated



(a) (b) (c)

Figure 3.11: Dynamic range coverage percentage of (a)general, (b)inertial, and (c)visual characterization metrics of each dataset compared to their aggregation

accuracy.

3.5.4 Dataset Redundancy

Redundancy can either happen within a dataset among its sequences or among different datasets used for the same evaluation process. The detection of either can better guide the design of experiments via the proper selection of datasets or sequence mixes. This can pave the way to a systematic procedure for SLAM evaluation and will act as an entry point for *task-centric* selection of the evaluation and testing methodology and data. Moreover, the novelty of any newly introduced dataset can be assessed in comparison to what we already have via a systematic evaluation/comparison engine. For example, coverage analysis was conducted on the blurring score of images. The results provided in Figure 3.12 suggest that only two sequences from the TUM-VI dataset (Seq. 7 and Seq. 16) are sufficient for testing SLAM immunity to blurring as they provide the same coverage levels achieved by testing all the three datasets for blurring. Subsequently, the testing time of SLAM against blurring was reduced from running sixty-one sequences from three different datasets to only running two sequences of the TUM-VI dataset.

3.5.5 Dataset and Performance Correlation Analysis

The availability of dataset characteristics/features can be used in analyzing the correlation among characteristics or between characteristics and SLAM performance revealing the underlying relations between the two and pin-pointing the

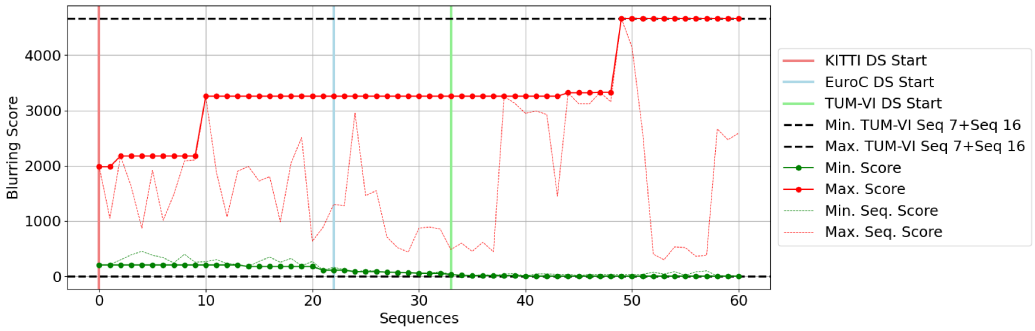


Figure 3.12: Blurring Score Coverage Analysis

causes for system failures, performance degradation, or system sensitivity to input data. Consequently, an important aspect of this work is the illustration of how dataset characterization can lead to predictable SLAM performance. Thus, in Figure 3.13, the correlation between the ATE of SLAM and the mean of characterization results is presented. We can observe the existence of highly correlated characterization metrics with ATE, which shows the significance of the selected characterization metrics and their suitability for usage as a dataset descriptor.

3.5.6 Limitation of the characterization framework

Characterization of SLAM data aims at providing a sufficient descriptor of the dataset on the measurement, sequence, and dataset level. One challenge in this regard is whether the selected hand crafted characterization metrics are sufficient to provide complete description of the data. This can be validated by performing regression analysis where we try to deduce SLAM performance given the selected metrics in a univariate and multivariate set-up.

Additionally, the proposed framework can only characterize visual-inertial datasets, which is a limitation given the increased dependency on LiDARs as a basic and fundamental sensor. However, the proposed framework can be extended to other sensors and metrics easily with very limited efforts due to its modular design.

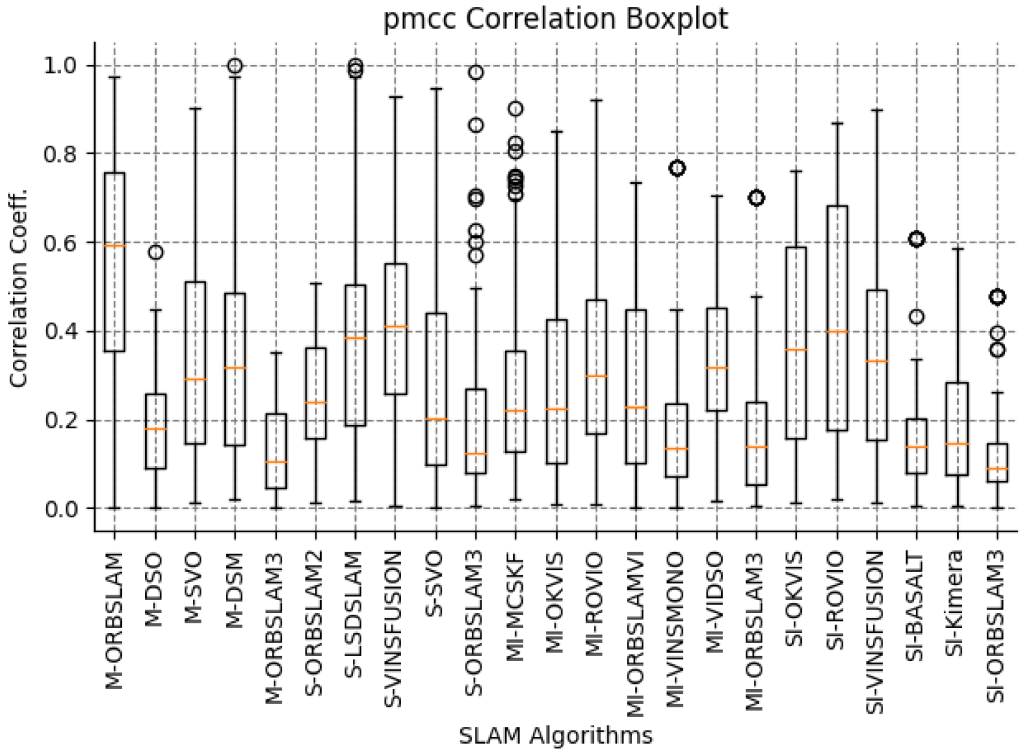


Figure 3.13: PMCC Correlation Coefficients between characterization metrics mean for each sequence and the reported ATE of selected SLAM algorithms. (M, S, MI, SI) refer to (Mono, Stereo, Mono-Inertial, Stereo-Inertial) operating modes.

3.6 Summary

The key points of this chapter can be summarized as follows:

- The problem of quantitative characterization of SLAM datasets is discussed, demonstrating how it can measure operating ranges and conditions, thus serving as a key to measuring robustness and resilience.
- A novel dataset characterization framework has been introduced and described in detail, providing a systematic methodology for designing experiments by exploiting certain environment conditions.
- The framework is used to characterize three different datasets, showcasing its capabilities in measurements and visualization, highlighting

undiscovered anomalies, and opening the door to various research studies.

- The link between data characteristics and algorithm performance can be established, leading to a systematic evaluation methodology for SLAM system research and development, while guiding the introduction of new datasets by detecting anomalies and providing measures for diversity, redundancy, and coverage in previously introduced ones.

Chapter 4

Optimization of the SLAM Evaluation Process

4.1 Introduction and Motivation

The robustness and resilience of SLAM cannot be determined, guaranteed, or transferred without the quantification of the design and testing conditions first, and then the validation and deployment conditions [183]. The performance of SLAM is often evaluated by subjecting it to a temporal sequence of sensor measurements in the form of a pre-recorded dataset. Thus, this gives rise to the need to provide quantitative characterization of such datasets.

Usually, SLAM evaluation begins with the selection of a number of datasets where each dataset consists of a number of sequences. The selection of the datasets and their corresponding sequences are thought to capture the environment conditions and anomalies sufficiently to establish an objective evaluation of the proposed SLAM algorithm. This selection has typically been done qualitatively without an explicit consideration of the dynamic range of a certain environmental parameter. In fact, as this study proposes, there exists a huge level of redundancy and similarity among datasets and among measurement sequences in the same dataset. Thus, we attempt to analyze and identify the coverage of dynamic range achieved by a certain experimental setup, which plays an important role in quantifying the robustness of a SLAM system. This identification can lead to an optimal selection of testing sequences to achieve the same level of coverage in terms of time and effort. With the aid of the

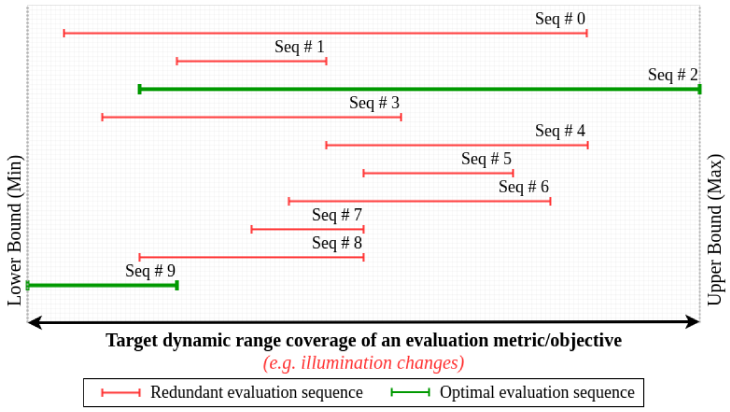


Figure 4.1: An illustration of how evaluation of SLAM is usually conducted and the level of redundancy present in the process (red), and the subset of optimal evaluation sequences given a defined evaluation objective (green)

characterization framework proposed in [5], one can identify the level of redundancy present in SLAM datasets, as shown in Figure 4.1, and can utilize the characterization results for optimizing the evaluation process of SLAM.

The significance of the characterization metrics stems from the observed correlation between the characterization results and the corresponding *Absolute Trajectory Error (ATE)*. In Figure 4.2, the non-monotonic correlation coefficient (Kendall-Tau) is measured between different characterization metrics of datasets and ORB-SLAM3 [25] ATE. We can observe medium to high correlation between those characteristics and the SLAM ATE, which suggests SLAM sensitivity to the metrics measured. Thus, coverage of such metrics must be considered in the evaluation process of SLAM.

In this chapter, we utilize the characterization results in [5] to measure the coverage of dynamic range achieved by a combination of several datasets. Based on the achieved coverage, and the characterization of each measurement sequence, an optimal subset of sequences for a set of characterization metrics is calculated. This subset ensures achieving the same level of coverage while minimizing either the number of testing sequences or the number of measurements processed.

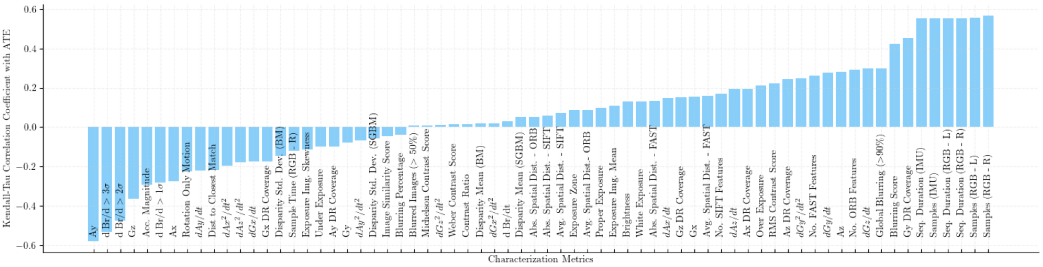


Figure 4.2: Non-monotonic correlation coefficient (Kendall-Tau) between different characterization metrics of datasets and their corresponding ORB-SLAM3 absolute trajectory error(ATE)

4.2 Problem Formulation

The existence of redundancy within and between datasets leads to inefficient design of experimental setup, where evaluation objectives are not efficiently addressed. This gives rise to the need to select a subset of evaluation sequences from available datasets that reaches similar coverage levels for certain quantifiable objectives. For instance, we seek to answer whether the full KITTI dataset is needed to test for illumination changes or not. If not, which sequences in KITTI dataset are representing of the coverage level achieved by the full dataset. In this section, we formally describe the problem we are solving. Also, we define key parameters needed in the course of this work.

Following the definition of the characterization process of a dataset in Section 3.2, the characterization vector q_i (as introduced in Equation 3.3) can be considered a continuous random variable that is represented by the inclusive bounded interval of minimum and maximum characterization value, as such:

$$\delta q_i = [\min_j q_{i,j}, \max_j q_{i,j}] \tag{4.1}$$

Thus, the dynamic range coverage for a a given dataset of sequences Q under a specific characterization metric is denoted by the interval ΔQ, and is given by:

$$\Delta Q = (\bigcup_{i=1}^{n} \delta q_i) \tag{4.2}$$

which is a bounded inclusive interval that is given by:

$$[\Delta Q_{min}, \Delta Q_{max}] = [\min_{i,j} q_{i,j}, \max_{i,j} q_{i,j}] \tag{4.3}$$

where j is an index of the minimum or maximum element in the joint set ΔQ.

Therefore, the outer measure [154] of the interval ΔQ is given by:

$$|\Delta Q| = |\Delta Q_{max} - \Delta Q_{min}| - \varepsilon \tag{4.4}$$

where ε is the numerical discontinuity in the ΔQ.

We seek to find a subset of sequences $\tilde{Q} \subseteq Q$ of size $\tilde{n}$ such that:

$$\tilde{n} \leq n \;\; \& \;\; \Delta\tilde{Q} = \Delta Q \tag{4.5}$$

To find the minimum set $\tilde{Q}$, we define two quantities to compare subsets. The first one is the cost of dynamic range coverage $C(\tilde{Q})$ defined by the number of processed measurements per unit dynamic range coverage:

$$C(\tilde{Q}) = (\sum_{i=1}^{\tilde{n}} m_i) \;/\; |\Delta\tilde{Q}| \tag{4.6}$$

The second is the dynamic range coverage percentage $P(\tilde{Q})$, which is the dynamic range coverage of subset $\tilde{Q}$ relative to the full set of sequences Q, which is given by:

$$P(\tilde{Q}) = \frac{|\Delta\tilde{Q}|}{|\Delta Q|} \; \% \tag{4.7}$$

Finally, we define our problem as one of finding the subset $\tilde{Q}$ that achieves the least number of SLAM evaluation sequences (LS) by solving:

$$\begin{aligned} &\arg\min_{\tilde{n}} \quad \tilde{n} = |\tilde{Q}| \\ s.t. \quad &\tilde{n} \leq n, \quad \tilde{Q} \subseteq Q, \quad \Delta\tilde{Q} = \Delta Q \end{aligned} \tag{4.8}$$

or the lowest possible cost (LC) of dynamic range coverage by solving:

$$\begin{aligned} &\arg\min_{\tilde{Q}} \quad C(\tilde{Q}) = (\sum_{i=1}^{\tilde{n}} m_i) \;/\; |\Delta\tilde{Q}| \\ s.t. \quad &\tilde{Q} \subseteq Q, \quad \Delta\tilde{Q} = \Delta Q \end{aligned} \tag{4.9}$$

The two objectives aspire reducing the footprint of SLAM evaluation while maintaining the same dynamic range coverage achieved by a pool of evaluation sequences.

To illustrate the relation between characterization metrics, and the aforementioned definition, we discuss the following example. Assume we want to

evaluate a SLAM system against illumination changes (the objective characterization metric) on the KITTI dataset, denoted as Q, that consists of 22 sequences ($n = 22$). After applying the illumination change characterization metric $f(.)$ on each sequence Q_i, we obtain a characterization vector of size m_i denoted as q_i. We calculate the dynamic range coverage ΔQ, and the cost of coverage $C(\tilde{Q})$ of processing all sequences available in KITTI. We seek to find a minimum set $\tilde{Q}$ with $\tilde{n}$ sequences. This subset $\tilde{Q}$ must achieve a dynamic range coverage for illumination changes that is equivalent to that of the full KITTI dataset by either reducing the number of processed sequences ($\tilde{n} \leq n$) or reducing the cost of dynamic range coverage ($C(\tilde{Q}) \leq C(Q)$).

4.3 Proposed Method

Given a set of dataset sequences, we seek to find an optimal minimal subset of sequences that achieves a defined evaluation objective, which is achieving a dynamic range coverage that is equivalent to the full input dataset over one of characterization metrics. As shown in Figure 4.3, the process starts by characterizing all sequences in the dataset w.r.t. the selected characterization metric using the method proposed in [5]. This step produces a characterization vector and a dynamic range coverage for each data sequence. After that, this information is sent to an optimization algorithm which iteratively selects an optimal subset of sequences satisfying the evaluation criteria. At each iteration, the algorithm computes the dynamic range coverage of a candidate and updates its internal state. Once all sequences are considered, the optimal subset is reported.

Computation of the dynamic range coverage is described in Algorithm 1, which depends on integrating sequences boundaries taking into consideration numerical discontinuity regions among intervals that represent sequences in the input dataset. The algorithm yields $\Delta\tilde{Q}$ which is the dynamic range coverage of a given set of sequences $\tilde{Q}$.

On the other hand, the problem of finding the optimal subset of intervals to match a target dynamic range was historically solved using greedy-based

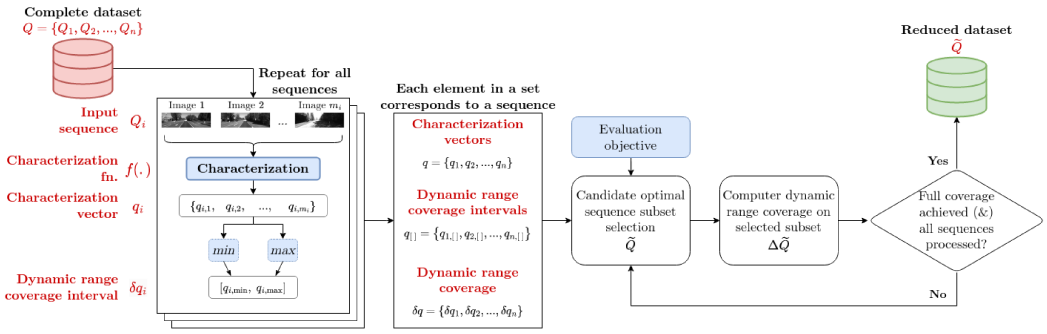


Figure 4.3: A block diagram of the system flow illustrating different system parameters

Algorithm 1 Dynamic Range Coverage Calculation

Input: δq : List of characterization vectors
Output: ΔQ : Dynamic range coverage
Initialization:

$\triangleright\ \varepsilon \leftarrow 0$, total discontinuity
$\triangleright\ \Delta Q_{min} \leftarrow \delta q_{1,min}$
$\triangleright\ \Delta Q_{max} \leftarrow \delta q_{1,max}$

BEGIN: Effective Coverage Calculation

Sort δq list on min. value in characterization vector
for i in $range(1, n)$ **do**
  $\delta = [\Delta Q_{min}, \Delta Q_{max}] \cap [\delta q_{i,min}, \delta q_{i,max}]$
  if $\delta == 0$ **then**
    $\triangleright$ compute discontinuity region ε
    $t_{max} \leftarrow max(\Delta Q_{max}, \delta q_{i,max})$
    $t_{min} \leftarrow min(\Delta Q_{min}, \delta q_{i,min})$
    $\varepsilon\ += (t_{max} - t_{min}) - |\Delta Q| - |\delta q_i|$
  end if
  $\Delta Q_{max} \leftarrow max(\Delta Q_{max}, \delta q_{i,max})$
  $\Delta Q_{min} \leftarrow min(\Delta Q_{min}, \delta q_{i,min})$
end for
Set $\Delta Q = [\Delta Q_{min}, \Delta Q_{max}]$
Set $|\Delta Q| = \Delta Q_{max} - \Delta Q_{min} - \varepsilon$
Return: ΔQ

approaches [144], due to their simplicity, resulting in an acceptable sub-optimal solution in a polynomial time.

Algorithm 2 Greedy Optimization Algorithm

Input: Q : set of sequences to optimize
Input: δq : set of characterization vectors
Output: $\tilde{Q}$: optimal subset of sequences
Initialization:
$\triangleright$ $P(\tilde{Q})$ $\leftarrow 0$, current coverage percentage
$\triangleright$ $\tilde{Q}$ $\leftarrow \{\}$, empty set
BEGIN: Greedy Optimization Approach

1: **Sort** δq list on min. value in characterization vector
2: $i \leftarrow 1$, current sequence to process
3: **while** $P(\tilde{Q}) < 100\%$ **do**
4: **if** $\delta q_{i,max} > \Delta\tilde{Q}_{max}$ **then**
5: **Update** $\Delta\tilde{Q}_{max} \leftarrow \delta q_{i,max}$
6: **end if**
7: **if** $\delta q_{i,min} < \Delta\tilde{Q}_{min}$ **then**
8: **Update** $\Delta\tilde{Q}_{min} \leftarrow \delta q_{i,min}$
9: **end if**
10: **Add** Q_i to $\tilde{Q}$
11: **Update** $P(\tilde{Q}) \leftarrow (|\Delta\tilde{Q}|/|\Delta Q|)\%$
12: **Increment** i
13: **end while**
14: **Return:** $\tilde{Q}$

4.3.1 A Greedy algorithm

Greedy algorithm is a methodology for solving optimization problems that depends on selecting the best available options at the time of decision [37] and is considered the baseline method to which our proposed method is compared. The algorithm does not allow rolling back a taken decision based on observed better alternatives. Thus, it can yield a non-optimal solution upon termination due to its reliance on achieving local-optimality as illustrated in Algorithm 2.

As the problem definition suggests, one can abstract the problem of finding the optimal set of sub-intervals to match the range of a target interval to the famous knapsack problem [157] where the optimal subset of objects are selected to fill a knapsack with defined capacity. Consequently, dynamic programming solutions can be used for the problem to achieve optimal solution with polynomial time [184].

4.3.2 Dynamic programming (DP)

DP is used for solving an optimization problem by dividing it into smaller and easier sub-problems. The final optimal solution is an incremental compilation of the solution of the sub-problems [37]. Moreover, DP has the ability of providing optimal solutions while maintaining reasonable and linear time complexity when the optimization space is discrete, and can provide near-optimal solution when the space is continuous due to the need to perform quantization. Tabulation-based techniques in DP depend on removing redundant calculations of sub-problems [19] ensuring the calculation of any sub-problem once at most as provided in Algorithm 3. The result of any sub-problem is stored in a table-like data structure and is re-used when needed. Similar to any recursive-based solution, a base case has to be defined and is used as a program entry point. In our case, the dynamic range coverage percentage $P(Q)$ is quantized into 10 regions where each represents 10% coverage of the range. In addition to that, an initial empty state representing 0% coverage has to be defined as a base case. Thus, the DP algorithm defines 11 states, where the first is the base state, and the last is required subset equivalent to $P(Q) = 100\%$. In tabulation-based DP, the optimization objective is embedded in the algorithm by defining the replacement function which is responsible for replacing the current state of a table cell with another. The replacement function behaviour changes based on the optimization objective. For instance, the dynamic range coverage of the potential state solution is computed and is compared to the current state solution. The one with higher dynamic range coverage is selected. Otherwise, either the one with the least number of sequences or the least coverage cost is selected.

4.3.3 Local optimality vs. sub-optimality

Local optimality in greedy-based algorithms seeks to find a solution given the current state with memoryless strategy. This means that the algorithm cannot undo a non-optimal step and would build upon it given any new information received. This local optimality is different from sub-optimality, where an algo-

Algorithm 3 Dynamic Programming Algorithm

Input: Q : set of sequences to optimize
Input: δq : set of characterization vectors
Input: s : number of states
Output: $\tilde{Q}$: optimal subset of sequences
Initialization:
$\triangleright\ T_{2..s} \leftarrow null$
$\triangleright\ T_1 \leftarrow \{\ \}$
BEGIN: DP Optimization Approach

for i in $range(1, s)$ **do**
  for j in n **do**
    $L \leftarrow \{\ \}$, temp list to current subset candidate
    if T_i *not null* **then**
      $L \leftarrow T_i \cup q_j$
      $P(L) \leftarrow |\Delta(L)|/|\Delta Q|\%$
      $loc \leftarrow \lfloor P(L)|/10 \rfloor + 1$
      if $loc < 11$ **then**
        if T_{loc} *not null* **then**
          $T_{loc} \leftarrow L$
        else
          if $|\Delta(T_{loc})| < |\Delta(L)|$ **then**
            $T_{loc} \leftarrow L$
          end if
          if $|\Delta(T_{loc})| = |\Delta(L)|$ **then**
            Execute: *Replacement Fn.*
          end if
        end if
      end if
    end if
  end for
end for
Set $\tilde{Q} \leftarrow T_s$
Return: $\tilde{Q}$

rithm can undo a step given newly revealed information, and would achieve a near optimal solution. The divergence from optimality in that case is a result of state aggregation or quantization errors.

4.3.4 Multi-objective optimization

Support of multiple objectives is also conducted to allow SLAM researcher to optimize for a number of objectives at once when needed. Formerly, DP

states are defined by the dynamic range coverage percentage $P(\tilde{Q})$. For multiple objectives, DP states are defined by the *average* dynamic range coverage percentage $\overline{P(\tilde{Q})}$, which is given by:

$$\overline{P(\tilde{Q})} = \sum_{i=1}^{k} P(\tilde{Q})_i \tag{4.10}$$

where k represents the number of characterization metrics (i.e. objectives) to optimize for.

In order to enable multiple objective optimization to happen in polynomial time, two extensions to the original DP algorithm are introduced:

1. **Adaptive state quantization:** Since the dynamic range coverage is a continuous random variable, the need to perform quantization is inevitable. This introduces some shortcomings which are improper choice of granularity of quantization levels and can lead to local minima, which will results in the inability of the DP algorithm to achieve a solution. For that reason, we enhanced the traditional DP structure to adaptively selecting the quantization level when a solution cannot be reached as such, the algorithm refines the quantization granularity gradually by increasing the number of states represents the total dynamic coverage range when until a solution is reached.

2. **State aggregation:** Aggregation in dynamic programming is a well-known method used in order to limit the number of states to maintain the polynomial time complexity of the algorithm [14]. With the introduction of multiple objectives optimization, representing all possible combinations of objectives is not feasible specially when dealing with an increasing number of objectives. For that reason, state aggregation is performed as such, the average dynamic range coverage percentage $\overline{P(\tilde{Q})}$ is used to represent the state space on which quantization is performed. Despite the ability of state aggregation to limit the state space in which the DP algorithm is operating, it adds to its sub-optimality due to the treatment of multiple real-world states are a single state, which is considered an approximation.

4.4 Experimental Setup

Three datasets, KITTI [59], EuroC-MAV [22], and TUM-VI [163], along with their characterizations, were used to demonstrate the performance of the proposed method. When combined, they provide a total of 61 sequences. The corresponding average combined dynamic range is used as the target coverage dynamic range (100% coverage). The proposed method is evaluated in two different experimental set-ups that differ in the number of evaluation objectives the algorithm aspires to optimize for.

Single-objective optimization: We start by evaluating the performance of our proposed method given a single objective criterion. Characterization metrics were divided into three groups: general, visual, and inertial characterization metrics. A baseline is defined to be the process of running all available sequences in order till required coverage is achieved. In this case, the ordering of sequences is following the traditional practice of running complete datasets sequentially, and the stopping criteria is achieving dynamic range coverage equivalent to the whole pool of sequences. In order to show the advantage of our proposed method over traditional SLAM evaluation practice, we compare our results to using: a single dataset, the complete pool of datasets (61 sequences), and the baseline.

Multiple-objective optimization: After that, we examine the performance of the proposed method given multiple objective criterion and we combine all characterization metrics in a fourth group, which includes the metrics from the three aforementioned groups of metrics. To examine the performance of our algorithm against multiple objective optimization, we vary the number of objectives to cover ϵ in the range $\epsilon \in [1, k]$, where k is the number of objectives in a given group. For each step ϵ, we apply the proposed method to 100 randomly selected subsets of objectives, each of size ϵ. The process is conducted twice to cover two optimization goals: least number of sequences (LE) and least possible cost (LC). A more challenging baseline was selected for the multiple-objective case. Thus, greedy algorithm is selected as the baseline to which our algorithm is compared to, in order to provide an evidence for the

Table 4.1: Dynamic range coverage analysis of general (G), inertial (I), and visual (V) characterization metrics for optimization objective of least possible sequences (LS) and least possible cost (LC) for single dataset, all datasets, baseline, and our proposed selection method. Results are averaged over all characterization metrics in a given group.

	General Metrics			Inertial Metrics			Visual Metrics		
Method subset $\tilde{Q}$	$C(\tilde{Q})$	$P(\tilde{Q})$	$\tilde{n}$	$C(\tilde{Q})$	$P(\tilde{Q})$	$\tilde{n}$	$C(\tilde{Q})$	$P(\tilde{Q})$	$\tilde{n}$
KITTI	598433	17.24	–	–	–	–	3856.62	58.16	–
EURO-C	6.39242e+10	23.96	–	1119.14	81.94	–	1689.78	64.07	–
TUM-VI	5.78004e+9	84.23	–	19954.86	62.37	$N-$	15864.8	80.33	–
KITTI & EuroC & TUM-VI	6.32624e+9	100.0	61.0	10845.5	100.0	61.0	17969.97	100.0	61.0
Baseline	4.29366e+9	100.0	44.5	2713.21	100.0	18.54	14470.19	100.0	42.14
LS - Greedy (Ours)	4.71574e+8	100.0	3.6	1013.93	100.0	2.58	2171.16	100.0	12.59
LS - DP (Ours)	4.7144e+8	100.0	2.6	446.38	100.0	1.58	516.74	100.0	2.78
LC - Greedy (Ours)	3.52865e+9	100.0	12.3	2063.65	100.0	11.85	4218.88	100.0	27.51
LC - DP (Ours)	4.71439e+8	100.0	2.6	424.53	100.0	1.58	512.78	100.0	2.81

superiority of the proposed algorithm. Time and space complexity are analyzed for the baseline and the proposed algorithm to show the edge of using DP algorithms compared to other available options.

4.5 Results and Discussion

In this section we present the experimental results of the proposed DP algorithm compared to the selected baseline. We start by evaluating the performance in terms of the size of the resulting subset compared to the original subset of SLAM evaluation sequences. Next, we discuss how the SLAM performance outcomes compare when run on the complete pool of sequences vs. the reduced subset of sequences. After that, we discuss a real-world case study where we aspire to optimize the evaluation process of a SLAM algorithm given its design objectives. Next, we discuss the time and memory complexity of our proposed method compared to the baseline and other available methods. Finally, we conclude by illustrating some of the limitations of DP algorithms.

4.5.1 Single-objective subset optimization

In order to show the advantage of our proposed method over traditional SLAM evaluation practice, we compare our results to using: a single dataset, the complete pool of datasets (61 sequences), and the baseline. Table 4.2 shows

Table 4.2: Sample optimal subset given a single metric objective divided into 3 groups: general metrics, inertial metrics, and visual metrics.

	Least sequences (LS)			Least cost (LC)		
Characterization Metric	**subset $\tilde{Q}$**	**# seq. $\tilde{n}$**	**Reduction %**	**subset $\tilde{Q}$**	**# seq. $\tilde{n}$**	**Reduction %**
RGB samples	outdoor6[3]	1	98.3 %	outdoor6[3]	1	98.3 %
IMU sample rate	outdoors5[3], corridor5[3]	2	96.7 %	outdoors5[3], corridor5[3]	2	96.7 %
IMU/RGB ts mismatch	MH_05_difficult[2], magistrale2[3], MH_04_difficult[2]	3	95.08 %	V2_03_difficult[2], magistrale2[3], MH_04_difficult[2]	3	95.08 %
X-axis acceleration	MH_01_easy[2]	1	98.3 %	MH_01_easy[2]	1	98.3 %
X-axis angular velocity	room5[3], room4[3]	2	96.7 %	room5[3], room4[3]	2	96.7 %
Rotation only motion	room6[6]	1	98.3 %	room6[6]	1	98.3 %
Avg. ORB spatial distribution	corridor1[3], 02[1], corridor3[3], slides1[3]	4	93.44 %	corridor1[3], 02[1], corridor3[3], slides2[3]	4	93.44 %
Blur percentage	12[1], slides1[3]	2	96.7 %	14[1], slides2[3]	2	96.7 %
Inter-image brightness change	magistrale3[3]	1	98.3 %	magistrale3[3]	1	98.3 %

[1,2,3] refer to sequences from KITTI, EuroC, and TUM-VI datasets respectively.

a sample of the reduced testing subsets ($\tilde{Q}$) for each of the characterization metric groups (i.e. general (G), inertial (I), and visual (V)). The same process was applied on all available characterization metrics (73 characterized metrics presented in [5]) and the results were aggregated for each group of metrics, which are summarized in Table 4.1.

4.5.2 Multiple-objective subset optimization

A greedy algorithm is defined to be the baseline to which we compare our results in terms of the subset size and cost of the produced evaluation subset. For that to happen, a heuristic must be selected to govern the operation of the greedy approach, which was selected to be the average dynamic range coverage percentage when optimizing for least number of experiments, and the average cost when optimizing for least cost. Figure 4.4 shows the performance of the DP algorithm compared to the greedy approach on the four characterization metric groups while varying the number of objectives to cover for the two aforementioned goals. We can observe the superiority of the dynamic programming algorithm in selecting the optimal subset compared to the greedy approach regardless of the objective group, number of objectives to optimize for, and optimization goal. The superior performance is achieved in polynomial time and is a result of the adaptive setting of quantization states. As mentioned in Sec 4.3, the DP algorithm adjusts the number of states adaptively to a finer granularity when a certain quantization level is not able achieve a solution, and stops when a solution is acquired. Figure 4.5 provides an histogram of different quantization levels and their success rate. As shown, the adaptive

setting of quantization steps is very useful when objectives are originating from different groups, allowing the algorithm to be more general and suitable for multi-sensor SLAM systems.

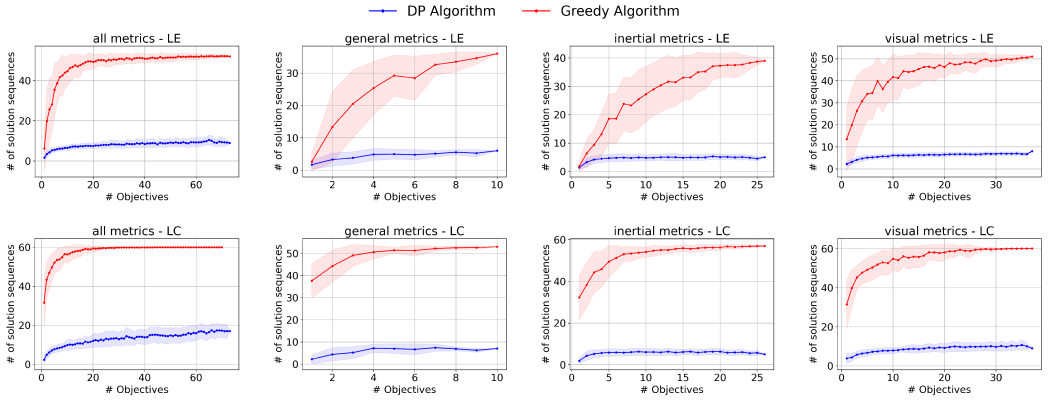


Figure 4.4: Evaluation subsets resulting from both greedy and DP approaches while optimizing for least experiments (first row) and least cost (second row). Each column represents one objectives group (all metrics, general, inertial, and visual respectively). Each figure represents the average evaluation subset size and the standard deviation of running the algorithm on 100 randomly selected set of objectives

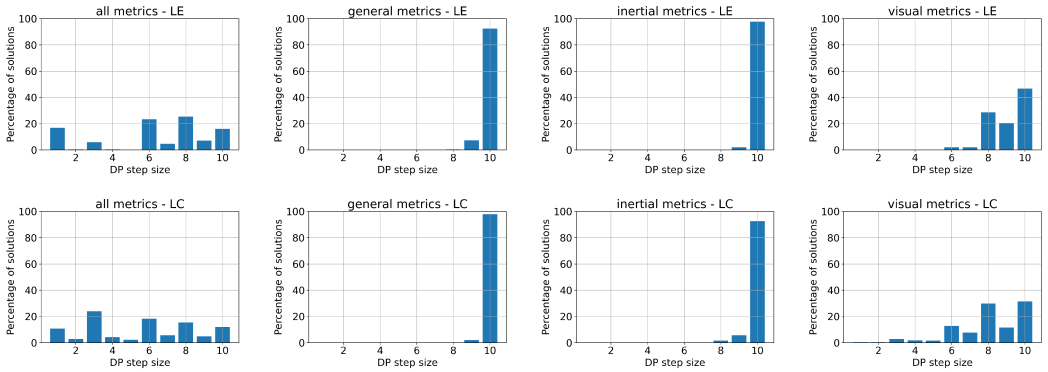


Figure 4.5: Quantization step size and their success rate for each optimization group

4.5.3 SLAM performance under reduced subset

Achieving robustness is tightly coupled with linking SLAM performance to the associated dynamic range [145]. Thus, we evaluated the ATE properties of two SLAM algorithms which are: ORB-SLAM3 [25] and VINS-Mono [146] on the

Table 4.3: ATE Properties with Reduced Evaluation Footprint for All, general (G), inertial (I), and Visual (V) Characterization Metrics

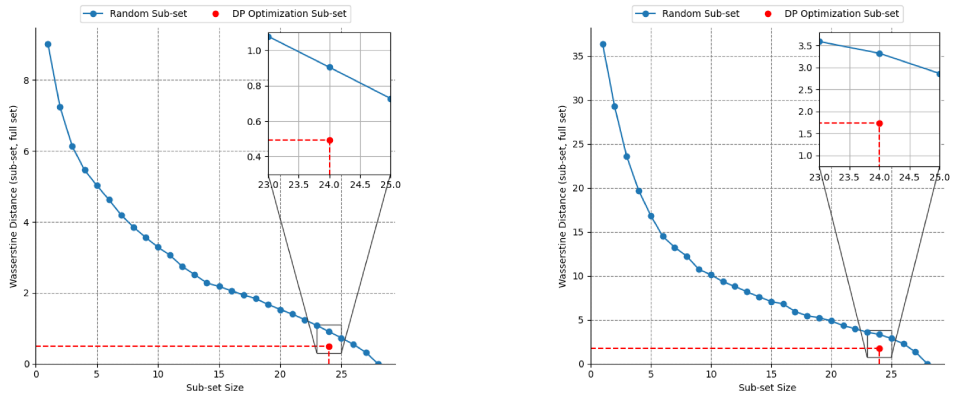
		ORB-SLAM3			VINS-Mono		
Eval. Setup	**# Seq**	μ	**Med**	σ	μ	**Med**	σ
Full TUM-VI	**28**	**5.56**	**0.48**	**11.62**	**23.29**	**0.89**	**43.20**
Reduced TUM-VI (All)	**24**	**5.94**	**0.44**	**12.38**	**23.45**	**0.89**	**44.39**
Reduced TUM-VI (G)	6	2.99	1.17	3.78	44.83	1.94	61.73
Reduced TUM-VI (I)	**19**	**7.44**	**0.47**	**13.52**	**29.43**	**1.56**	**48.14**
Reduced TUM-VI (V)	**24**	**5.94**	**0.44**	**12.38**	**23.45**	**0.89**	**44.39**

Table 4.4: Wasserstine distance of a randomly selected vs. DP-selected sequence sub-set

Eval. Setup	**# Seq**	**ORB-SLAM3**	**VINS-Mono**
Random Sub-set	24	0.9042067	3.2687920
DP-Selected Sub-set	24	0.4923214	1.7422619
DP vs. random selection different		**45.5% (P=0.003)**	**46.7% (P=0.0015)**

complete TUM-VI dataset and the optimized TUM-VI dataset taking into consideration the multi-objective coverage of 73 characterization metrics. The results presented in Table 4.3 show that, the SLAM ATE properties are the same after reducing the evaluation footprint by 15%, and is almost the same after a 30% reduction using only inertial metrics. To measure the significance of the selected subset of sequences in describing the ATE properties of a SLAM algorithm, the DP-optimized subset was compared to the random selection of sequences of the same size. Due to the unbalanced distribution of the ATE values, Wasserstein distance [129] is used to measure the similarity between the probability distribution of the reduced TUM-VI subset and a randomly selected TUM-VI subset. The experiment included increasing the subset size until the full TUM-VI dataset is included, and comparing the 24 sequence cut-off with the DP-based results as shown in Fig. 4.6. The experiment outcomes were averaged over 2000 iterations. Calculating the probability of producing the same DP results from random experiments showed that, DP subset is closer in distance to the original subset with distance less than 45% of its random counterpart. Additionally, the probability of producing the same DP outcomes randomly is very improbable with a probability approaching zero as shown in

Table 4.4. The results hold true for both ORB-SLAM3 and VINS-Mono.



(a) (b)

Figure 4.6: Wasserstine Distance of randomly selected sub-set vs. DP-Selected Subset, compared to the full TUM-VI dataset ATE distribution of (a) ORB-SLAM3 and (b) VINS-Mono

4.5.4 Case study: Illumination changes in direct SLAM

Illumination changes in one of the challenges faced by SLAM. In [131], this problem was addressed in direct SLAM systems, where the evaluation was conducted on a manually selected subset of public datasets. One of those datasets is TUM-RGBD [177] where five sequences were manually selected from the dataset to test for illumination changes. The question now is whether those five sequences are representing all variations in illumination and illumination changes in the dataset or not, and whether there is an optimal reduced subset that we should test against. To answer those questions, we characterized TUM-RGBD dataset using [5], and applied our DP algorithms on the characterization results for both illumination and illumination change to extract the optimal representing subset of sequences to be used for testing. As shown in Table 4.5, only two sequences are needed to cover all illumination conditions, and only one sequence is needed to cover inter-image illumination changes. Additionally, we can see that the resulting subsets are completely different from what was used in [131] to represent TUM-RGBD dataset, which implies how our proposed method can help objective evaluation of SLAM given a defined criterion.

Table 4.5: Comparison between the manual selection and our methdology for selection with defined evaluation criterion.

Selection Method	**Evaluation Subset $\tilde{Q}$**	**# seq. $\tilde{n}$**
Manually, used in [131]	fr1/desk, fr1/desk2, fr1/plant, fr1/room, fr2/desk	5
DP [Illumination]*	fr2/360-kidnap, fr3/nostruct-notext-near-withloop	2
DP [Illumination Changes]*	fr2/360-kidnap	1

*Using the method introduced in this work
[...] indicates the selection objective

4.5.5 Time and memory complexity analysis

One of the main advantages of dynamic programming approaches is the ability to reach an optimal solution in polynomial time compared to brute force solutions. However, this comes with the cost of using more memory. To explain the trade-off, we compare the time and memory complexity of our proposed DP approach with the greedy approach (baseline), exhausted search approach (brute force), and random search approach. As shown in Table 4.6, one can observe the ability of the DP algorithm to achieve a near optimal solution in polynomial time while maintaining an acceptable memory complexity. On the other hand, other solutions are not par with the DP solution in either their optimality level (e.g. random search and greedy approaches) or in their time complexity (e.g. brute force exhausted search).

Table 4.6: Time and memory complexity of different approaches compared to their optimality level

Method	**Time**	**Memory**	**Optimality**
Brute force	$O(n * 2^n)$	$O(n)$	Optimal
Random search	N/A*	$O(n)$	Not optimal
Greedy approach	$O(nlog(n)$	$O(n)$	Not optimal
DP algorithm (ours)	$O(n * m)$	$O(m)$	Sub optimal

* time complexity depends on the no. of iterations tried
n : no. of sequences
m : no. of characterization metrics

4.5.6 DP optimization limitations

A number of parameters determine the performance and level of optimality the DP algorithm can achieve. These parameters are the granularity of continuous space quantization, the choice of the state replacement function, and the choice of the state aggregation method. For instance, the definition of the DP states has a huge influence on the performance specially when the space we want to cover is continuous in nature and quantization is required. Therefore, the granularity of quantization has a direct impact on the optimality and time/memory complexity. Moreover, the definition of the replacement function in case of collision also has an impact on the performance as it can lead to local minima. Finally, the choice of the state aggregation method can also add to the algorithm sub-optimality if improper aggregation is conducted. Despite the mentioned shortcomings of the DP algorithm, it can still outperform the greedy approach.

4.6 Summary

The key points of this chapter can be summarized as follows:

- The problem of measuring the dynamic range coverage and objective evaluation of SLAM was discussed, which is a fundamental requirement in robustness and resilience in SLAM.
- An approach for optimizing the selection of the testing set was introduced with two different optimization objectives: minimization of the number of SLAM evaluation sequences and minimization of the total cost of dynamic range coverage, with results for each presented and discussed.
- The results for each characterization category were presented in detail, showing that the DP algorithm provided superior performance in selecting the evaluation dataset compared to the greedy algorithm, while maintaining polynomial time complexity given varying evaluation criteria set size.

- The DP-based approach reduced the mix of sequences needed to achieve the same coverage objectives of the whole evaluation pool, highlighting redundancy in SLAM datasets and providing a systematic approach for designing SLAM experiments. This work shifts the focus of SLAM evaluation from quantity to quality and establishes a framework for the objective evaluation of SLAM, facilitating proper measurement of robustness and resilience in a quantitative manner.

Chapter 5

Ensemble Learning Regression of SLAM Errors

5.1 Introduction and Motivation

Simultaneous localization and mapping (SLAM) is a fundamental building block that gives modern robotic systems the ability to estimate its location while building a map of the navigated environment [12]. Over the last few decades, SLAM research has evolved significantly in terms of architecture, accuracy, requirements, and challenges [24]. One of the major challenges faced by SLAM is the robustness and resilience of the system when deployed in the real world [145]. Robustness of SLAM is the ability of the system to provide acceptable performance when operating under predefined conditions. Resilience is the ability of a system to converge to an acceptable performance when operating outside of the predefined conditions, which implicitly highlights the importance of having internal error prediction and tolerance mechanisms in SLAM to allow for this convergence to happen effectively [5]. For that reason, researchers have directed their attention towards the introduction of integrity indicators of either some blocks in the SLAM pipeline [28], or the final SLAM outcome [103].

Absolute Trajectory Error (ATE) [205] is considered the de-facto metric for measuring the accuracy of localization in SLAM and is used by most state-of-the-art solutions such as ORB-SLAM3 [25], VINS-Mono [146], among many others. *ATE* is defined to be the root mean square (RMS) of the *Absolute Pose*

Error (APE), which is the instantaneous error between corresponding poses in the traversed trajectory. The relation between ATE and APE is given by:

$$APE = ||\hat{X}_i - X_i|| \tag{5.1}$$

$$ATE = \sqrt{(\frac{1}{N}\sum_{n=0}^{N}(APE)^2)} \tag{5.2}$$

where $\hat{X}_i$ and X_i are the estimated and ground truth pose at keyframe i respectively.

Therefore, on-line prediction of SLAM ATE is an integral part of the quest to reach robust and resilient SLAM as it provides SLAM systems with internal indicators of the integrity of their estimates, which can be used to correct estimation errors, govern switching between localization alternatives, and improve robotics safety when deployed.

In this chapter, we propose a novel methodology for predicting the absolute trajectory error (ATE) of a SLAM algorithm using 1-D global pooling of input data characteristics and an ensemble learning-based regression model. This methodology is motivated by the high correlation observed and reported in our previous work [5] between the SLAM performance of multiple algorithms on one side, and the characterization metrics measured on different SLAM datasets on the other side. Since ATE is considered a coarse performance metric, the method was also evaluated for suitability to predict APE as well.

5.2 Problem Formulation

Following the definition of the characterization process of a dataset in Section 3.2, the characterization vector q_i (as introduced in Equation 3.3), we define y_i which is a scalar variable corresponding to the error of the trajectory.

Each characterization matrix $\mathbf{q_i}$ is transformed to a 1-D vector $\mathbf{z_i}$ of size $(m \times 1)$ by applying a 1-D global pooling function $g(.)$ as follows:

$$\mathbf{z_i} = g(\mathbf{x_i}) \tag{5.3}$$

Sequences can differ in size and thus will result in unequal feature vectors in the 2-D space. Therefore, this transformation is required to perform dimensionality reduction. The transformation process results in a unified feature vector with fixed size for all training, test, and evaluation samples. Consequently, the transformed dataset $\mathcal{D}'$ is defined as:

$$\mathcal{D}' = \{(\mathbf{z_i}, y_i), i = 1, ..., N\} \tag{5.4}$$

We seek to learn a model to predict SLAM errors $\hat{y}_i$ given an unseen 1-D vector $\mathbf{z_i}$ from the transformed dataset $\mathcal{D}'$. In this chapter, we aspire to learn two different SLAM errors which are the Absolute Trajectory Error (ATE) and the Absolute Pose Error (APE).

5.3 Proposed Method

We propose a supervised learning pipeline for the prediction of SLAM errors using an ensemble learning regression model. The process starts by characterizing the input data sent to SLAM using the framework we proposed in [5]. Then, we utilize the concept of 1-D global pooling [98] to reduce the dimensionality of the input to a unified vector format. Afterwards, we train a random forest-based regression model to predict the SLAM ATE. Additionally, we utilize the same model to predict SLAM APE as well to provide a fine measure of performance to allow SLAM of self-assessment. Figure 5.1 illus-

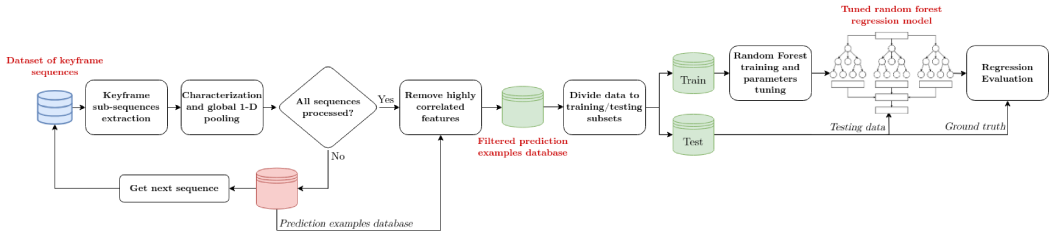


Figure 5.1: A block diagram of the proposed SLAM errors prediction methodology

trates the proposed pipeline and shows how the different system components interact with each other. The figure provides a visual representation of the flow of data and processes in the proposed approach.

5.3.1 Data example generation

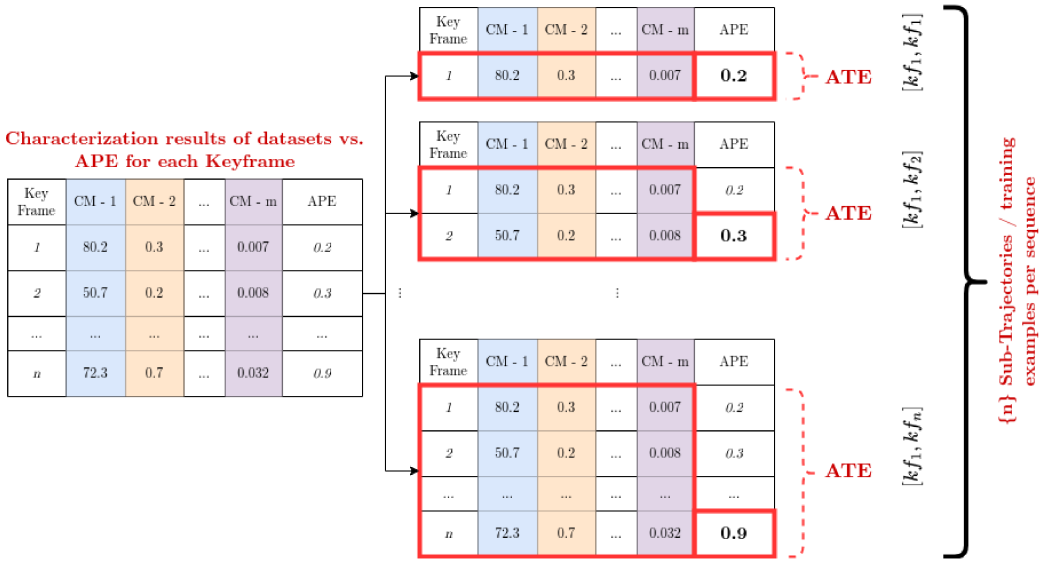


Figure 5.2: Extraction of training examples based on sub-trajectories and corresponding SLAM errors (APE and ATE)

To generate examples for training the error prediction model, we run a SLAM algorithm on all sequences available in several datasets. For each of the run sequences, a number of sub-sequences are calculated that corresponds to the keyframes selected by the SLAM algorithm. Thus, we utilize the concept of sub-trajectories [205] in order to expand the number of training examples.

Given an input sequence of size $\mathcal{K}$ keyframes, we can extract $\mathcal{K}$ examples, where each example is a sub sequence of keyframes (kf) in the inclusive range of $[(kf)_1, (kf)_k]$ where $k = \{1, 2...\mathcal{K}\}$. The corresponding error of each sub trajectory is calculated and is associated with each trajectory to construct a training example. Figure 5.2 illustrate the process in detail and shows how the training examples extraction and error association take place. Sub-trajectories used for model training and testing are generated sequentially from available data sequences in a dataset running in a specific operation mode. The split of the available data for training and testing is done without randomization, meaning that training and testing are conducted on sub-sequences generated from the same dataset but from different sequences.

5.3.2 Sequence characterization and 1-D global pooling

Each generated sub-sequence is considered to be an independent sequence of images/sensor readings. We apply the characterization framework introduced in [5] which contains an array of characterization metrics (e.g. measuring brightness, contrast ... etc.) that generate a characterization vector for each image/sensor reading in the sequence. As seen in Figure 5.3, characterization generates a 2D matrix of size $(m \times n)$, where each row represents a characterization metric outcome, and each column represents an input sub-sequence. Due to the variability in sequence sizes, the generated 2D matrices are not of the same dimension. Thus, to reduce the dimensionality and provide unified feature vectors for training, we apply a 1-D global pooling function on 2-D matrices to generate 1-D vectors of unified size of $(m \times 1)$. This is achieved by reducing each row in the characterization matrix into a single scalar value using the pooling function.

Global 1-D pooling is a technique that was introduced in [98] as a solution to the problem of overfitting in neural networks. The technique does this by reducing the spatial dimension of a feature map to a single value using a global pooling function (e.g., average, min, max, etc.) across all features. Essentially, this reduction replaces a detailed feature map with an abstract, descriptive characteristic of it. Learning those characteristics instead of the examples themselves was proven to enhance the generalization of the learned model [62]. In this work, we examined different statistical 1-D global pooling functions and their impact on prediction quality, which resulted in choosing 1-D global average pooling (GAP) due to its superior performance compared to others.

In this chapter, we utilize one of 12 different pooling functions that include statistical pooling functions (e.g. mean, min, max ... etc.) and diversity pooling functions (e.g. entropy, simpson diversity index and its variants). In order to provide the prediction model with more descriptive features, we also concatenate all 1-D global pooled features into a single feature vector, study its impact on the prediction quality, and compare its performance to using a

single 1-D global pooling function.

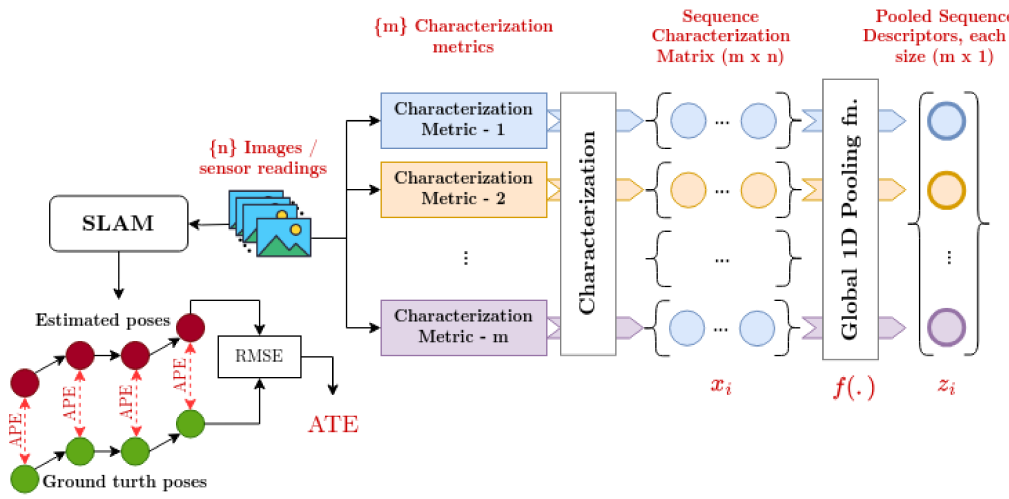


Figure 5.3: Generation of feature vectors using input sub-sequence characterization and 1-D global pooling

Table 5.1: Tuned Hyperparameters in the random forest and their corresponding ranges

Hyperparameter	Range
Number of tree estimators	[10, 1000]
Minimum sample required for a split	{2,5,10}
Minimum samples required at a leaf	{1,2,4}
Maximum number of features used for a split	{None, sqrt, log2}
Maximum depth a tree can grow up to	[10,100]
Bootstrapping for tree building	{True, False}

5.3.3 Removal of highly correlated features

The existence of collinearity between independent input features is a potential problem in regression and can lead to numerically unstable results [41]. To detect highly correlated features, we calculate Pearson correlation coefficient (PMCC) [15] between each feature and all other available features. After that, highly correlated features are grouped where the PMCC between any two features in a group is greater than a threshold of 95%. Then, only one feature is selected from each group which is then used for the training of the

regression model in order to ensure prediction stability of the trained regression model.

5.3.4 Random forest regression model

A random forest regression model is trained and tuned on 70 % of the data examples available for each test case. After that, the model is tested on the remaining unseen 30 % in order to determine its performance. We utilize the random forest regression implementation provided in scikit learn library [133] due to its efficiency and ease-of-use. It also exposes a number of hyperparameters that we can tune for optimal performance of the model.

Tuning the random forest hyperparameters is essential to achieve the best prediction performance. For that, we perform a randomized grid search with cross validation on the multi-dimensional space of hyperparameters provided in Table 5.1. This method is proven efficient in selecting the best hyperparameters while maintaining reasonable complexity and execution time [143].

5.3.5 Performance Evaluation

To quantitatively evaluate the regression quality of our method, four different metrics are utilized, which are defined as follows.

1. Coefficient of determination (R^2)

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=0}^{n}(y_i - \bar{y}_i)^2} \tag{5.5}$$

2. Mean absolute percentage error ($MAPE$)

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{5.6}$$

3. Mean absolute error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5.7}$$

4. Root mean squared errors ($RMSE$)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{5.8}$$

where y is the ground truth, $\hat{y}$ is the predicted value, and n is the number of testing samples.

Those metrics differ in terms of their allowable range, and their indication of the quality of performance. Together, they give a clear indication of the performance and suppress any corner cases or anomalies any metric can suffer from.

5.4 Experimental Setup

In this section, we describe and discuss our experimental setup and associated experimental results. As mentioned in Section 5.3, we run ORB-SLAM3 [25] on three different datasets and in four different modes of operations, resulting in 10 test cases as illustrated in Table 5.2. For each test case, we train and tune the random forest, and evaluate the model performance. Additionally, we study the impact of reducing the amount of training data on the ATE prediction quality to show how our proposed prediction model can still perform relatively well when limited data is available for training.

After that, the same experiments are repeated to predict SLAM APE instead of ATE to evaluate the suitability of the proposed method for the prediction of instantaneous errors of SLAM.

5.5 Results and Discussion

In this section, we show the experimental results of the proposed methodology along with a discussion of the performance of the model, how does it compare to the selected baseline, and the performance of the model on out-of-distribution data.

The experimental results show that the proposed method is able to predict SLAM ATE with a mean accuracy of 93.1 %. On the other hand, the same methodology was able to predict APE with a mean accuracy of 80.45 %, which is a direct indication of the efficacy of our method, the validity of using the characterization metrics as data descriptors, and the proper choice of the 1-D global pooling function for the SLAM ATE/APE prediction task.

Table 5.2: The number of sub-trajectories available from each dataset at different operation modes

		# sub-trajectories available			
Dataset	**# Seq**	**M**	**S**	**M-I**	**S-I**
KITTI	22	11799	23201	-	-
EuroC-MAV	11	3348	1956	3043	1484
TUM-VI	28	2049	1161	4230	1924

* M, S, M-I, and S-I refer to monocular, stereo, monocular-inertial, and stereo-inertial respectively.

5.5.1 Training data generation

To generate examples for training the prediction model, we ran ORB-SLAM3 [25] on all sequences available in three different datasets, which are: KITTI [59], EuroC-MAV [22], and TUM-VI [163]. We apply our proposed data example generation process, which resulted in a great increase in the number of available examples for training, testing, and validation. The method is applied on four different modes of ORB-SLAM3 [25] which are monocular, monocular-inertial, stereo, and stereo-inertial, resulting in 10 different test cases. Table 5.2 shows the number of training examples generated for each of the test cases.

5.5.2 Selection of regression algorithm

In order to validate our selection of the regression algorithm, we examined a number of famous regression models with their default hyperparameter values provided in [133]. These algorithms are: dummy regression that takes the average of input features, linear regression, decision tree, random forest, Ada boosting, and gradient boosting. The evaluation is done using R^2 and $MAPE$ metrics to allow comparison of different test cases as they provide an absolute measure of performance regardless of the value and range of the predicted variable. As shown in Figure 5.4, random forests outperform other regression algorithms resulting in the highest R^2 value and the lowest $MAPE$ value as well. This confirms the argument outlined in [64] where it was found that tree-based solutions can outperform other methods include neural networks when the data is in a structured format. In our case, the characterization

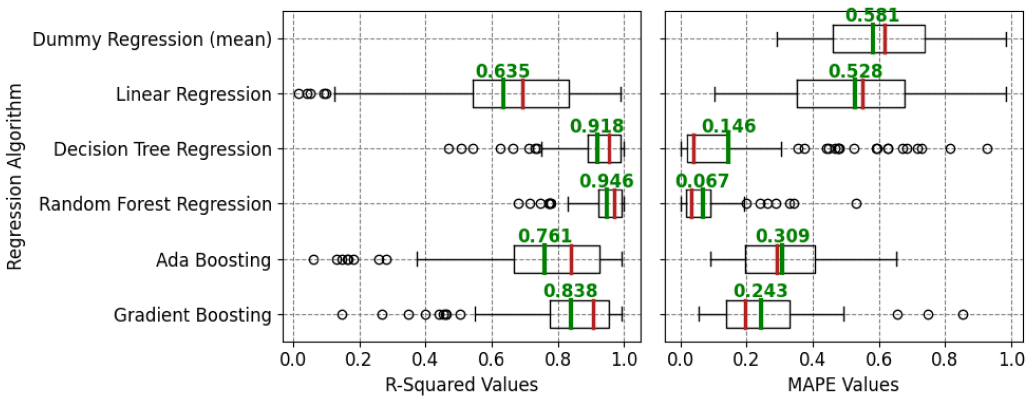


Figure 5.4: Quantitative comparison of different regression Models for ATE prediction

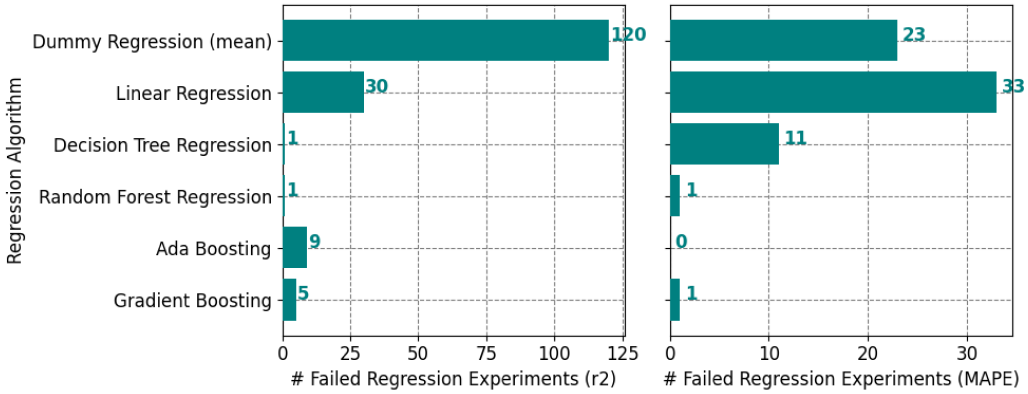


Figure 5.5: Failure rate statistics of different regression models

process transformed the unstructured data (images) into a structured data (characterization vectors). Additionally, we can clearly observe the overfitting problem of boosting algorithms [127] when we examine Ada boosting and gradient boosting performance compared to random forests.

Another experiment included identifying if the algorithm is able to perform regression on a given test case. For that, we utilize the out-of-range concept for both R^2 and $MAPE$ as an indication for failed regression as such: $R^4 \in [0, 1]$ and $MAPE \in [0, 1]$. In Figure 5.5, we can observe that random forests had the least amount of failed regressions when considering both metrics compared to other algorithms. In fact, we observed that the failure case for random forests happen when testing ORB-SLAM3 in stereo mode on TUM-VI due to the lack

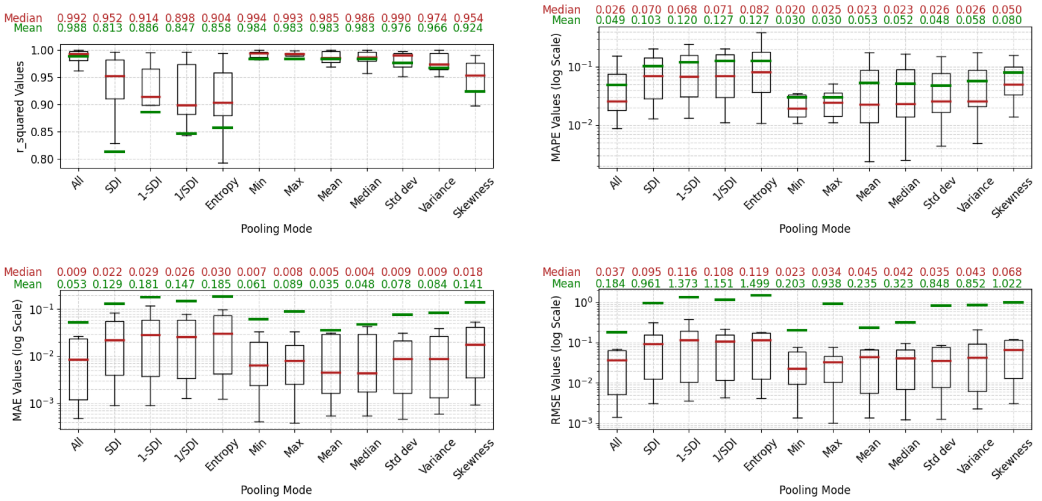


Figure 5.6: Comparison of regression quality for different 1-D global pooling functions after training on 70 % of the data

of enough samples for training which was resolved by further tuning.

5.5.3 Evaluation of 1-D global pooling functions

Another major component of our proposed solution is the 1-D global pooling of training sequences. As described in Section 5.3, we examine 12 different pooling functions where 11 of them are either statistical or diversity indicators, while the last one is the concatenation of the outcomes of all of those functions.

In order to compare the impact of those functions on the regression quality, all test cases are repeated using each pooling function, and performance is evaluated and compared. Figure 5.6 shows the comparison results for each metric and highlights the mean and median performance across all test cases for each pooling function.

It can be observed the superiority of using the concatenated version over using a single pooling function when we look at the R^2 and $MAPE$ values. However, mean and median pooling functions provided a very close performance to the concatenated version, and even reduced the average MAE and $RMSE$ for examined test cases. Thus, we can see the clear overall superiority of using the mean or the median 1-D global function due to the balance it provides in terms of the four measured metrics for all test cases.

Table 5.3: Comparison between using the APE and ATE at 40 % and 20 % of the trajectory respectively as a predictor for the whole trajectory APE and ATE vs. our proposed random forest method when trained on similar available data

	Baseline (APE) at 40%		**Random forest** (APE)		**Baseline** (ATE) at 20%		**Random forest** (ATE)	
Mode - Dataset	R^2	$MAPE$	R^2	$MAPE$	R^2	$MAPE$	R^2	$MAPE$
M-EuroC	-12.9044	1.3333	0.9606	0.1061	0.1557	0.3683	0.9973	0.0106
M-KITTI	0.8171	0.5775	0.9976	0.0280	0.7621	0.3710	0.9992	0.0079
M-TUMVI	-1.2853	0.7225	0.6807	0.3281	0.9036	0.2503	0.9663	0.1246
MI-EuroC	0.1706	0.5437	0.8006	0.1686	-0.0637	0.2909	0.9888	0.0199
MI-TUMVI	-1.3701	0.8984	0.7945	0.2999	0.8571	0.2539	0.9835	0.1221
S-EuroC	0.3305	0.8435	0.9896	0.1562	0.9899	0.2699	0.9993	0.0125
S-KITTI	0.5943	0.4515	0.9995	0.0147	0.9559	0.1627	0.9999	0.0013
S-TUMVI	0.0979	0.5595	0.6494	0.3324	0.6202	0.1904	0.8458	0.0359
SI-EuroC	0.2107	1.1129	0.4613	0.2364	0.6864	0.1954	0.9900	0.0189
SI-TUMVI	-5.6590	1.3394	0.6079	0.3760	0.9381	0.1681	0.9723	0.1979
Mean	**-1.8998**	**0.8382**	**0.7942**	**0.2046**	**0.6805**	**0.2521**	**0.9743**	**0.0552**

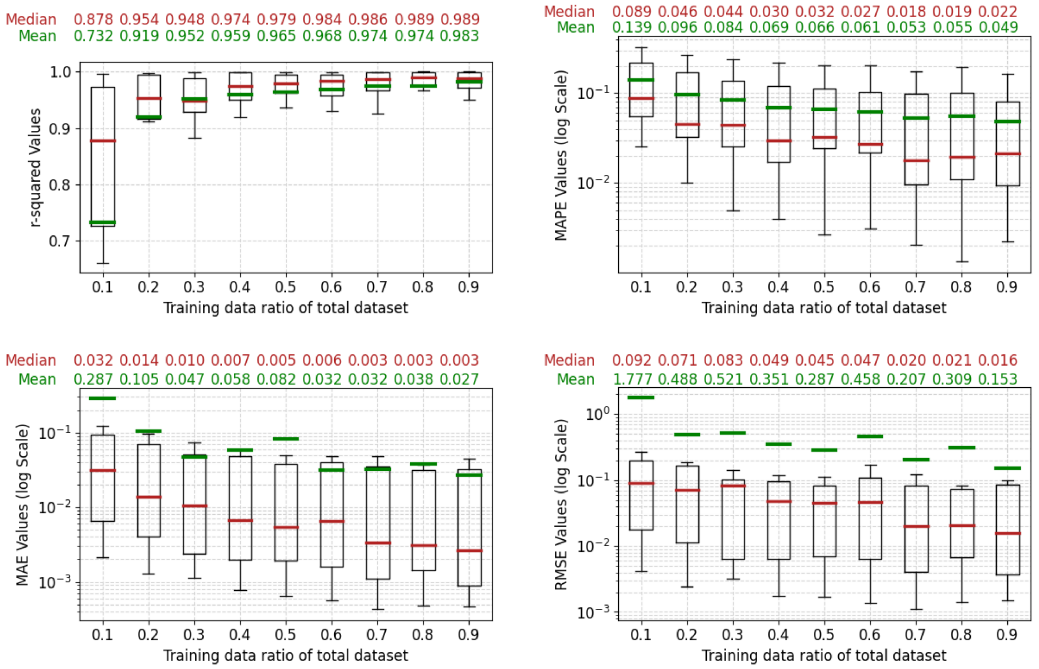


Figure 5.7: Effect of reducing training data size on ATE prediction quality using 1-D GAP

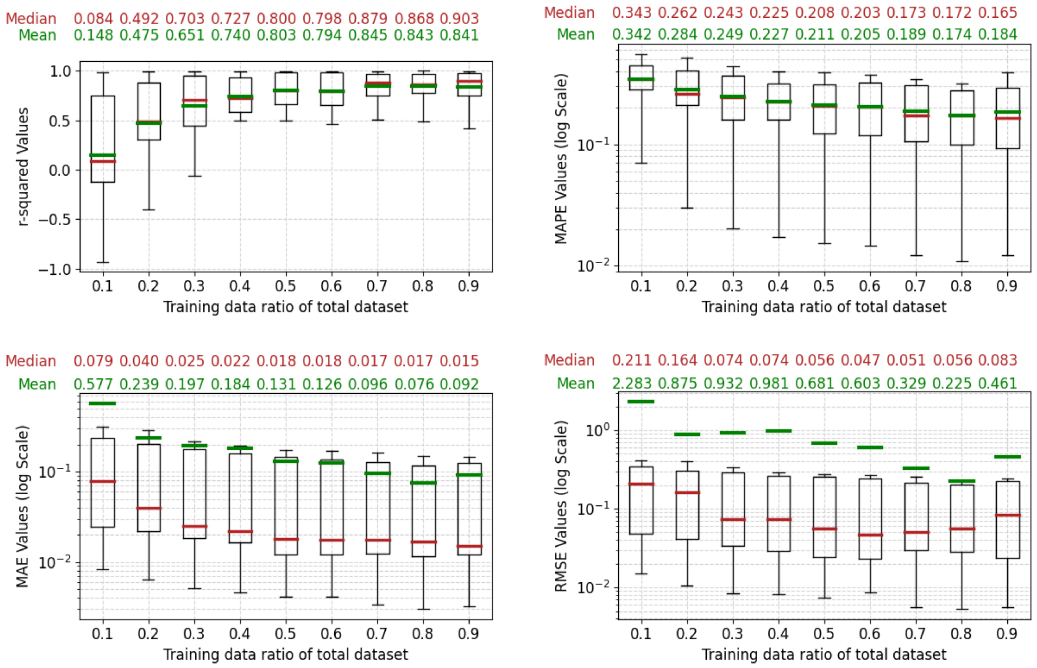


Figure 5.8: Effect of reducing training data size on APE prediction quality using 1-D GAP

5.5.4 Performance comparison to baseline

Due to the nature of SLAM errors and how they evolve over time, we aspire to compare our model when trained on a limited amount of data (20 % of the data in case of ATE and 40 % of the data in case of APE) to corresponding observed errors after traversing the same percentage of a given data example. These experiments highlight the need for training a prediction model to predict SLAM errors since the observed errors during the course of the trajectory are not correlated to that at the end of the trajectory. For that, we compute R^2 and $MAPE$ between the SLAM errors at 20 % for ATE and 40% for APE of the trajectory and ATE/APE at the end of the trajectory. Then, we compare the outcomes with that of our prediction model. As shown in Table 5.3, we can observe that the prediction model outperforms the baseline (ATE at 20 % and APE at 40%) and provides more accurate outcomes.

5.5.5 Impact of limited training data on SLAM error prediction

The supervised learning formulation of the problem requires ground truth data to produce training examples. The availability of such data may be challenging and limited, thus, we examine our method against limited training data in order to provide evidence of its adaptability to such challenging situations.

Not surprisingly, reducing the data available for training will reduce the quality of ATE and APE predictions. However, the question is how much reduction should one expect in case of having limited amount of data available for training. In addition to that, we seek to examine whether the proposed method can be utilized to reduce evaluation efforts of SLAM by training on a small portion of the dataset and the prediction of the rest of the dataset.

As shown in Figure 5.7, we can observe the normal behaviour of increased prediction quality when more training data is utilized. When we look at the R^2 and $MAPE$ metrics, we can observe that we are able to properly predict ATE while training on only 20% of each test case. In that case, the reduction in R^2 is limited to only 6.51% on average. On the other hand, $MAPE$ also dropped by only 4.7%.

On the other hand, in Figure 5.8 we can observe the same increase in predication quality with the usage of more training data. A closer look at R^2 and $MAPE$ metrics show the ability to properly predict APE with only 40% of the data produces a reduction of 10.1% and 4% in R^2 and $MAPE$ respectively.

5.5.6 ATE prediction accuracy

The comparison between the actual ATE and the predicted ATE using 1-D GAP and random forests is presented in Figure 5.9 for the 10 testcases we examined. Moreover, we present the kernel distribution estimate (KDE) of the absolute error percentage of all testing example in each of the 10 testcases in Figure 5.10. One can observe that we are able to predict the ATE value

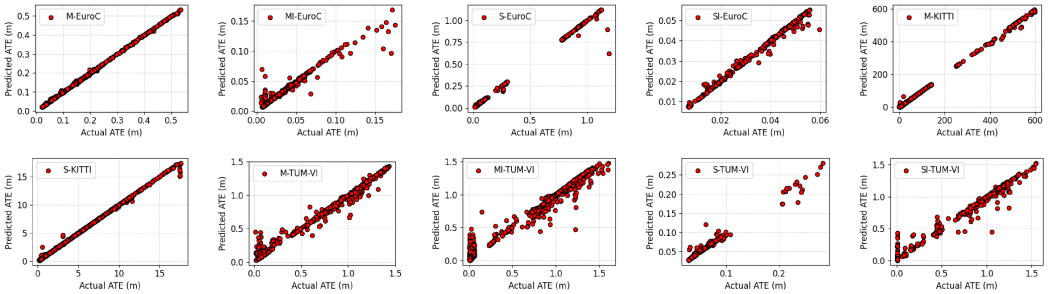


Figure 5.9: Actual vs. predicated ATE for all testcases using the 1-D GAP and random forests after training on 70 % of the data

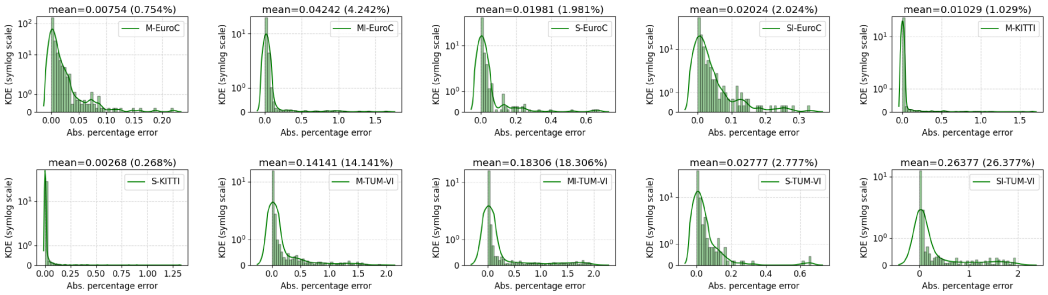


Figure 5.10: Absolute error percentage of all testcases using 1-D GAP and random forests after training on 70 % of the data

within an average error of 7% of the actual ATE with peak performance of an average error that is less than 5% of the actual ATE value in 7 out of 10 testcases examined.

5.5.7 APE prediction accuracy

Additionally, we compare the actual APE and the predicted APE using the same setup in Figure 5.11. A closer look to the KDE of the absolute error percentage provided in Figure 5.12 shows that we are able to predict the APE value within an average error of 19% with peak performance of average error that is less than 14% of the actual APE in 7 out of 10 testcases examined.

Although the method proposed is able to adapt to both ATE and APE, the performance when predicting ATE is much better in terms of accuracy. The reason is that ATE is a smoothed signal compared to APE which observes more changes over time.

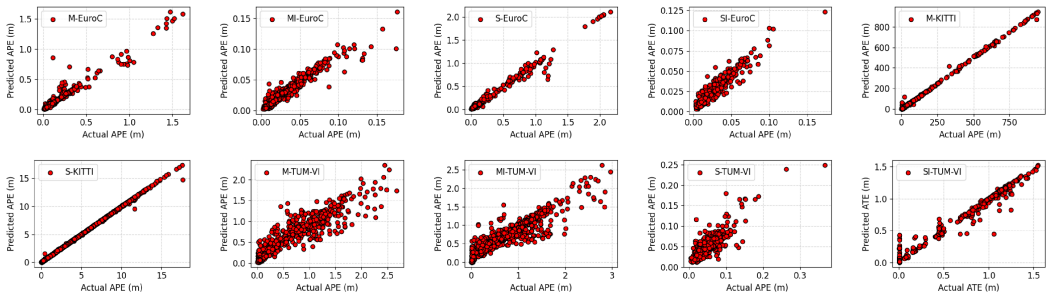


Figure 5.11: Actual vs. predicated APE for all testcases using the 1-D GAP and random forests after training on 70 % of the data

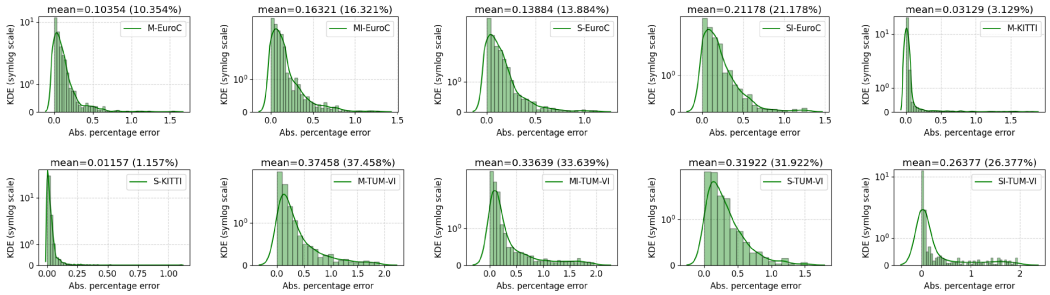


Figure 5.12: Absolute error percentage of all testcases using 1-D GAP and random forests after training on 70 % of the data

5.5.8 Out of distribution (OOD) prediction

Supervised machine learning operates under the closed-world assumption, such that both training and testing assume independent and identical distributions (*i.i.d*) [166]. When input data shifts from this distribution, we witness what is called covariate shift [147]. This causes the predictor to provide non-accurate estimates as the testing data follows a different distributions compared to the training one.

A dataset shift was observed when trying to test the model on data from a testcase that is different from the training one, which resulted in a huge reduction in prediction quality of the model. In order to prove the dataset shift, sinkhorn distance [39] was measured between different testcases and is provided in Figure 5.13. Sinkhorn distance is used to measure the shift in distributions between training and testing data when both are coming from different testcases.

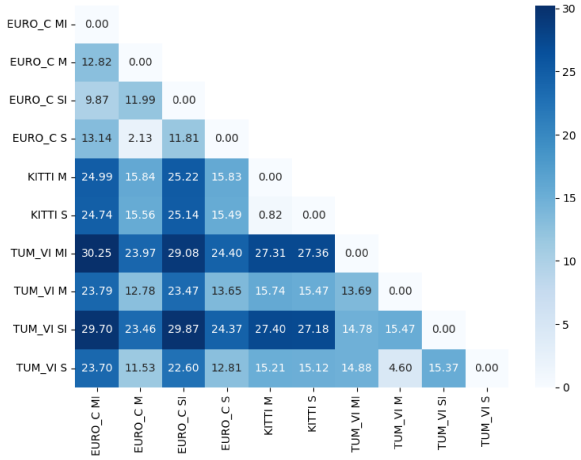


Figure 5.13: Sinkhorn distance between different testcases

5.6 Summary

The key points of this chapter can be summarized as follows:

- The problem of performance prediction in SLAM is addressed as a fundamental requirement for robustness and resilience, starting with a formal mathematical formulation of the problem.
- The methodology for predicting SLAM errors using an ensemble learning technique and 1-D global average pooling of input data characterization results is introduced and validated against various regression models, confirming the selection of random forest regression as the most proper regression model for the task.
- The methodology is tested on 10 different test cases, demonstrating its adaptability to accurately predict SLAM errors when faced with different scenarios from different datasets. Experimental results show the superiority of using random forests over other regression models, achieving prediction accuracy of 93.1% for ATE and 80.45% for APE, and proving that reducing training data to 20% for ATE and 40% for APE maintains proper prediction quality.
- The study also examines the method's suitability for predicting out-of-distribution data, providing evidence of dataset shift between different

test cases.

- The study highlights the method's capability to predict both coarse and fine SLAM error metrics, enabling SLAM algorithms to self-assess and enhance robustness and resilience.

Chapter 6

On-line Fault Detection in VI-SLAM

6.1 Introduction and Motivation

Robustness and resilience of SLAM systems are becoming critical requirements in modern robotic systems due to their impact on our ability to safely deploy robot systems in more diverse and challenging environments [5]. One of the main requirements for having robustness and resilience is the ability of SLAM to self-assess its estimation performance and provide mechanisms for the detection and recovery from different faults.

What is a fault in SLAM? A proper definition of a fault in SLAM is a critical first step that shall precede the proposal of any fault detection or monitoring mechanism. A closer look into the literature reveals that the definition of a fault is divided into two categories. The first is the inability of the system to produce an output with limited attention to the reliability of such output [25], [146]. While the second category defines a fault as the degradation of performance beyond a defined boundary [103]. While the former definition is easier to detect. For instance, RTAB-Map [91] monitors the transformation between current and previous keyframes, and declares a fault when an identity transformation is witnessed over a defined period of time. Nevertheless, this category of fault detection methods was proven ineffective and led to many false negative situations where fault recovery mechanisms were not engaged while needed as shown in Figure 6.1. As shown, the error in ORB-SLAM3

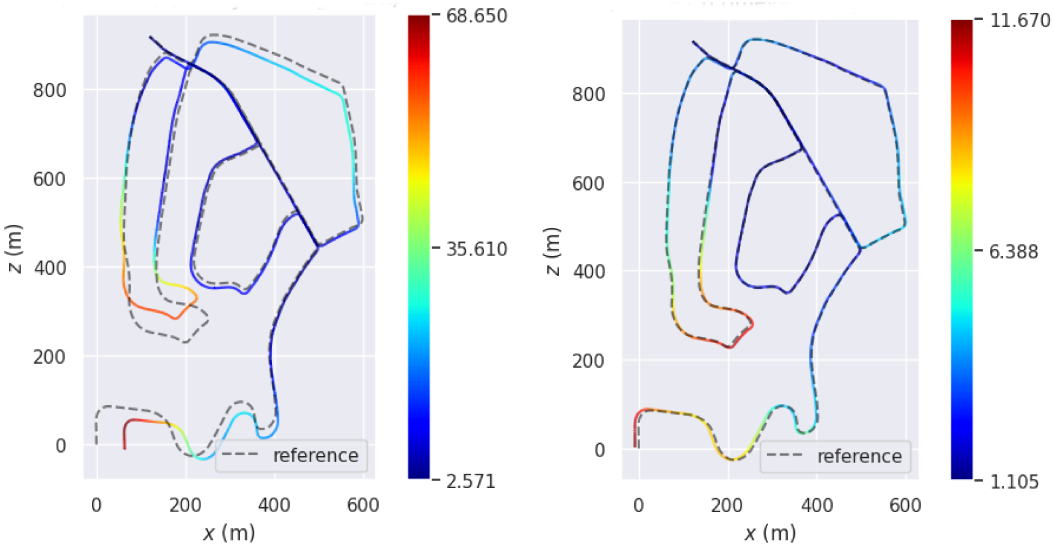


Figure 6.1: An example of ORB-SLAM3 performance on KITTI seq.02 in monocular and stereo modes. Fault detection and recovery mechanisms were not engaged despite the large error observed

output grew to the level of almost 70 meters while the system is not aware of such degradation. Despite having an internal fault recovery mechanism, recovery measures were not engaged due to the inefficacy of the built-in fault detection technique. The latter definition provides a more generalized and practical perspective of the fault in VI-SLAM and ties it to the application in which the performance degradation can be of immense importance. For that reason, we adopt the latter definition in this proposed work.

On the other hand, modern VI-SLAM relies on tightly-coupled integration where a number of front-end modules feed a single back-end module with constraints with the purpose of structuring an over-constrained system to estimate a number of parameters that may include robot poses, landmark locations, and IMU sensor biases [151]. With the lack of fault detection mechanisms, negative feedback loops can occur, and thus corrupt the estimates to unfixable levels. For instance, in ORB-SLAM3 [25], such errors in the estimated accelerometer biases, and due to the tightly-coupled integration, led to a 70-meter error in localization when the system was operating in outdoor settings.

Moreover, Inertial Measurement Units (IMUs) are considered a standard sensor in modern robotics applications due to their cost and size efficiency.

Due to the nature of the measurements of the IMU, deducing a navigational state (e.g., orientation, position, and velocity) can be challenging due to the accumulated error that leads to estimates drift [125]. However, with proper calibration [141] and restriction of the IMU to operate in the short term, one can achieve relatively acceptable estimates that can complement some of the shortcomings of visual systems such as operating in textureless environments or in poor lighting conditions to name a few.

In this chapter, we seek to provide a solution to the problem of performance monitoring in VI-SLAM. The method relies on using a decoupled IMU-based kinematic model in the short term that can act as a reliable supervisory signal for VI-SLAM. With the existence of this supervisory signal, consistency of VI-SLAM outcomes can be evaluated and faults can be detected robustly. To achieve that, we start by providing a modified version of DUET [99] (i.e., *Sequential DUET*), which is a data-driven IMU calibration method responsible for the prediction of the IMU errors. Afterwards, we propose the utilization of trajectory alignment using a buffered history of poses from VI-SLAM and the IMU-based kinematics model to allow the measurement of consistency between the two sources, and thus, the detection of faults if present. This concept is employed in a wide spectrum of application such as multi-agent robotics [100], redundant multi-source localization sensor fusion [189] [178], and fault detection and isolation systems [75]. The proposed method neither relies on the availability of ground truth data nor the building of a model for SLAM, which makes it algorithm-agnostic, and suggests its suitability for integration with any SLAM system retroactively. Additionally, the decoupled nature of the proposed architecture avoids the negative feedback loop caused by the traditional tightly-coupled architectures used in VI-SLAM [25].

6.2 Problem Formulation

Given a set of timestamped VI-SLAM poses $\mathcal{X}_s$ of size N_s, defined as:

$$\mathcal{X}_s = \{(t_{s,i}, x_{s,i}) \mid i = 1, 2, \ldots, N_s\} \tag{6.1}$$

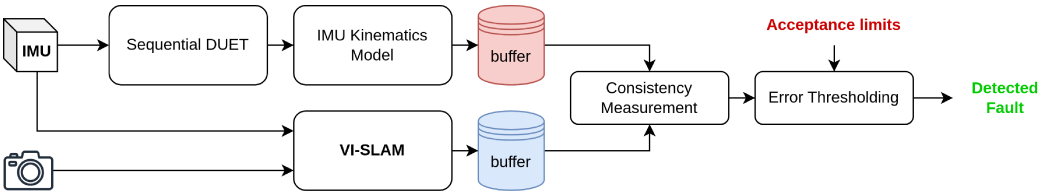


Figure 6.2: A block diagram of the system architecture used for performance monitoring and fault detection

and a set of timestamped IMU kinematics model poses $\mathcal{X}_m$ of size N_m, defined as:

$$\mathcal{X}_m = \{(t_{m,j}, x_{m,j}) \mid j = 1, 2, \ldots, N_m\} \tag{6.2}$$

We seek to find a binary error signal $\hat{\xi}(t)$, which can be given by:

$$\hat{\xi}(t) = \begin{cases} 1 & \text{if } E(t) > \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

Where ϵ is a an error indicator representing the inconsistency between $\mathcal{X}_s$ and $\mathcal{X}_m$.

6.3 Proposed Method

The proposed methodology for performance monitoring relies on the utilization of an IMU-based kinematics model, which can provide reliable odometry information in the short term. This cannot be achieved without having a reliable calibration module that estimates the stochastic errors in both the gyroscope and accelerometer measurements. Afterwards, the stream of poses from VI-SLAM and the IMU-based kinematics model is compared and analyzed for consistency. Finally, given the acceptance criteria, faults can be detected by thresholding the discrepancy between the two sources. An abstract block diagram of the flow is provided in Figure 6.2, while the next few subsections discuss in detail each building block in the proposed system and the different design choices made in each.

6.3.1 Sequential DUET for IMU Noise Calibration

DUET [99] is a deep data-driven model that was recently introduced to perform online calibration of IMU errors using a loss function designed based on

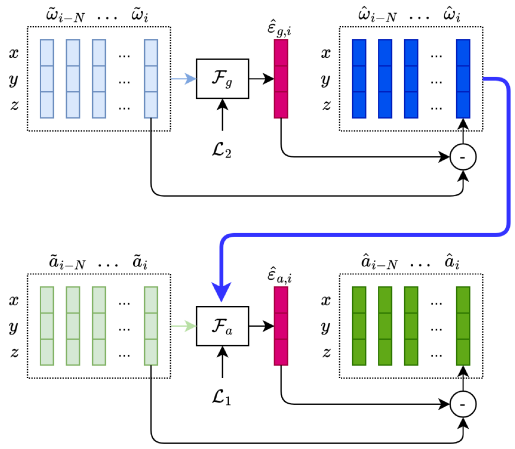


Figure 6.3: The sequential DUET model for online calibration of raw gyroscope and accelerometer measurements

the kinematics motion model of the IMU. The system consists of two connected dilated CNN networks [200] which estimate the gyroscope errors and the accelerometer errors jointly using the loss function that seeks to reduce the error in both orientation and position simultaneously. Practical evaluation showed that simultaneous training of both the gyroscope and accelerometer error networks can result in performance degradation in terms of error estimates. Thus, following the advice of the authors[1], we provide a sequential training strategy *(Sequential DUET)* where the gyroscope error network is trained first in isolation using the raw uncalibrated gyroscope measurements. The resulting corrected gyroscope measurements are fed to the accelerometer error network alongside the raw accelerometer measurements and are trained afterwards.

6.3.1.1 Network Structure

The network structure of *Sequential DUET* is illustrated in Figure 6.3. As shown, we adopt a sequential training strategy where the first dilated CNN $\mathcal{F}_g$ is trained only on $N+1$ raw gyroscope measurements and the ground truth localization information to estimate the gyroscope error $\hat{\varepsilon}_{g,i}$ of sample $\tilde{\omega}_i$ as

[1]https://github.com/kk9six/duet/issues/1

such:

$$\hat{\varepsilon}_{g,i} = \mathcal{F}_g(\tilde{\omega}_{i-N}, \ldots, \tilde{\omega}_i) \tag{6.4}$$

The predicted gyroscope error $\hat{\varepsilon}_{g,i}$ and an optimized multiplier $\hat{\mathbf{C}}_g$ are then used to correct the raw gyroscope measurement as follows:

$$\hat{\omega}_i = \hat{\mathbf{C}}_g(\tilde{\omega}_i - \hat{\varepsilon}_{g,i}) \tag{6.5}$$

Afterwards, the corrected $N+1$ gyroscope measurements $\hat{\omega}$ and raw accelerometer measurements $\tilde{a}$ are fed to the second dilated CNN $\mathcal{F}_a$ to estimate the accelerometer error $\hat{\varepsilon}_{a,i}$ of sample $\tilde{a}$ as follows:

$$\hat{\varepsilon}_{a,i} = \mathcal{F}_a((\hat{\omega}_{i-N}, \tilde{a}_{i-N}), \ldots, (\hat{\omega}_i, \tilde{a}_i)) \tag{6.6}$$

Then, the predicted accelerometer error $\hat{\varepsilon}_{a,i}$ and an optimized multiplier $\hat{\mathbf{C}}_a$ are used to produce the corrected accelerometer measurement $\hat{a}_i$ as follows:

$$\hat{a}_i = \hat{\mathbf{C}}_a(\tilde{a}_i - \hat{\varepsilon}_{a,i}) \tag{6.7}$$

The original loss function $\mathcal{L}$ is disjoint to accommodate the new training strategy as follows.

6.3.1.2 Loss Function

The proposed joint loss function outlined in [99] depends on joint optimization of both the orientation and position error using a weighted average between two loss functions: $\mathcal{L}_1((p, R), \hat{a})$ and $\mathcal{L}_2((R), \hat{\omega})$, where the first aims at optimizing for the position error given the ground truth position p and orientation R and the second aims at optimizing for the orientation error given the ground truth orientation R only. This joint loss function was designed to accommodate the original strategy for simultaneous training of the two CNNs. However, in *Sequential DUET*, we utilize each loss function separately due to our detached structure and sequential training strategy.

6.3.2 IMU Kinematic Model

To determine the pose of the carrier using the calibrated angular velocity vector $\hat{\omega}$ and calibrated acceleration vector $\hat{a}$, we utilize the kinematics model [85], illustrated in Figure 6.4.

Given the initial state of a moving carrier ($R_{(0)}$, $v_{(0)}$, and $p_{(0)}$) and the calibrated IMU measurements represented by the angular velocity vector $\hat{\omega}$ and acceleration vector $\hat{a}$:

$$\hat{\omega} = [\hat{\omega}_x, \hat{\omega}_y, \hat{\omega}_z]^T \tag{6.8}$$

$$\hat{\mathbf{a}} = [\hat{a}_x, \hat{a}_y, \hat{a}_z]^T \tag{6.9}$$

We start by calculating the skew-symmetric matrix of the calibrated angular velocity vector $\hat{\omega}$:

$$\mathbf{\Omega} = \begin{bmatrix} 0 & -\hat{\omega}_z & \hat{\omega}_y \\ \hat{\omega}_z & 0 & -\hat{\omega}_x \\ -\hat{\omega}_y & \hat{\omega}_x & 0 \end{bmatrix} \tag{6.10}$$

The time derivative of the rotation matrix is:

$$\dot{\mathbf{R}}_{(t)} = \mathbf{R}_{(t)} \cdot \mathbf{\Omega}_{(t)} \tag{6.11}$$

Using the Taylor expansion and ignoring higher-order terms, for a small timestep Δt, the rotation matrix at time $t + \Delta t$ is approximated as:

$$\mathbf{R}_{(t+\Delta t)} \approx \mathbf{R}_{(t)}(\mathbf{I} + \mathbf{\Omega}_{(t)}\Delta t) \tag{6.12}$$

To obtain the carrier velocity and position, the acceleration vector $\hat{\mathbf{a}}$ is transformed from the body frame to the inertial frame:

$$\hat{\mathbf{a}}_{inertial} = \mathbf{R} \cdot \hat{\mathbf{a}}_{body} \tag{6.13}$$

and then compensated to remove the gravity component $\mathbf{g} = [0, 0, 9.81]^T$:

$$\hat{\mathbf{a}}_{net} = \hat{\mathbf{a}}_{inertial} - \mathbf{g} \tag{6.14}$$

The carrier velocity is updated as:

$$\mathbf{v}_{(t+\Delta t)} = \mathbf{v}_{(t)} + \hat{\mathbf{a}}_{net,(t)} \cdot \Delta t \tag{6.15}$$

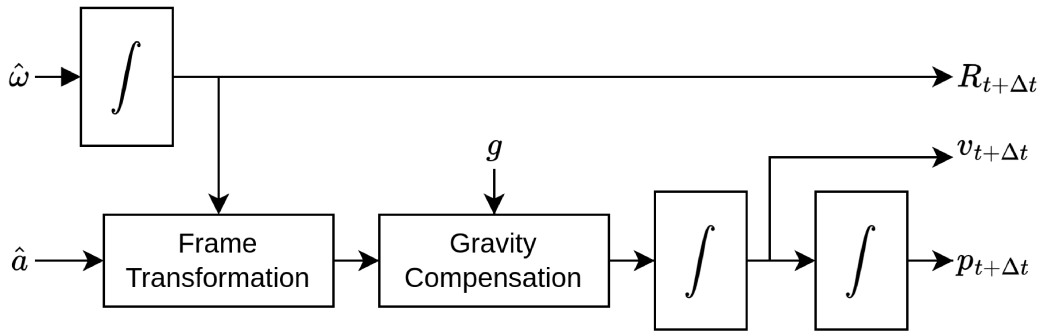


Figure 6.4: An illustration of the IMU kinematics model

Subsequently, the carrier position is obtained using:

$$\mathbf{p}_{(t+\Delta t)} = \mathbf{p}_{(t)} + \mathbf{v}_{(t)} \cdot \Delta t + \frac{1}{2}\hat{\mathbf{a}}_{net,(t)} \cdot (\Delta t)^2 \tag{6.16}$$

By utilizing Equations (6.12), (6.15), and (6.16), we determine the current state of the carrier using the calibrated IMU measurements.

6.3.3 VI-SLAM Pose Buffering

The method relies on monitoring SLAM performance by comparing it to an IMU kinematics model using calibrated measurements in the short term. However, the two sources may operate in different frames of reference, requiring pose alignment for consistency measurement. To achieve this, we adopt a technique used in *Absolute Trajectory Error (ATE)* calculation, specifically trajectory alignment [205]. Therefore, we store the last n poses in the current active map in VI-SLAM in a leaky-memory buffer to form a trajectory for alignment. Similarly, outcomes from the IMU kinematic model are also buffered to form a trajectory. The buffer size for VI-SLAM differs from that of the IMU due to differences in their data frequency. The depth of the VI-SLAM buffer ensures a reliable initial pose in the trajectory, which not only initializes the IMU kinematics model but also plays a crucial role in aligning the two trajectories using Horn's method [72], [174].

6.3.4 Consistency Measurement

To measure the consistency between the buffered VI-SLAM poses and the buffered IMU kinematics model poses, a number of operations need to be

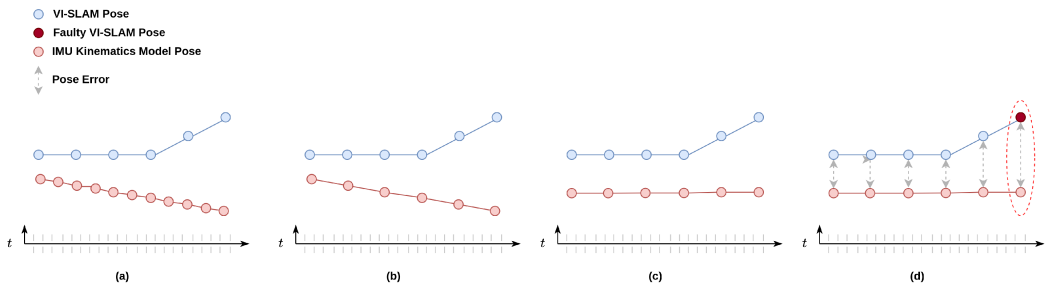


Figure 6.5: The process of consistency measurement for VI-SLAM fault detection. (a) represents the buffered trajectory from VI-SLAM and IMU kinematics model, (b) shows how the re-sampling is done by utilizing linear interpolation, (c) shows the aligned trajectories using Horn's method, and (d) represent the calculation of pose error and detection of fault in VI-SLAM

conducted. We illustrate these processes in detail and provide a graphical representation of the process in Figure 6.5.

6.3.4.1 IMU Poses Re-Sampling

Given a set of timestamped VI-SLAM poses $\mathcal{X}_s$ of size N_s, defined as:

$$\mathcal{X}_s = \{(t_{s,i}, x_{s,i}) \mid i = 1, 2, \ldots, N_s\} \tag{6.17}$$

and a set of timestamped IMU kinematics model poses $\mathcal{X}_m$ of size N_m, defined as:

$$\mathcal{X}_m = \{(t_{m,j}, x_{m,j}) \mid j = 1, 2, \ldots, N_m\} \tag{6.18}$$

We re-sample the IMU kinematics model poses $\mathcal{X}_m$ to match the timestamps of VI-SLAM poses $\mathcal{X}_s$ using linear interpolation [142]. For two bracketing IMU kinematics model poses $x_{m,j}$ and $x_{m,j+1}$, and a target timestamp $t_{s,i}$ where $(t_{m,j} \leq t_{s,i} \leq t_{m,j+1})$, the corresponding IMU kinematics model pose is interpolated as:

$$x_{m,i} = x_{m,j} + \frac{t_{s,i} - t_{m,j}}{t_{m,j+1} - t_{m,j}}(x_{m,j+1} - x_{m,j}) \tag{6.19}$$

This results in a new set of poses after interpolation aligned with the timestamps of the SLAM poses, denoted as:

$$\mathcal{X}_{m,int} = \{(t_{s,i}, x_{m,i}) \mid i = 1, 2, \ldots, N_s\} \tag{6.20}$$

6.3.4.2 Poses Alignment

Using Horn's Method [72], [174], VI-SLAM poses $\mathcal{X}_s$ and interpolated IMU kinematics model poses $\mathcal{X}_{m,int}$ are aligned. VI-SLAM poses are considered the source trajectory and interpolated IMU kinematics model poses the destination trajectory. The transformation from the source trajectory to the destination trajectory is found and applied, aligning the two trajectories for comparison.

6.3.4.3 Pose and Trajectory Errors Calculation

Operating in real-time, we focus on error at the current time step. With a history of poses in the current active map of the last n poses, errors can be calculated both at the pose level and the trajectory level. On the pose level, we utilize the translational and rotational pose error (APE) provide in Equations 2.6 and Equation 2.7 respectively. On the trajectory level, we employ the translational and rotation trajectory (ATE) outline in Equation 2.8 and Equation 2.9 respectively.

6.3.5 VI-SLAM Fault Thresholding

As outlined in Section 6.1, a fault is determined using threshold values $\epsilon_{APE,t}$, $\epsilon_{APE,r}$, $\epsilon_{ATE,t}$, $\epsilon_{ATE,r}$ for acceptable pose error and trajectory error at time step t. A fault $\hat{\xi}$ is detected by thresholding an error indicator $E(t)$:

$$\hat{\xi}(t) = \begin{cases} 1 & \text{if } E(t) > \epsilon \\ 0 & \text{otherwise} \end{cases} \tag{6.21}$$

6.4 Experimental Setup

To evaluate the performance of the proposed methodology, we utilize the EuRoC-MAV dataset [22] and TUM-VI dataset [163] and run their sequences on the state-of-the-art ORB-SLAM3 [25] as our VI-SLAM system in monocular-inertial mode. We start by evaluating the performance of sequential DUET for IMU calibration using the *Relative Translation Error (RTE)*, showcasing the improvement witnessed when applying the kinematics model described in Section 6.3.2 on both raw and calibrated data. Afterwards, we compare the

Table 6.1: Relative translation error (RTE) performance of Sequential DUET on EuroC-MAV and TUM-VI datasets

	RTE (m)			
Sequence	**Raw**	**SeqDUET**	**Improvement**	**Improvement %**
EuRoC-MAV - MH_01_easy	0.915	0.192	0.789	**86.240**
EuRoC-MAV - MH_02_easy	0.882	0.168	0.809	**91.689**
EuRoC-MAV - MH_03_medium	0.715	0.218	0.685	**95.729**
EuRoC-MAV - MH_04_difficult	0.838	0.226	0.732	**87.375**
EuRoC-MAV - MH_05_difficult	0.809	0.181	0.774	**95.581**
EuRoC-MAV - V1_02_medium	0.684	0.334	0.449	**65.723**
EuRoC-MAV - V1_03_difficult	1.087	0.443	0.594	**54.678**
EuRoC-MAV - V2_01_easy	1.009	0.550	0.429	**42.516**
EuRoC-MAV - V2_02_medium	0.678	0.736	-0.184	**27.086**
EuRoC-MAV - V2_03_difficult	0.847	0.685	-0.012	**1.473**
TUMVI - Room 1	0.482	0.422	0.168	**34.737**
TUMVI - Room 2	0.511	0.452	0.158	**30.862**
TUMVI - Room 3	0.574	0.542	0.090	**15.675**
TUMVI - Room 4	0.606	0.599	-0.008	**-1.341**
TUMVI - Room 5	0.537	0.521	0.042	**7.768**
TUMVI - Room 6	0.590	0.629	-0.096	**-16.346**

APE and ATE errors from SLAM and the calibrated IMU kinematics model in the short term, which is the cornerstone for the proposed method. Then, we study the impact of the sub-trajectory size on the performance of fault detection. Finally, we compare our method for fault detection to the methods used in reputable SLAM systems such as ORB-SLAM3 [25] and VINS-Mono [146].

6.5 Results and Discussion

6.5.1 IMU Calibration Using Sequential DUET

To evaluate the performance of sequential DUET with our proposed sequential training strategy, we employ *Relative Translation Error (RTE)* [205], the same performance metric used in DUET [99]. This metric examines the performance on sub-trajectories to explore the impact of trajectory size on calculated error. As shown in Table 6.1, one can observe significant improvement in the performance of the IMU kinematics model when calibration is applied to raw IMU measurements in the EuroC-MAV dataset. However, the improvement is limited when it comes to TUM-VI sequences due to the pre-calibration conducted

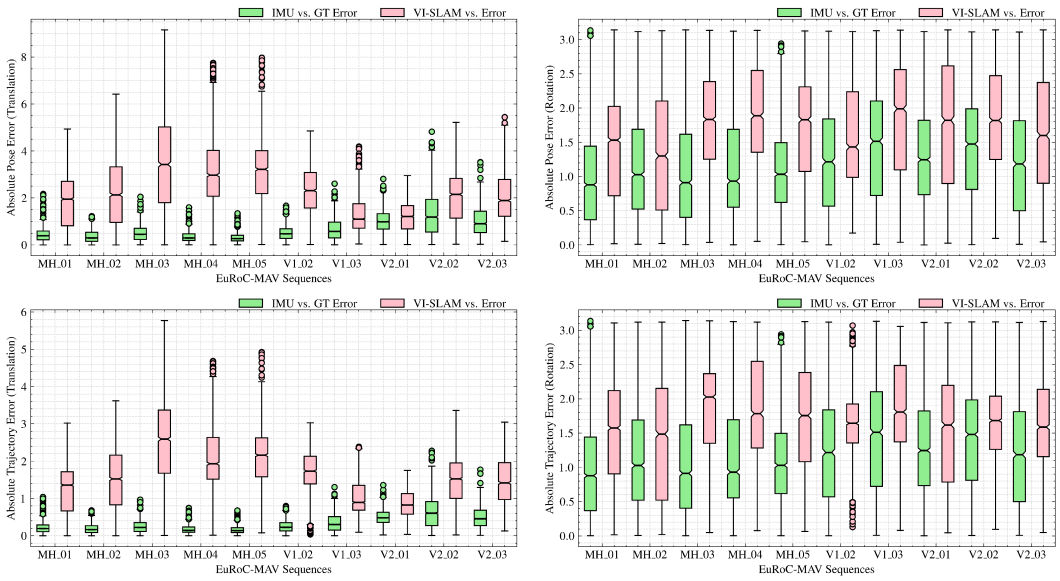


Figure 6.6: Comparison between the errors in poses generated from the IMU-based kinematics model and the VI-SLAM (Monocular-Inertial ORB-SLAM3) for the last 20 poses on EuroC-MAV dataset

on the data by the dataset authors [163].

6.5.2 Performance of Calibrated IMU-based Kinematics Model vs. VI-SLAM

A critical part of the proposed solution is the ability of the IMU-based kinematics model to provide high-accuracy signals for performance monitoring and fault detection in the short term. For this reason, we examined its performance against ground truth within the last 20 keyframes of the current active map. Moreover, we compared its performance to ORB-SLAM3 when running in monocular-inertial mode. As shown in Figure 6.6 and Figure 6.7, we can observe that both APE and ATE errors in translation are very low compared to those in VI-SLAM. However, for rotational errors, there are closer magnitudes between the two sources in the EuroC-MAV dataset.. A closer examination suggests that higher weight should be given to translation errors due to their superior performance in accurately detecting faults.

Another point worth mentioning is the similarity in error patterns when comparing APE and ATE behaviors. This similarity arises due to operating

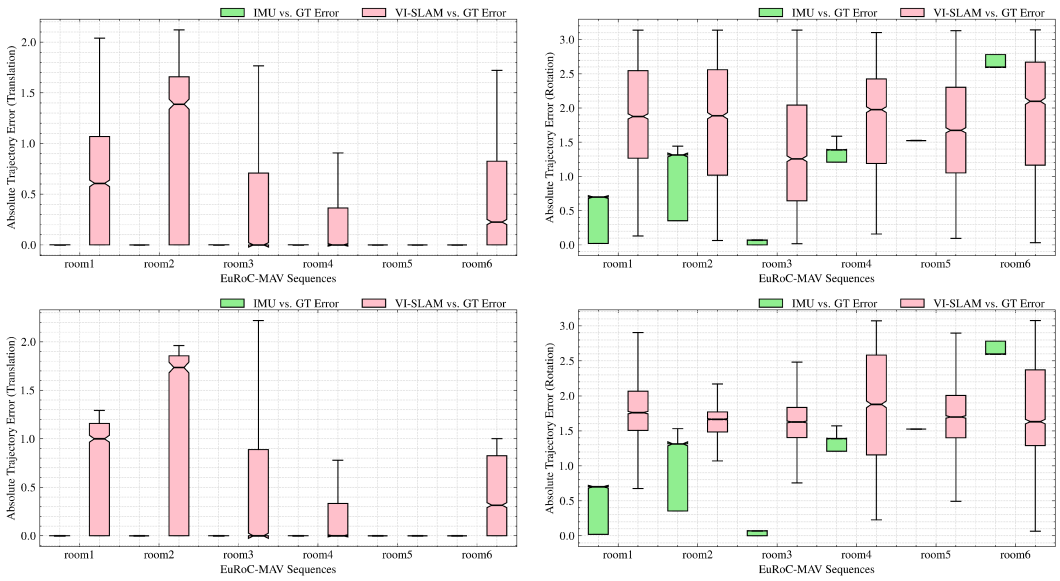


Figure 6.7: Comparison between the errors in poses generated from the IMU-based kinematics model and the VI-SLAM (Monocular-Inertial ORB-SLAM3) for the last 20 poses on TUM-VI dataset

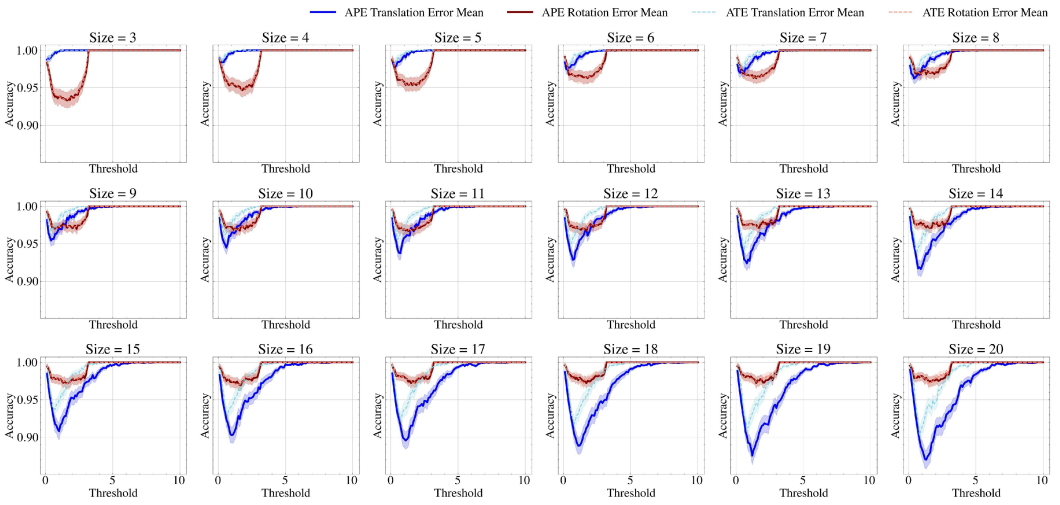


Figure 6.8: Fault detection accuracy and confidence intervals for difference error indicator with varying trajectory size and error thresholds for EuRoC-MAV dataset

within a narrow window of keyframe poses (20 poses), resulting in similar trends with varying error magnitudes.

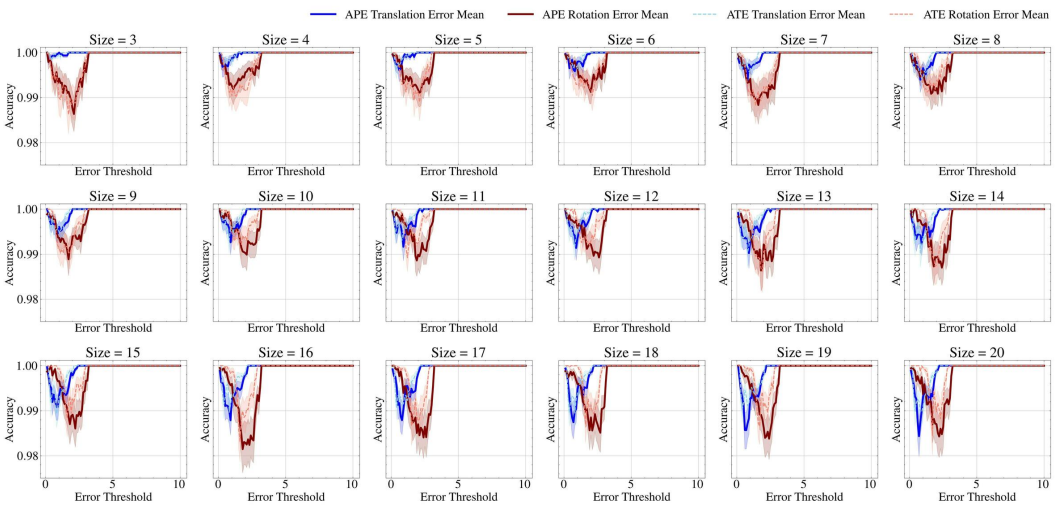
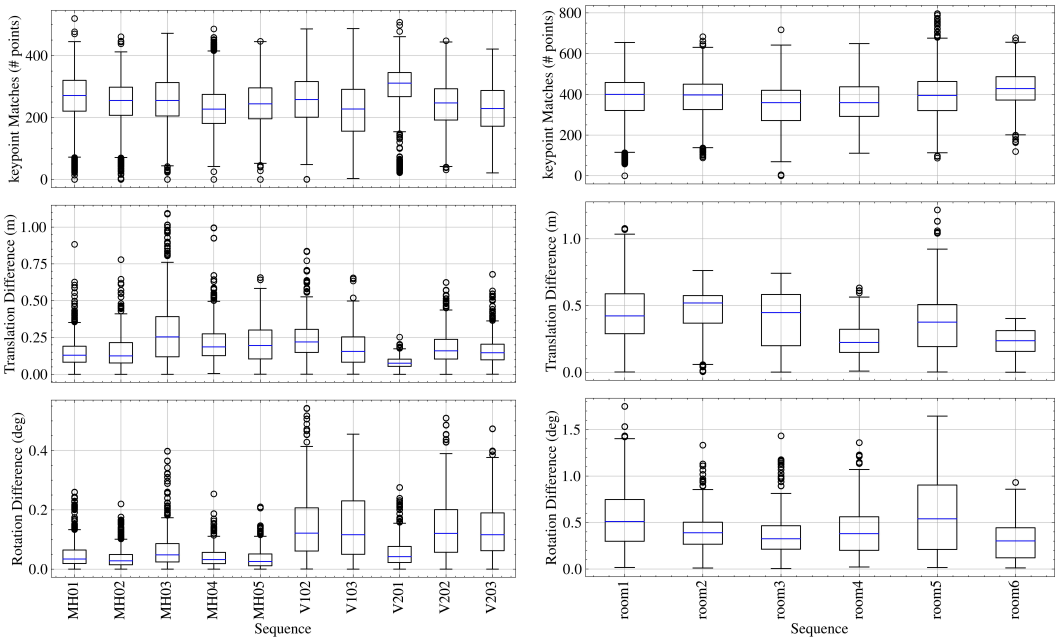


Figure 6.9: Fault detection accuracy and confidence intervals for difference error indicator with varying trajectory size and error thresholds for TUM-VI dataset

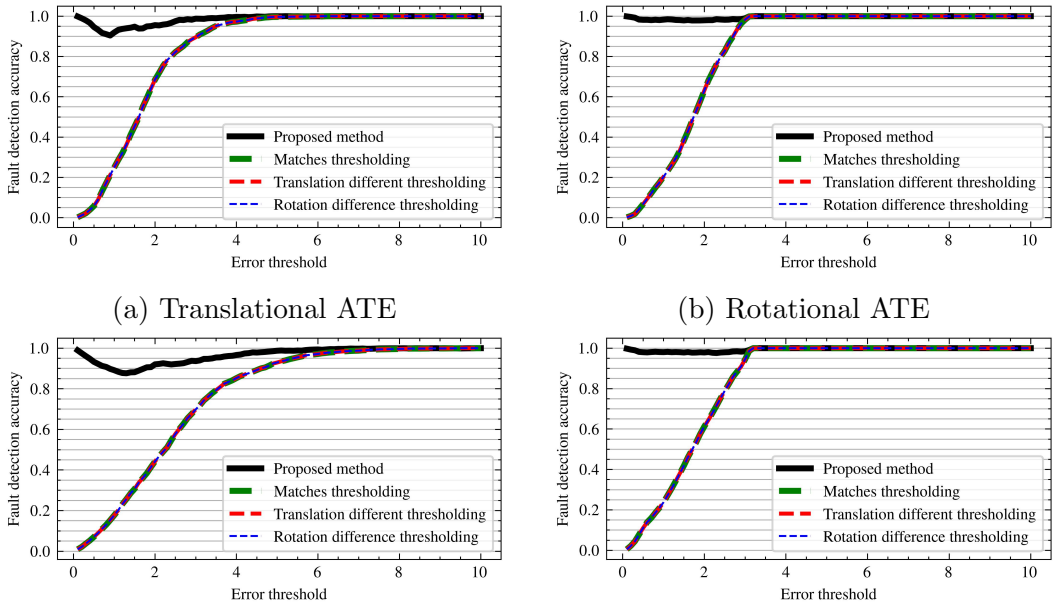
6.5.3 Accuracy and Confidence Intervals

To evaluate the performance of the proposed method, we calculated the accuracy of different fault indicators and their confidence intervals using bootstrapping techniques [46] over 1000 iterations. As shown in Figure 6.8 and Figure 6.9, we observe similarities between ATE and APE indicators due to relying on short-term sub-trajectories of poses. The method shows sensitivity to both trajectory size and fault thresholding, with high accuracy observed when thresholds are low. This is followed by a decrease in accuracy with an increase in the confidence interval range, and then an increase in accuracy with high thresholds. Due to the reliance on thresholding for fault detection, the proposed method tends to treat all samples as faults when thresholds are low. As thresholds increase, the method becomes more conservative, reducing false positives but possibly increasing false negatives, leading to a drop in accuracy. Finally, with high threshold values, the methodology detects faults with high confidence.



(a) EuRoC-MAV (b) TUM-VI

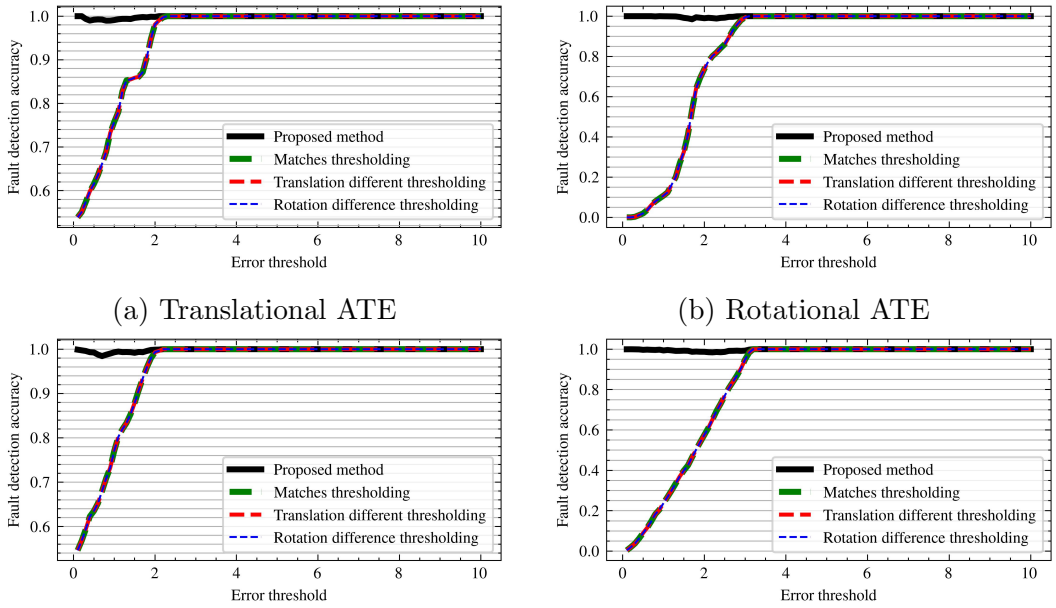
Figure 6.10: Baseline fault indicators used in ORB-SLAM3 (first row) and VINS-Mono (second and third rows)



(a) Translational ATE (b) Rotational ATE

(c) Translational APE (d) Rotational APE

Figure 6.11: Accuracy of different error indicators vs. baseline methods for fault detection on EuRoC-MAV dataset



(a) Translational ATE

(b) Rotational ATE

(c) Translational APE

(d) Rotational APE

Figure 6.12: Accuracy of different error indicators vs. baseline methods for fault detection on TUM-VI dataset

6.5.4 Comparison to Traditional Fault Detection Methods in VI-SLAM

Fault detection is not a standard component in VI-SLAM due to reliance on backend optimization and loop closure constraints for error correction. However, some VI-SLAM systems include fault detection modules to guide SLAM operation and engage fault correction mechanisms, crucial for robust real-time performance when loop closure opportunities are scarce. In this subsection, we evaluate these methods and their ability to detect faults.

ORB-SLAM3 [25] depends on thresholding the number of matched points to declare a visually lost condition when fewer than 15 points are matched in the current frame. VINS-Mono [146] combines this approach with other heuristics, such as detecting significant changes in translation or rotation estimates from SLAM. We evaluate the three heuristics from both VI-SLAM systems in Figure 6.10. As shown, all heuristics fail to indicate a fault and accept performance degradation due to tracking a large number of points or minimal motion produced by VI-SLAM. The observed behavior suggests that

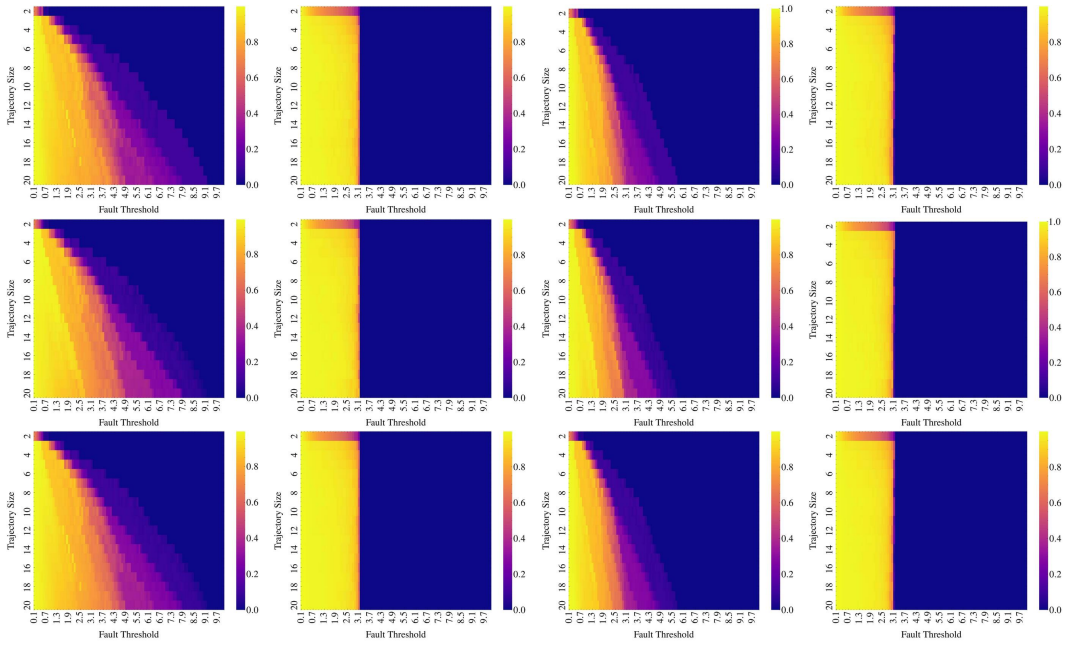


Figure 6.13: Heat maps results on EuRoC-MAV dataset for Precision, Recall, and F1-Score (rows) results of fault detection for different fault detectors APE translation, APE rotation, ATE translation, and ATE rotation respectively (columns)

these heuristics can only detect significant errors that jeopardize system safety in real-world scenarios. In contrast, our methodology can be tuned to detect small errors, providing early indications and real-time monitoring of system performance. The three aforementioned heuristics are then used for fault detection by applying a binary thresholding function. The output is a binary signal that is evaluated against our proposed method. As shown in Figure 6.11 and Figure 6.12, the proposed fault indicators outperform the three methods as such they exhibit consistent accuracy over varying error threshold criterion. The three baseline methods are highly sensitive to the error threshold criteria which suggests their unsuitability for fault detection.

6.5.5 Impact of Trajectory Size and Error Thresholds

Due to the reliance on thresholding in our methodology, fault detection can be viewed as a classification problem with a binary signal compared to ground truth. Therefore, precision, recall, and F1-Score are better performance indicators than accuracy [156] for evaluating the impact of trajectory size and

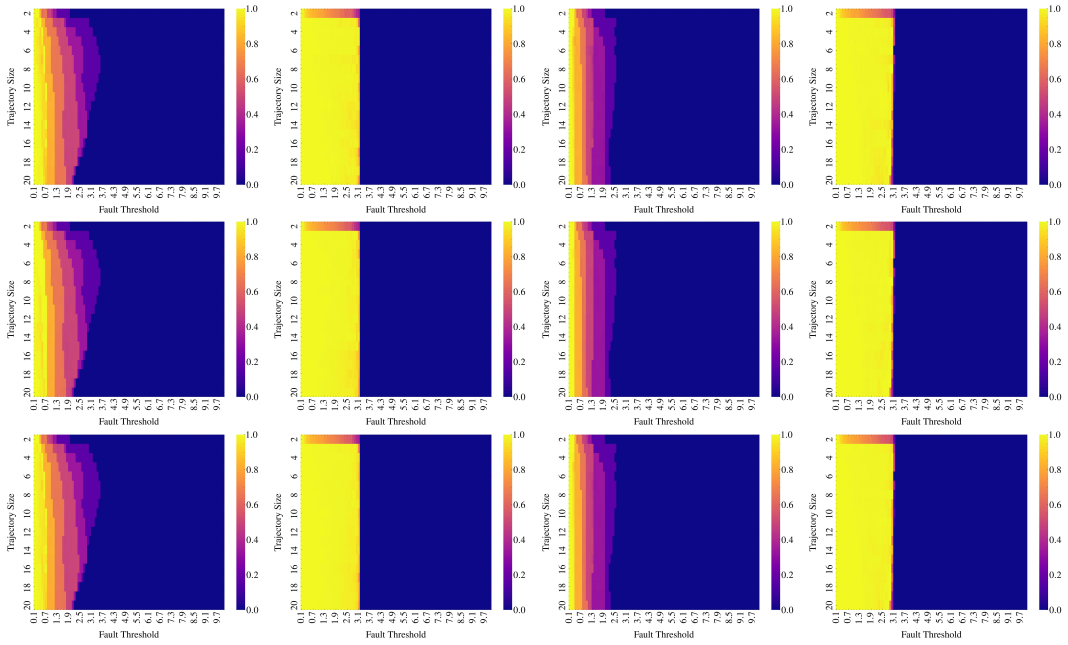


Figure 6.14: Heat maps results on TUM-VI dataset for Precision, Recall, and F1-Score (rows) results of fault detection for different fault detectors APE translation, APE rotation, ATE translation, and ATE rotation respectively (columns)

error thresholds on different fault indicators [63]. Figure 6.13 and Figure 6.14 provide a heatmap illustrating the relationship between trajectory size, error threshold, and performance metrics (precision, recall, and F1-Score), while Figure 6.15 and Figure 6.16 show a 3D surface for better visualization. For translation fault indicators and the same threshold, an increase in true positives and true negatives is observed with increasing trajectory size. Larger trajectory sizes provide more context for alignment in the proposed method, improving initial values for the IMU-based kinematics model. Conversely, no obvious impact of trajectory size is observed for rotational fault indicators. The same behaviour is witnessed for both EuroC-MAV dataset and TUM-VI dataset, which suggests that the proposed method is invariant to the dataset used for evaluation.

Additionally, the sensitivity of the method to error thresholds is evident, explaining changes in accuracy shown in Figure 6.8 and Figure 6.9. Precision and recall heatmaps demonstrate initial false negatives, followed by conservative fault detection, and finally, detection of only large faults with high

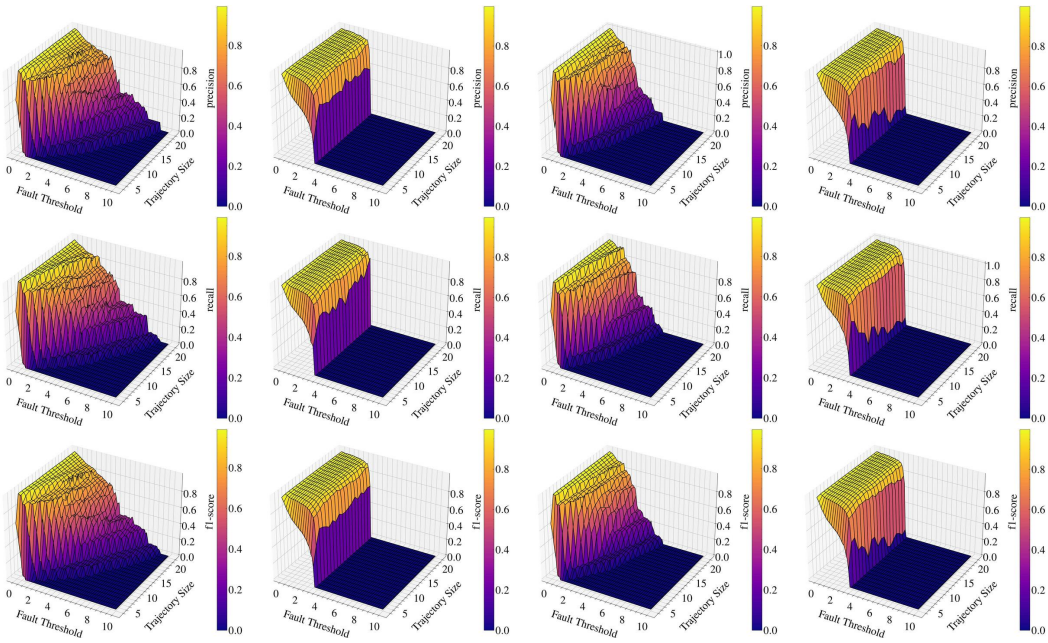


Figure 6.15: EuRoC-MAV dataset results for 3D surfaces of Precision, Recall, and F1-Score (rows) results of fault detection for different fault detectors APE translation, APE rotation, ATE translation, and ATE rotation respectively (columns) showing the relation between each metric and both trajectory size and threshold

confidence.

6.6 Summary

The key points of this chapter can be summarized as follows:

- We address the challenges of performance degradation monitoring and fault detection in VI-SLAM, which is crucial for ensuring robustness and resilience in SLAM systems.
- Our IMU-based methodology for performance monitoring and fault detection is introduced, which relies on a calibrated IMU to generate a supervisory pose stream for comparison with VI-SLAM estimates, employing four fault indicators: translational APE, rotational APE, translational ATE, and rotational ATE.
- Experimental results demonstrate high accuracy in error detection across varying trajectory sizes, with translational indicators showing greater

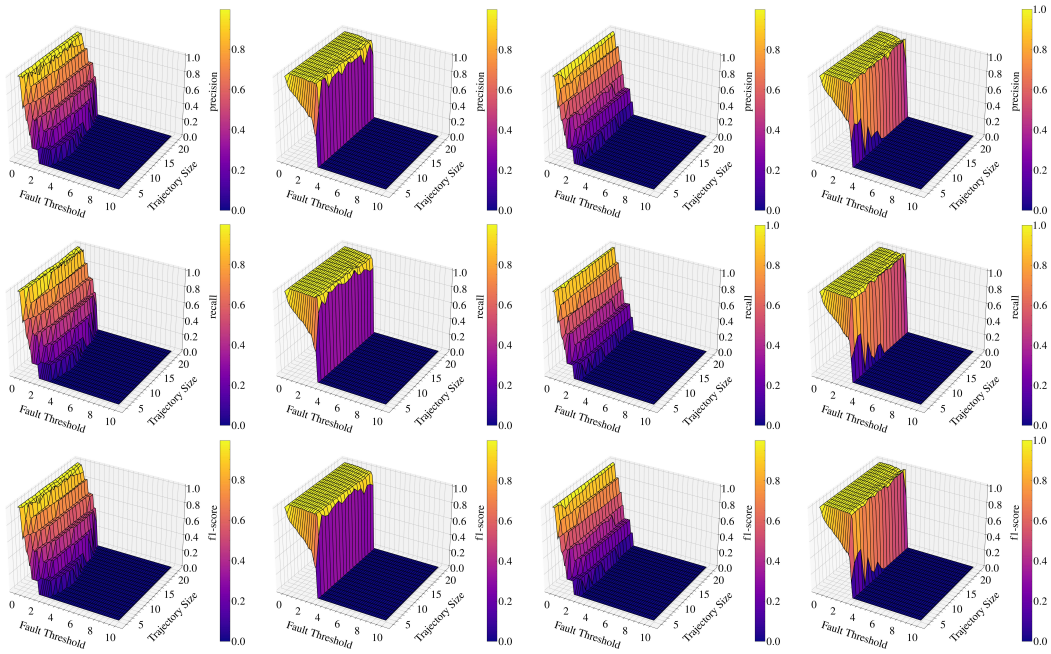


Figure 6.16: TUM-VI dataset results for 3D surfaces of Precision, Recall, and F1-Score (rows) results of fault detection for different fault detectors APE translation, APE rotation, ATE translation, and ATE rotation respectively (columns) showing the relation between each metric and both trajectory size and threshold

reliability due to higher accuracy compared to rotational ones. However, sensitivity to error thresholds with direct binary thresholding is revealed, leading to increased false negatives for minor errors and false positives for major errors.

- A comparison with standard fault detection methods used in VINS-Mono and ORB-SLAM3 underscores the superiority of our approach, proving it to be simple yet effective, and playing a crucial role in achieving robust and resilient SLAM capabilities.

Chapter 7

Conclusion and Future Work

7.1 Conclusions

In this thesis, we systematically addressed the problem of robustness and resilience in SLAM by introducing a formal and rigorous definition for both terms. These definitions allowed us to extract a number of fundamental requirements by which a SLAM system can achieve robustness and resilience. We proposed a set of algorithms and solutions that address each requirement in robustness and resilience in SLAM. Through extensive experimentation, analysis, and comparison to available methodologies in the literature, we have proved that our solutions directly tackle the problem of robustness and resilience and provide an ecosystem where both objectives can be achieved.

- **In Chapter 3**, the problem of quantitative characterization of SLAM datasets is discussed. It has been shown that quantitative characterization of datasets can provide a measurement for the operating ranges and conditions and serve as a key to measuring robustness and resilience. Moreover, such characterization can provide a systematic methodology for designing experiments where a certain environment condition is exploited. Thus, a novel dataset characterization framework has been introduced and described in detail. As an example, the framework is used to characterize three different datasets with the objective of showing its capabilities in terms of measurements and visualization. The characterization process highlights a number of undiscovered anomalies present

in some of the datasets and opens doors to a wide range of studies that can be conducted. The link between the characteristics of the data and the algorithm performance can finally be established and can lead to a systematic evaluation methodology for SLAM system research and development. On the other hand, introduction of new datasets can be guided by the framework outcomes in terms of detecting anomalies by providing a measure for diversity, redundancy, and coverage.

- **In Chapter 4**, the problem of measuring the dynamic range coverage of SLAM was discussed. The study started with a brief review of the topic in closely related disciplines. After that, we introduced an approach for optimizing the selection of the validation set with two different optimization objectives: minimization of the number of validation sequences, and minimization of the total cost of dynamic range coverage. The results of each optimization objective were presented and discussed. After that, the results for each characterization category were presented in detail. It was shown that the DP algorithm was able to provide superior performance in selecting the evaluation dataset compared to the greedy algorithm, while maintaining polynomial time complexity. Utilizing the DP-based approach provided a reduced mix of sequences that achieves the same coverage objectives as the whole evaluation pool of multiple datasets. The reduction achieved highlights the redundancy present in SLAM datasets and provides a systematic approach to design SLAM experiments given defined criteria. This work directs the attention of SLAM evaluation from quantity to quality and provides a framework for objective evaluation of SLAM. This shall open doors to proper measurement of robustness and resilience of SLAM solutions in a quantitative manner.

- **In Chapter 5**, the problem of performance prediction in SLAM is addressed as a fundamental requirement for robustness and resilience in SLAM. The study starts by giving a brief review of the literature related to this topic and provides a basis for the proposed algorithm. After that,

we introduce our methodology for predicting SLAM errors using an ensemble learning technique and 1-D global average pooling of input data characterization results. Our methodology is first compared to a multitude of regression models to validate our selection of random forests as our regression model. Then, the methodology is tested on 10 different test cases to quantify its adaptability to different datasets. The experimental results showed superiority in using random forest compared to our selected baseline and provided evidence for the ability to predict ORB-SLAM3 errors using characterized and pooled features with accuracy that can reach 93.1% and 80.45% for ATE and APE respectively. Additionally, the paper studied the impact of reducing the amount of training data on error prediction quality, and it is shown that we are able to use only 20% for ATE and 40% for APE to maintain proper prediction quality. This is critical due to the limited availability of ground truth data in practical settings. Finally, we study the suitability of the method proposed to predict out-of-distribution data and provide evidence on the dataset shift observed between different test cases examined. The study illustrated the possibility to predict both coarse and fine SLAM error metrics, that can equip SLAM algorithms with the ability to self-assess their estimates and enhance their robustness and resilience capabilities.

- **In Chapter 6**, we address the challenges of performance degradation monitoring and fault detection in VI-SLAM, crucial for ensuring robustness and resilience in SLAM systems. The study begins with a review of prior literature and synthesizes previous approaches proposed to solve these issues. We then introduce our IMU-based methodology for performance monitoring and fault detection, which relies on a calibrated IMU to generate a supervisory pose stream for comparison with VI-SLAM estimates. Our method employs four fault indicators: translational APE, rotational APE, translational ATE, and rotational ATE. Experimental results demonstrate high accuracy in error detection across varying trajectory sizes. While all four indicators detect faults effectively, trans-

lational indicators show greater reliability due to their higher accuracy compared to rotational ones. However, our analysis reveals sensitivity to error thresholds with direct binary thresholding, leading to increased false negatives for minor errors and false positives for major errors. Finally, a comparison with standard fault detection methods used in VINS-Mono and ORB-SLAM3 underscores the superiority of our approach. Our proposed method is proven to be simple yet effective, playing a crucial role in achieving robust and resilient SLAM capabilities.

7.2 Limitations and Future Directions

A wide spectrum of research directions can be explored in order to build upon the contributions of this thesis. We summarize some of them in the following points.

- In Chapter 3, an extendible and generic framework was proposed to address the problem of dataset characterization. The framework provided a very systematic and automatic way to characterize datasets, and also provided a wide spectrum of analysis tools by which we can understand SLAM datasets more deeply. The system was evaluated on three different datasets. However, we think that more datasets should be characterized and aggregations of these characterization results should be done. This allows us to identify the gaps in SLAM dataset literature, provide a method to measure the novelty of any newly introduced work, and also push the limits of SLAM by highlighting their sensitivity to different properties of the environment.

- Objective evaluation of SLAM was discussed in Chapter 4 where a tabulation-based dynamic programming algorithm was proposed to eliminate redundancy in evaluation and provide only relevant data for evaluation. The method assumes that the variability is only in the metrics in the evaluation criteria set of metrics. However, other metrics are also variable and not constant, which may impact the evaluation process. For that

reason, the method proposed can be augmented by taking into account the variability of non-objective metrics as well by considering their correlation to the performance of the SLAM system under test. Another limitation of the system is state aggregation which was critical to ensure a solution in polynomial time. The impact of this aggregation should be studied to quantitatively evaluate how far our sub-optimal solution is from optimality.

- A method for predicting SLAM errors was proposed where an ensemble learning regression model was used to predict both the absolute trajectory error (ATE) and the absolute pose error (APE) in Chapter 5. The method showed high prediction accuracy when the training and testing data relate to the same environment. However, it showed poor quality of predictions when faced with out-of-distribution samples. This is due to the supervised learning formulation of the problem. A number of options are available to overcome this problem. First, this problem can be tackled by exploring semi-supervised and self-supervised techniques where the out-of-distribution prediction accuracy can be enhanced by augmenting training data with different transformations of the available samples. Secondly, the method depends on the utilization of the metrics outlined in the framework presented in Chapter 3, which may not be a sufficient descriptor of the data. Utilization of deep neural networks to provide a descriptor of the data can become handy in this situation and can provide a more generalized descriptor for training data.

- A decoupled method for the detection of performance degradation in VI-SLAM was proposed in Chapter 6. First, the method proposed operates under the assumption of having the IMU-based kinematics model as a high-accuracy signal that is in fact more accurate than VI-SLAM. This may not hold true for future SLAM algorithms where performance is continuously being improved. Thus, automatic validation of this assumption is required prior to the dependency on the methodology as the sole fault indicator. Secondly, The method showed satisfactory results compared

to standard methods, which suggests its suitability for integration with SLAM pipeline where fault correction/tolerance can be triggered based on the outcomes of the proposed fault detection mechanism. Despite the fact that this would require an alteration in the internal architecture of a VI-SLAM algorithm, it can optimize the localization accuracy and optimize the outcomes sent to subsequent tasks as well. Thirdly, the method proposed relies on Horn's method for alignment which depends on the alignment of the translation portion of the robot pose which proven to be sufficient. However, the impact of other techniques for alignment on the performance, such as Umeyama method [205], should be evaluated as well. This allows SLAM researchers to make informed decisions on which technique to choose for their implementation.

References

[1] D. Adolfsson, M. Castellano-Quero, M. Magnusson, A. J. Lilienthal, and H. Andreasson, "Coral: Introspection for robust radar and lidar perception in diverse environments using differential entropy," *Robotics and Autonomous Systems*, vol. 155, p. 104 136, 2022.

[2] D. Adolfsson, M. Karlsson, V. Kubelka, M. Magnusson, and H. Andreasson, "Tbv radar slam–trust but verify loop candidates," *IEEE Robotics and Automation Letters*, 2023.

[3] R. Aldera, D. De Martini, M. Gadd, and P. Newman, "What could go wrong? introspective radar odometry in challenging environments," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 2835–2842.

[4] I. Ali, S. Wan, and H. Zhang, "Prediction of slam ate using an ensemble learning regression model and 1-d global pooling of data characterization," *arXiv preprint arXiv:2303.00616*, 2023.

[5] I. Ali and H. Zhang, "Are we ready for robust and resilient slam? a framework for quantitative characterization of slam datasets," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2022, pp. 2810–2816.

[6] Z. Alsayed, G. Bresson, A. Verroust-Blondet, and F. Nashashibi, "Failure detection for laser-based slam in urban and peri-urban environments," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–7.

[7] M. S. Amin, M. B. I. Reaz, S. S. Nasir, M. A. S. Bhuiyan, M. Ali, A. Mohd, *et al.*, "A novel vehicle stationary detection utilizing map matching and imu sensors," *The Scientific World Journal*, vol. 2014, 2014.

[8] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer, "Fast and incremental method for loop-closure detection using bags of visual words," *IEEE transactions on robotics*, vol. 24, no. 5, pp. 1027–1037, 2008.

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

[10] S. Arshad and G.-W. Kim, "Role of deep learning in loop closure detection for visual and lidar slam: A survey," *Sensors*, vol. 21, no. 4, p. 1243, 2021.

[11] A. Asudeh, Z. Jin, and H. Jagadish, "Assessing and remedying coverage for a given dataset," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, IEEE, 2019, pp. 554–565.

[12] J. Aulinas, Y. Petillot, J. Salvi, and X. Lladó, "The slam problem: A survey," *Artificial Intelligence Research and Development*, pp. 363–371, 2008.

[13] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, Springer, 2006, pp. 404–417.

[14] J. C. Bean, J. R. Birge, and R. L. Smith, "Aggregation in dynamic programming," *Operations Research*, vol. 35, no. 2, pp. 215–220, 1987.

[15] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*, Springer, 2009, pp. 1–4.

[16] S. Bezryadin, P. Bourov, and D. Ilinih, "Brightness calculation in digital image processing," in *International symposium on technologies for digital photo fulfillment*, Society for Imaging Science and Technology, vol. 2007, 2007, pp. 10–15.

[17] S. Bhamidipati and G. X. Gao, "Integrity monitoring of graph-slam using gps and fish-eye camera," *Navigation*, vol. 67, no. 3, pp. 583–600, 2020.

[18] U. I. Bhatti and W. Y. Ochieng, "Failure modes and models for integrated gps/ins systems," *The Journal of Navigation*, vol. 60, no. 2, pp. 327–348, 2007.

[19] R. S. Bird, "Tabulation techniques for recursive programs," *ACM Computing Surveys (CSUR)*, vol. 12, no. 4, pp. 403–417, 1980.

[20] J.-L. Blanco, F.-A. Moreno, and J. Gonzalez, "A collection of outdoor robotic datasets with centimeter-accuracy ground truth," *Autonomous Robots*, vol. 27, no. 4, pp. 327–351, 2009.

[21] J.-L. Blanco-Claraco, F.-A. Moreno-Duenas, and J. González-Jiménez, "The málaga urban dataset: High-rate stereo and lidar in a realistic urban scenario," *The International Journal of Robotics Research*, vol. 33, no. 2, pp. 207–214, 2014.

[22] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[23] Y. Cabon, N. Murray, and M. Humenberger, "Virtual kitti 2," *arXiv preprint arXiv:2001.10773*, 2020.

[24] 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 Transactions on robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[25] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. Montiel, and J. D. Tardos, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[26] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[27] S. Carpenter, B. Walker, J. M. Anderies, and N. Abel, "From metaphor to measurement: Resilience of what to what?" *Ecosystems*, vol. 4, no. 8, pp. 765–781, 2001.

[28] H. Carson, J. J. Ford, and M. Milford, "Predicting to improve: Integrity measures for assessing visual localization performance," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9627–9634, 2022.

[29] D. Caruso, J. Engel, and D. Cremers, "Large-scale direct slam for omnidirectional cameras," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 141–148.

[30] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, and P. Taddei, "Rawseeds ground truth collection systems for indoor self-localization and mapping," *Autonomous Robots*, vol. 27, no. 4, pp. 353–371, 2009.

[31] F. Chang, E. Jafarzadeh, J. Del Gatto, G. Cran, and H. Sadjadi, "Failure mode investigation to enable lidar health monitoring for automotive application," in *Annual Conference of the PHM Society*, vol. 15, 2023.

[32] K. Chen, J. Zhang, J. Liu, Q. Tong, R. Liu, and S. Chen, "Semantic visual simultaneous localization and mapping: A survey," *arXiv preprint arXiv:2209.06428*, 2022.

[33] W. Chen, G. Shang, A. Ji, C. Zhou, X. Wang, C. Xu, Z. Li, and K. Hu, "An overview on visual slam: From tradition to semantic," *Remote Sensing*, vol. 14, no. 13, p. 3010, 2022.

[34] Y. Choi, N. Kim, S. Hwang, K. Park, J. S. Yoon, K. An, and I. S. Kweon, "Kaist multi-spectral day/night data set for autonomous and assisted driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 3, pp. 934–948, 2018.

[35] G. Chustz and S. Saripalli, "Rooad: Rellis off-road odometry analysis dataset," *arXiv preprint arXiv:2109.08228*, 2021.

[36] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[37] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[38] S. Cortés, A. Solin, E. Rahtu, and J. Kannala, "Advio: An authentic dataset for visual-inertial odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 419–434.

[39] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in neural information processing systems*, vol. 26, 2013.

[40] S. Daftry, S. Zeng, J. A. Bagnell, and M. Hebert, "Introspective perception: Learning to predict failures in vision systems," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 1743–1750.

[41] J. I. Daoud, "Multicollinearity and regression analysis," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 949, 2017, p. 012 009.

[42] A. J. Davison, "Futuremapping: The computational structure of spatial ai systems," *arXiv preprint arXiv:1803.11288*, 2018.

[43] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[44] F. Dellaert, M. Kaess, *et al.*, "Factor graphs for robot perception," *Foundations and Trends® in Robotics*, vol. 6, no. 1-2, pp. 1–139, 2017.

[45] J. Delmerico, T. Cieslewski, H. Rebecq, M. Faessler, and D. Scaramuzza, "Are we ready for autonomous drone racing? the UZH-FPV drone racing dataset," in *IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019.

[46] T. J. DiCiccio and B. Efron, "Bootstrap confidence intervals," *Statistical science*, vol. 11, no. 3, pp. 189–228, 1996.

[47] P. M. Djuric, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, "Particle filtering," *IEEE signal processing magazine*, vol. 20, no. 5, pp. 19–38, 2003.

[48] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[49] R. Dubois, A. Eudes, and V. Frémont, "Airmuseum: A heterogeneous multi-robot dataset for stereo-visual and inertial simultaneous localization and mapping," in *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, IEEE, 2020, pp. 166–172.

[50] K. Ebadi, L. Bernreiter, H. Biggie, G. Catt, Y. Chang, A. Chatterjee, C. E. Denniston, S.-P. Deschênes, K. Harlow, S. Khattak, *et al.*, "Present and future of slam in extreme environments: The darpa subt challenge," *IEEE Transactions on Robotics*, 2023.

[51] J. Engel, V. Usenko, and D. Cremers, "A photometrically calibrated benchmark for monocular visual odometry," in *arXiv:1607.02555*, Jul. 2016.

[52] Epic Games, *Unreal engine*, version 4.22.1, Apr. 25, 2019. [Online]. Available: `https://www.unrealengine.com`.

[53] M. Faessler, F. Fontana, C. Forster, and D. Scaramuzza, "Automatic re-initialization and failure recovery for aggressive flight with a monocular vision-based quadrotor," in *2015 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2015, pp. 1722–1729.

[54] M.-A. Félix and A. Wagner, "Robustness and evolution: Concepts, insights and challenges from a developmental model system," *Heredity*, vol. 100, no. 2, pp. 132–140, 2008.

[55] A. Fornasier, M. Scheiber, A. Hardt-Stremayr, R. Jung, and S. Weiss, "Vinseval: Evaluation framework for unified testing of consistency and robustness of visual-inertial navigation system algorithms," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 13 754–13 760.

[56] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig, "Virtual worlds as proxy for multi-object tracking analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4340–4349.

[57] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, Oct. 2012, ISSN: 1552-3098.

[58] A. R. Gaspar, A. Nunes, A. M. Pinto, and A. Matos, "Urban@ cras dataset: Benchmarking of visual odometry and slam techniques," *Robotics and Autonomous Systems*, vol. 109, pp. 59–67, 2018.

[59] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, pp. 3354–3361.

[60] A. S. Gibson, *Exposure and Understanding the Histogram*. Peachpit Press, 2014.

[61] B. Glocker, S. Izadi, J. Shotton, and A. Criminisi, "Real-time rgb-d camera relocalization," in *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, IEEE, 2013, pp. 173–179.

[62] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[63] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and f-score, with implication for evaluation," in *European conference on information retrieval*, Springer, 2005, pp. 345–359.

[64] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" *Advances in neural information processing systems*, vol. 35, pp. 507–520, 2022.

[65] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics*, vol. 23, no. 1, pp. 34–46, 2007.

[66] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[67] M. Helmberger, K. Morin, N. Kumar, D. Wang, Y. Yue, G. Cioffi, and D. Scaramuzza, "The hilti slam challenge dataset," *arXiv preprint arXiv:2109.11316*, 2021.

[68] H. Herrman, D. E. Stewart, N. Diaz-Granados, E. L. Berger, B. Jackson, and T. Yuen, "What is resilience?" *The Canadian Journal of Psychiatry*, vol. 56, no. 5, pp. 258–265, 2011.

[69] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2d lidar slam," in *2016 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2016, pp. 1271–1278.

[70] T. Hinzmann, T. Stastny, G. Conte, P. Doherty, P. Rudol, M. Wzorek, E. Galceran, R. Siegwart, and I. Gilitschenski, "Collaborative 3d reconstruction using heterogeneous uavs: System and experiments," in *International Symposium on Experimental Robotics*, Springer, 2016, pp. 43–56.

[71] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 328–341, 2007.

[72] B. K. Horn, "Closed-form solution of absolute orientation using unit quaternions," *Josa a*, vol. 4, no. 4, pp. 629–642, 1987.

[73] S. Hossain and X. Lin, "Uncertainty-aware tightly-coupled gps fused lio-slam," *arXiv preprint arXiv:2209.10047*, 2022.

[74] A. S. Huang, M. Antone, E. Olson, L. Fletcher, D. Moore, S. Teller, and J. Leonard, "A high-rate, heterogeneous data set from the darpa urban challenge," *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1595–1601, 2010.

[75] I. Hwang, S. Kim, Y. Kim, and C. E. Seah, "A survey of fault detection, isolation, and reconfiguration methods," *IEEE transactions on control systems technology*, vol. 18, no. 3, pp. 636–653, 2009.

[76] A. Jafarzadeh, M. L. Antequera, P. Gargallo, Y. Kuang, C. Toft, F. Kahl, and T. Sattler, "Crowddriven: A new challenging dataset for outdoor visual localization," *arXiv preprint arXiv:2109.04527*, 2021.

[77] H. Jeong and H. Lee, "Cnn-based fault detection of scan matching for accurate slam in dynamic environments," *Sensors*, vol. 23, no. 6, p. 2940, 2023.

[78] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex urban dataset with multi-level sensors from highly diverse urban environments," *The International Journal of Robotics Research*, p. 0 278 364 919 843 996, 2019.

[79] L. Jinyu, Y. Bangbang, C. Danpeng, W. Nan, Z. Guofeng, and B. Hujun, "Survey and evaluation of monocular visual-inertial slam algorithms for augmented reality," *Virtual Reality & Intelligent Hardware*, vol. 1, no. 4, pp. 386–410, 2019.

[80] B. Kellalib, N. Achour, and F. Demim, "Sensors faults detection and isolation using ekf-slam for a mobile robot," in *2019 International Conference on Advanced Electrical Engineering (ICAEE)*, IEEE, 2019, pp. 1–7.

[81] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering.," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[82] A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba, "Undoing the damage of dataset bias," in *European Conference on Computer Vision*, Springer, 2012, pp. 158–171.

[83] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, IEEE, 2007, pp. 225–234.

[84] S. Klenk, J. Chui, N. Demmel, and D. Cremers, "Tum-vie: The tum stereo visual-inertial event dataset," in *International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[85] M. Kok, J. D. Hol, and T. B. Schön, "Using inertial sensors for position and orientation estimation," *arXiv preprint arXiv:1704.06053*, 2017.

[86] K. Konolige, "Projected texture stereo," in *2010 IEEE International Conference on Robotics and Automation*, IEEE, 2010, pp. 148–155.

[87] S. Koos, A. Cully, and J.-B. Mouret, "Fast damage recovery in robotics with the t-resilience algorithm," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1700–1723, 2013.

[88] A. Kortylewski, B. Egger, A. Morel-Forster, A. Schneider, T. Gerig, C. Blumer, C. Reyneke, and T. Vetter, "Can synthetic faces undo the damage of dataset bias to face recognition and facial landmark detection?" *arXiv preprint arXiv:1811.08565*, 2018.

[89] C. B. Kuhn, M. Hofbauer, G. Petrovic, and E. Steinbach, "Introspective black box failure prediction for autonomous driving," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 1907–1913.

[90] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *2011 IEEE international conference on robotics and automation*, IEEE, 2011, pp. 3607–3613.

[91] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *Journal of field robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[92] K. Leung, D. Lühr, H. Houshiar, F. Inostroza, D. Borrmann, M. Adams, A. Nüchter, and J. Ruiz del Solar, "Chilean underground mine dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 16–23, 2017.

[93] K. Y. Leung, Y. Halpern, T. D. Barfoot, and H. H. Liu, "The utias multi-robot cooperative localization and mapping dataset," *The International Journal of Robotics Research*, vol. 30, no. 8, pp. 969–974, 2011.

[94] D. Li, X. Shi, Q. Long, S. Liu, W. Yang, F. Wang, Q. Wei, and F. Qiao, "Dxslam: A robust and efficient visual slam system with deep features," in *2020 IEEE/RSJ International conference on intelligent robots and systems (IROS)*, IEEE, 2020, pp. 4958–4965.

[95] R. Li, S. Wang, and D. Gu, "Deepslam: A robust monocular slam system with unsupervised deep learning," *IEEE Transactions on Industrial Electronics*, vol. 68, no. 4, pp. 3577–3587, 2020.

[96] W. Li, S. Saeedi, J. McCormac, R. Clark, D. Tzoumanikas, Q. Ye, Y. Huang, R. Tang, and S. Leutenegger, "Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset," *arXiv preprint arXiv:1809.00716*, 2018.

[97] J. Lin and F. Zhang, "R 3 live: A robust, real-time, rgb-colored, lidar-inertial-visual tightly-coupled state estimation and mapping package," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 10 672–10 678.

[98] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.

[99] H. Liu, X. Wei, M. Perusquia-Hernandez, N. Isoyama, H. Uchiyama, and K. Kiyokawa, "Duet: Improving inertial-based odometry via deep imu online calibration," *IEEE Transactions on Instrumentation and Measurement*, 2023.

[100] J. Liu, K. Chen, R. Liu, Y. Yang, Z. Wang, and J. Zhang, "Robust and accurate multi-agent slam with efficient communication for smart mobiles," in *2022 International Conference on Robotics and Automation (ICRA)*, IEEE, 2022, pp. 2782–2788.

[101] Y. Liu, Y. Fu, F. Chen, B. Goossens, W. Tao, and H. Zhao, "Simultaneous localization and mapping related datasets: A comprehensive survey," *arXiv preprint arXiv:2102.04036*, 2021.

[102] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[103] M. Luperto, V. Castelli, and F. Amigoni, "Predicting performance of slam algorithms," *arXiv preprint arXiv:2109.02329*, 2021.

[104] A. Macario Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A comprehensive survey of visual slam algorithms," *Robotics*, vol. 11, no. 1, p. 24, 2022.

[105] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[106] A. L. Majdik, C. Till, and D. Scaramuzza, "The zurich urban micro aerial vehicle dataset," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 269–273, 2017.

[107] S. Mani, A. Sankaran, S. Tamilselvam, and A. Sethi, "Coverage testing of deep learning models using dataset characterization," *arXiv preprint arXiv:1911.07309*, 2019.

[108] P. Martin-Breen and J. M. Anderies, "Resilience: A literature review," 2011.

[109] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 18 039–18 048.

[110] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, "Real-time semantic mapping for autonomous off-road navigation," in *Field and Service Robotics*, Springer, 2018, pp. 335–350.

[111] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "Semanticfusion: Dense 3d semantic mapping with convolutional neural networks," in *2017 IEEE International Conference on Robotics and automation (ICRA)*, IEEE, 2017, pp. 4628–4635.

[112] N. Merrill, Y. Guo, X. Zuo, X. Huang, S. Leutenegger, X. Peng, L. Ren, and G. Huang, "Symmetry and uncertainty-aware object slam for 6dof object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 901–14 910.

[113] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[114] M. J. Milford, G. F. Wyeth, and D. Prasser, "Ratslam: A hippocampal model for simultaneous localization and mapping," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, IEEE, vol. 1, 2004, pp. 403–408.

[115] M. Miller, S.-J. Chung, and S. Hutchinson, "The visual–inertial canoe dataset," *The International Journal of Robotics Research*, vol. 37, no. 1, pp. 13–20, 2018.

[116] S. Mokssit, D. B. Licea, B. Guermah, and M. Ghogho, "Deep learning techniques for visual slam: A survey," *IEEE Access*, vol. 11, pp. 20 026–20 050, 2023.

[117] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit, *et al.*, "Fastslam: A factored solution to the simultaneous localization and mapping problem," *Aaai/iaai*, vol. 593598, pp. 593–598, 2002.

[118] C. Mostegel, J. Ye, Y. Luo, and Y. Liu, "Overlap displacement error: Are your slam poses map-consistent?" In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2021, pp. 5336–5343.

[119] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[120] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: A versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[121] R. Mur-Artal and J. D. Tardos, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[122] V. Narayanan and M. Likhachev, "Deliberative object pose estimation in clutter," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 3125–3130.

[123] P. K. Nathan Silberman Derek Hoiem and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *ECCV*, 2012.

[124] F. Negrello, M. Garabini, G. Grioli, N. Tsagarakis, A. Bicchi, and M. G. Catalano, "Benchmarking resilience of artificial hands," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 8374–8380.

[125] A. Noureldin, T. B. Karamat, and J. Georgy, *Fundamentals of inertial navigation, satellite-based positioning and their integration*. Springer Science & Business Media, 2012.

[126] P. W. Oliveira, G. A. Barreto, *et al.*, "A general framework for optimal tuning of pid-like controllers for minimum jerk robotic trajectories," *Journal of Intelligent & Robotic Systems*, pp. 1–20, 2020.

[127] D. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *Journal of artificial intelligence research*, vol. 11, pp. 169–198, 1999.

[128] E. Palazzolo, J. Behley, P. Lottes, P. Giguere, and C. Stachniss, "Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 7855–7862.

[129] V. M. Panaretos and Y. Zemel, "Statistical aspects of wasserstein distances," *Annual review of statistics and its application*, vol. 6, pp. 405–431, 2019.

[130] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford campus vision and lidar data set," *International Journal of Robotics Research*, vol. 30, no. 13, pp. 1543–1552, 2011.

[131] S. Park, T. Schöps, and M. Pollefeys, "Illumination change robustness in direct visual slam," in *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 4523–4530.

[132] R. Pearson, *Exploring data in engineering, the sciences, and medicine*. OUP USA, 2011.

[133] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[134] E. Peli, "Contrast in complex images," *JOSA A*, vol. 7, no. 10, pp. 2032–2040, 1990.

[135] S. Pertuz, D. Puig, and M. A. Garcia, "Analysis of focus measure operators for shape-from-focus," *Pattern Recognition*, vol. 46, no. 5, pp. 1415–1432, 2013.

[136] T. Peynot, S. Scheding, and S. Terho, "The Marulan Data Sets: Multi-Sensor Perception in Natural Environment with Challenging Conditions," *International Journal of Robotics Research*, vol. 29, no. 13, pp. 1602–1607, Nov. 2010.

[137] B. Pfrommer, N. Sanket, K. Daniilidis, and J. Cleveland, "Penncosyvio: A challenging visual inertial odometry benchmark," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 3847–3854.

[138] C. Pirchheim, D. Schmalstieg, and G. Reitmayr, "Handling pure camera rotation in keyframe-based slam," in *2013 IEEE international symposium on mixed and augmented reality (ISMAR)*, IEEE, 2013, pp. 229–238.

[139] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. J. Berlles, "S-ptam: Stereo parallel tracking and mapping," *Robotics and Autonomous Systems*, vol. 93, pp. 27–42, 2017.

[140] T. Pire, M. Mujica, J. Civera, and E. Kofman, "The rosario dataset: Multisensor data for localization and mapping in agricultural environments," *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 633–641, 2019.

[141] S. Poddar, V. Kumar, and A. Kumar, "A comprehensive overview of inertial sensor calibration techniques," *Journal of Dynamic Systems, Measurement, and Control*, vol. 139, no. 1, p. 011 006, 2017.

[142] W. H. Press, *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press, 2007.

[143] P. Probst, M. N. Wright, and A.-L. Boulesteix, "Hyperparameters and tuning strategies for random forest," *Wiley Interdisciplinary Reviews: data mining and knowledge discovery*, vol. 9, no. 3, e1301, 2019.

[144] A. Prolubnikov, "The set-cover problem with interval weights and the greedy algorithm for its solution," *Comput. Technologies*, vol. 20, no. 6, pp. 70–84, 2015.

[145] A. Prorok, M. Malencia, L. Carlone, G. S. Sukhatme, B. M. Sadler, and V. Kumar, "Beyond robustness: A taxonomy of approaches towards resilient multi-robot systems," *arXiv preprint arXiv:2109.12343*, 2021.

[146] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[147] J. Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, *Dataset shift in machine learning*. Mit Press, 2008.

[148] S. Rabiee and J. Biswas, "Iv-slam: Introspective vision for simultaneous localization and mapping," in *Conference on Robot Learning*, PMLR, 2021, pp. 1100–1109.

[149] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The newer college dataset: Handheld lidar, inertial and vision with ground truth," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[150] J. A. Rice, *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[151] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: An open-source library for real-time metric-semantic localization and mapping," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 1689–1696.

[152] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European conference on computer vision*, Springer, 2006, pp. 430–443.

[153] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, Ieee, 2011, pp. 2564–2571.

[154] W. Rudin *et al.*, *Principles of mathematical analysis*. McGraw-hill New York, 1976, vol. 3.

[155] S. Saeedi, E. D. Carvalho, W. Li, D. Tzoumanikas, S. Leutenegger, P. H. Kelly, and A. J. Davison, "Characterizing visual localization and mapping datasets," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 6699–6705.

[156] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets," *PloS one*, vol. 10, no. 3, e0118432, 2015.

[157] H. M. Salkin and C. A. De Kluyver, "The knapsack problem: A survey," *Naval Research Logistics Quarterly*, vol. 22, no. 1, pp. 127–144, 1975.

[158] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[159] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of computer vision*, vol. 37, no. 2, pp. 151–172, 2000.

[160] T. Schops, T. Sattler, and M. Pollefeys, "Bad slam: Bundle adjusted direct rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 134–144.

[161] S. H. Schot, "Jerk: The time rate of change of acceleration," *American Journal of Physics*, vol. 46, no. 11, pp. 1090–1094, 1978.

[162] D. Schubert, N. Demmel, L. von Stumberg, V. Usenko, and D. Cremers, "Rolling-shutter modelling for direct visual-inertial odometry," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 2462–2469.

[163] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The tum vi benchmark for evaluating visual-inertial odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2018, pp. 1680–1687.

[164] F. Secci and A. Ceccarelli, "On failures of rgb cameras and their effects in autonomous driving applications," in *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*, IEEE, 2020, pp. 13–24.

[165] T. Shan, B. Englot, C. Ratti, and D. Rus, "Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *2021 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2021, pp. 5692–5698.

[166] Z. Shen, J. Liu, Y. He, X. Zhang, R. Xu, H. Yu, and P. Cui, "Towards out-of-distribution generalization: A survey," *arXiv preprint arXiv:2108.13624*, 2021.

[167] X. Shi, D. Li, P. Zhao, Q. Tian, Y. Tian, Q. Long, C. Zhu, J. Song, F. Qiao, L. Song, Y. Guo, Z. Wang, Y. Zhang, B. Qin, W. Yang, F. Wang, R. H. M. Chan, and Q. She, "Are we ready for service robots? the OpenLORIS-Scene datasets for lifelong SLAM," in *2020 International Conference on Robotics and Automation (ICRA)*, 2020, pp. 3139–3145.

[168] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *Proceedings of the International Conference on Computer Vision - Workshop on 3D Representation and Recognition*, 2011.

[169] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Computer Vision, IEEE International Conference on*, IEEE Computer Society, vol. 3, 2003, pp. 1470–1470.

[170] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, 2009.

[171] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare: A flexible quadrotor simulator," 2020.

[172] M. Steinbuch, J. C. Terlouw, and O. H. Bosgra, "Robustness analysis for real and complex perturbations applied to an electro-mechanical system," in *1991 American Control Conference*, IEEE, 1991, pp. 556–561.

[173] J. Stelling, U. Sauer, Z. Szallasi, F. J. Doyle III, and J. Doyle, "Robustness of cellular functions," *Cell*, vol. 118, no. 6, pp. 675–685, 2004.

[174] I. Stojmenovic, *Handbook of sensor networks: algorithms and architectures*. John Wiley & Sons, 2005, vol. 49.

[175] D. Stoten and H. Benchoubane, "Robustness of a minimal controller synthesis algorithm," *International Journal of Control*, vol. 51, no. 4, pp. 851–861, 1990.

[176] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular slam: Why filter?" In *2010 IEEE International Conference on Robotics and Automation*, IEEE, 2010, pp. 2657–2664.

[177] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[178] P. Sundvall and P. Jensfelt, "Fault detection for mobile robots using redundant positioning systems," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, IEEE, 2006, pp. 3781–3786.

[179] G. J. Sussman, "Building robust systems an essay," *Citeseer*, vol. 113, p. 1324, 2007.

[180] M. Thirumarimurugan, N. Bagyalakshmi, and P. Paarkavi, "Comparison of fault detection and isolation methods: A review," in *2016 10th International Conference on Intelligent Systems and Control (ISCO)*, IEEE, 2016, pp. 1–6.

[181] S. Thrun, "Particle filters in robotics.," in *UAI*, Citeseer, vol. 2, 2002, pp. 511–518.

[182] ——, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[183] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *CVPR 2011*, IEEE, 2011, pp. 1521–1528.

[184] P. Toth, "Dynamic programming algorithms for the zero-one knapsack problem," *Computing*, vol. 25, no. 1, pp. 29–45, 1980.

[185] B. Turhan, "On the dataset shift problem in software engineering prediction models," *Empirical Software Engineering*, vol. 17, no. 1, pp. 62–74, 2012.

[186] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 2043–2050.

[187] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," 2020.

[188] O. Wasenmüller, M. Meyer, and D. Stricker, "Corbs: Comprehensive rgb-d benchmark for slam using kinect v2," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2016, pp. 1–7.

[189] L. Wei, C. Cappelle, and Y. Ruichek, "Camera/laser/gps fusion method for vehicle positioning under extended nis-based sensor validation," *IEEE transactions on Instrumentation and Measurement*, vol. 62, no. 11, pp. 3110–3122, 2013.

[190] W. Wen, Y. Zhou, G. Zhang, S. Fahandezh-Saadi, X. Bai, W. Zhan, M. Tomizuka, and L.-T. Hsu, "Urbanloco: A full sensor suite dataset for mapping and localization in urban scenes," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 2310–2316.

[191] P. Wenzel, R. Wang, N. Yang, Q. Cheng, Q. Khan, L. von Stumberg, N. Zeller, and D. Cremers, "4Seasons: A cross-season dataset for multi-weather SLAM in autonomous driving," in *Proceedings of the German Conference on Pattern Recognition (GCPR)*, 2020.

[192] H. D. Whyte, "Simultaneous localisation and mapping (slam): Part i the essential algorithms," *Robotics and Automation Magazine*, 2006.

[193] M. Wied, J. Oehmen, and T. Welo, "Conceptualizing resilience in engineering systems: An analysis of the literature," *Systems Engineering*, vol. 23, no. 1, pp. 3–13, 2020.

[194] R. R. Wilcox, "Trimmed means," *Wiley StatsRef: Statistics Reference Online*, 2014.

[195] B. Williams, M. Cummins, J. Neira, P. Newman, I. Reid, and J. Tardós, "A comparison of loop closing techniques in monocular slam," *Robotics and Autonomous Systems*, vol. 57, no. 12, pp. 1188–1197, 2009.

[196] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1127–1134, 2020.

[197] Y. Yang and G. Huang, "Observability analysis of aided ins with heterogeneous features of points, lines, and planes," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1399–1418, 2019.

[198] W. Ye, Y. Zhao, and P. A. Vela, "Characterizing slam benchmarks and methods for the robust perception age," *Workshop on Dataset Generation and Benchmarking of SLAM Algorithms for Robotics and VR/AR, IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2019.

[199] H. Yin, L. Tang, X. Ding, Y. Wang, and R. Xiong, "A failure detection method for 3d lidar based localization," in *2019 Chinese Automation Congress (CAC)*, IEEE, 2019, pp. 4559–4563.

[200] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[201] N. Zeller, F. Quint, and U. Stilla, "A synchronized stereo and plenoptic visual odometry dataset," *arXiv preprint arXiv:1807.09372*, 2018.

[202] C. Zhang, J. Han, Y. Zou, K. Dong, Y. Li, J. Ding, and X. Han, "Detecting the anomalies in lidar pointcloud," *arXiv preprint arXiv:2308.00187*, 2023.

[203] J. Zhang, S. Singh, *et al.*, "Loam: Lidar odometry and mapping in real-time.," in *Robotics: Science and systems*, Berkeley, CA, vol. 2, 2014, pp. 1–9.

[204] P. Zhang, J. Wang, A. Farhadi, M. Hebert, and D. Parikh, "Predicting failures of vision systems," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3566–3573.

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

[206] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3d perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, 2018.