Localization and Mapping through Multi-Sensor Fusion for Pipe Inspection: From Theory To Deployment

Tina Tian

CMU-RI-TR-25-17
May 6, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Dr. Howie Choset, *chair*
Dr. George Kantor
Chao Cao

*Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Robotics.*

To the wonderful world and beautiful people.

Abstract

Pipelines are critical infrastructure for transporting a variety of materials, such as natural gas and stormwater. However, much of this infrastructure is aging, poorly documented, and difficult to inspect, particularly pipelines that are small in diameter or buried underground. Inadequate monitoring can lead to failures with serious consequences, including service disruptions, environmental damage, and public safety risks. This thesis investigates how accurate localization and mapping can be achieved inside space-constrained, long pipelines through multi-sensor fusion, advancing both algorithmic methods and practical deployment strategies. It introduces VILL-SLAM, a system and method that combines a monocular camera, inertial measurement unit (IMU), ring-shaped laser profilometer, and LiDAR to enable real-time localization and dense RGB-D mapping with sub-millimeter reconstruction accuracy and less than 1% drift in 12-inch pipes. This approach was validated through real-world deployments in underground pipes, demonstrating its practical effectiveness. Recognizing the importance of real-world readiness, the thesis also introduces a systems engineering methodology, Design for Deployment (D4D), which emphasizes formal operational modeling, early user engagement, and iterative development through field trials. Together, these contributions offer a comprehensive, field-validated framework for in-pipe robotic inspection, advancing confined-space localization and mapping, and providing a blueprint for translating robotics research into deployable solutions for the infrastructure maintenance industry.

Acknowledgments

First and foremost, I wish to give a Power Thank You to my advisor, Dr. Howie Choset, whose sharp insights, radical candor, and unwavering commitment to using robotics to better society have profoundly shaped my academic journey. His dedication to improving lives through innovation has been both inspiring and transformative, guiding me as a researcher and as an individual striving to create meaningful change.

I am also profoundly grateful to my mentor, Lu Li, for his invaluable guidance and encouragement throughout this journey. His ability to dissect problems from first principles, his rigor in practicing the scientific method, and his focus on clear communication have profoundly shaped my approach to research and learning. Beyond that, his innate curiosity and boundless creativity have been a constant source of inspiration. Words cannot adequately express my gratitude for his mentorship, which has not only made me a better thinker but also a more thoughtful communicator and listener.

I would also like to extend my sincere thanks to Avaneesh Murugesan, Mengning Wu, Luyuan Wang, Xinzhi Yan, and Geordan Gutow, whose insightful discussions and collaborative spirit have enriched my understanding of the research and methodology presented in this work. Additionally, I am deeply thankful to Ralph Boirum, Yizhu Gu, Nate Shoemaker-Trejo, and Jiaxi Zheng, who contributed to the design, iteration, and deployment of the robotic system that forms the backbone of this thesis. Their expertise, dedication, and teamwork have been instrumental in transforming ideas into reality.

My appreciation also goes to our field deployment partners. These collaborations provided invaluable real-world insights into the challenges and opportunities of transitioning robotics technology from the lab to practical applications.

Finally, while this thesis marks the culmination of one chapter, it is by no means the end of my journey. Robotics continues to be my lens for exploring the world, and I look forward to bringing meaningful value to society through the innovations and possibilities it unlocks.

Funding

This work was supported by the Department of Energy Advanced Research Projects Agency-Energy (ARPA-E) under the Rapid Encapsulation of Pipelines Avoiding Intensive Replacement (REPAIR) program.

Contents

1 Introduction 1

2 Background and Related Work 5

2.1 State Estimation and SLAM Preliminaries 5

2.1.1 Filtering-Based Approaches 6

2.1.2 Optimization-Based Approaches 7

2.1.3 Factor Graphs and Solvers 8

2.1.4 IMU Preintegration 8

2.1.5 SLAM in General Environments 9

2.2 In-Pipe Robot Sensing, Localization, and Reconstruction 10

2.2.1 Challenges in Confined-Space Pipe Environments 10

2.2.2 In-Pipe Localization and Mapping 12

3 VILL-SLAM: Localization and Dense RGB-D Reconstruction for Pipe Environments 15

3.1 System Overview 16

3.1.1 Hardware System Overview 16

3.1.2 Software System Overview 17

3.2 Visual-Inertial-Laser-LiDAR SLAM 18

3.2.1 Sensor Data Preprocessing 18

3.2.2 Visual-Depth Association 19

3.2.3 Estimator Initialization 19

3.2.4 Sliding-Window-Based Factor Graph Optimization 20

3.2.5 Map Registration 24

3.3 Evaluation Experiments 24

3.3.1 Qualitative Analysis 24

3.3.2 Localization Accuracy Evaluation 25

3.3.3 3D Reconstruction Evaluation 26

3.3.4 Simulation 28

3.4 Field Deployment Outcomes in Natural Gas Pipes 29

3.5 Conclusion and Discussion 31

4 Smoothing the Road from Lab to Field: A Design for Deployment (D4D) Framework for Robotic Systems Engineering 33

4.1 Limitations of Existing Systems Engineering Approaches 34
4.2 A Design-Inspired Systems Engineering Framework: Design for Deployment (D4D) 35
4.2.1 Concept of Operation 36
4.2.2 Requirement Definition and Conceptual Design 39
4.2.3 Detailed Design, Implementation, and Verification 39
4.2.4 Field Deployment 40
4.3 Case Study: Natural Gas Pipe Inspection Robot 40
4.3.1 Concept of Operation and Requirement Definition 40
4.3.2 System Verification 41
4.3.3 Field Deployment and On-Site Operations 41
4.3.4 System Limitations and Observed Gaps 42
4.3.5 Key Takeaways 43
4.4 Case Study: Stormwater Pipe Inspection Robot with Structured Deployment Planning 43
4.4.1 Concept of Operation and Requirement Definition 43
4.4.2 System Verification 44
4.4.3 Field Deployment and On-Site Operations 45
4.4.4 Key Takeaways 45
4.5 Conclusion and Discussion 46

5 Conclusion 47

Bibliography 51

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

1.1 Examples of pipe failures. (a) Natural gas pipeline explosion in San Bruno, California, in 2010, resulting in eight fatalities [1]. (b) Severely deformed stormwater pipe still in service [2]. (c) Failure of a duct bank (a group of underground electrical conduits) [3]. 1

1.2 Pipe inspection tools. (a) Pipe inspection gauges (PIGs) [4]. (b) Snake cameras [5]. (c) CCTV crawlers [6]. 2

3.1 The in-pipe mapping sensor hardware prototype using the proposed VILL-SLAM method to fuse multiple sensory information, which produces a photo-realistic dense RGB-D reconstruction of a 12-inch pipe segment's inner surface. 16

3.2 We capture visual and profiling image frames using one camera by quickly switching between visual and profiling frames using a micro-controller, measuring RGB and depth information in a near-concurrent fashion. 17

3.3 VILL-SLAM software architecture and information flow chart. 18

3.4 At the core of our method, a factor graph is used to represent various states with their related constraints within a sliding window. 21

3.5 LiDAR cylinder factor. 23

3.6 Test environment and mapped point cloud visualization. (a) Test site with a total pipe length of 50m. (b) Mapping results from VINS-Mono [7]. Note that other tested methods including LOAM [8], FAST-LIO2 [9], ORB-SLAM2 [10] and dense mapping pipelines like SSL-SLAM2 [11] and RGBDTAM [12] all failed to produce reasonable results under same testing conditions. (c) Dense colorized 3D map created by VILL-SLAM (ours) in the pipe test site and the zoom-in views of some regions of interest. (d) 3D map created by VILL-SLAM in a pipe elbow sample. 25

3.7 Example ATE plots of VLI-SLAM and VILL-SLAM. (a) Visualization of the ATE of our current work VILL-SLAM (with LiDAR) and our prior method VLI-SLAM (without LiDAR) using data collected in one trial. (b) The corresponding 3D maps of the pipe using the two methods. 27

3.8 Workflow of 3D reconstruction accuracy assessment using an evaluation jig and in-pipe 3D scanning experiments. 27

3.9 In-pipe sensing simulation in UE4. (a,b) Examples of visual and laser profiling frames captured inside a real pipe. (c,d) Corresponding visual and laser profiling frames captured inside a simulated bendy pipe. (e) The simulated pipe environment where frames (c) and (d) are captured. (f) 3D point cloud reconstruction of the simulated pipe interior. (g) Estimated path (red solid curve) versus true path (green dashed curve) within the simulated pipe, with the error (magnified 5 times for visualization) shaded in orange. 28

3.10 Data captured in a field deployment trial near a pipe dislocation spot. (a) Visual frame showing a gap, rust, and signs of water. (b) Out-of-roundness (OOR) plot of the corresponding laser profile, with warm colors indicating outward deformation and cool colors indicating inward deformation. (c) Mapped pipe trajectory overlaid on a satellite map, highlighting the dislocation spot. (d) Reconstructed 3D point cloud of the affected area. (e, f) Corresponding mesh with diameter and bump size analysis. 30

4.1 Systems engineering methods. (a) V-model [13]. (b) Agile method [14]. 35

4.2 The proposed Design for Deployment (D4D) framework, inspired by [13, 14, 15]. The process starts with 'Concept of Operation', followed by requirement definition, conceptual and detailed design, implementation, verification, and field deployment. Our method integrates user input (represented by the red color) from the beginning, which influences both technical development and operational planning. The goal is to align system capabilities with deployment conditions and user needs. 37

List of Tables

3.1 Mean and Standard Deviation of Localization Errors 26

Chapter 1

Introduction

Pipelines are a critical part of modern infrastructure, transporting essential resources and waste across cities and regions. However, much of this infrastructure is aging and ill-equipped to meet the demands of the future. Many pipelines in use today were installed over a century ago and are increasingly prone to failure due to material degradation, deformation, corrosion, and insufficient capacity. These failures can have severe consequences, ranging from service disruptions to environmental damage and public safety hazards (Fig. 1.1).

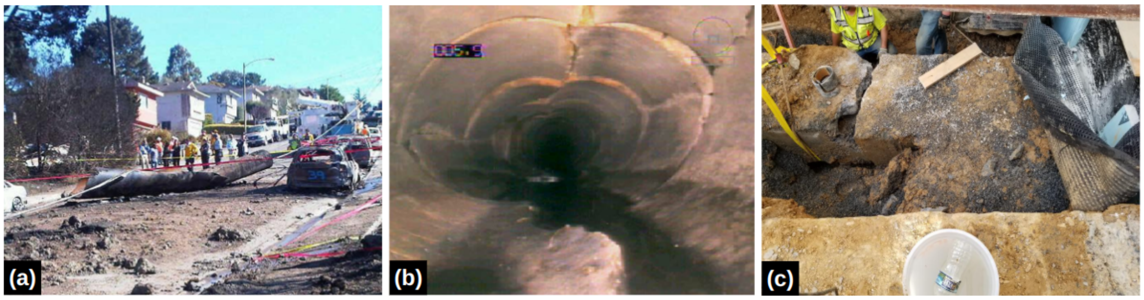

Figure 1.1: Examples of pipe failures. (a) Natural gas pipeline explosion in San Bruno, California, in 2010, resulting in eight fatalities [1]. (b) Severely deformed stormwater pipe still in service [2]. (c) Failure of a duct bank (a group of underground electrical conduits) [3].

In the case of natural gas pipelines, aging components pose serious risks. According to the Pipeline and Hazardous Materials Safety Administration (PHMSA), over 13,000 significant gas pipeline incidents have been reported in the United States over the past two decades, resulting in more than $11 billion in damages and numerous injuries and fatalities [16]. Stormwater systems face similar challenges. As climate change intensifies, storm and flood incidents in the U.S. are increasing by approximately 5% annually [17]. Inadequate and aging stormwater infrastructure contributes to an estimated $70 billion in damage annually, part of a broader cost of $180–496 billion attributed to flooding [18]. Electrical conduits are also under strain, with rising energy demands driven by data centers and artificial intelligence causing electricity consumption to double every four years [19]. Many of these underground conduits are crushed, deformed, or otherwise compromised, yet remain in active use.

Despite the clear need to maintain and upgrade pipeline systems, asset owners often lack basic information about pipe condition. Many pipelines are poorly documented, with records maintained only on outdated paper blueprints that may not reflect current ground truth. As a result, asset managers frequently do not know where pipelines are located, what condition they are in, or how close they are to failure. This hinders proactive maintenance, risk assessment, and system upgrades.

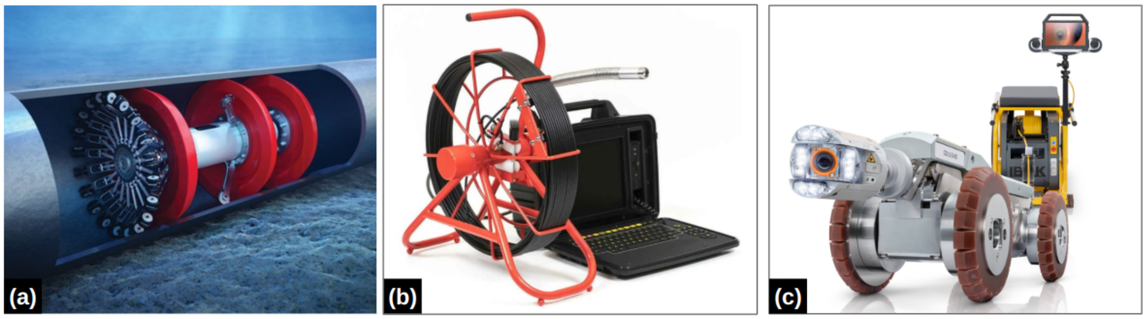

Figure 1.2: Pipe inspection tools. (a) Pipe inspection gauges (PIGs) [4]. (b) Snake cameras [5]. (c) CCTV crawlers [6].

Current inspection technologies (Fig. 1.2) are insufficient to meet these challenges. Common in-pipe inspection tools include pipeline inspection gauges (PIGs), borescopes (snake cameras), and closed-circuit television (CCTV) crawlers. PIGs [4], driven by internal pipeline flow, are effective for nondestructive evaluation tasks in large, pressurized pipes but are unsuitable for pipelines with significant deformation, blockages, or absent flow conditions. Snake cameras [5] are compact and inexpensive but rely entirely on manual operation, making them difficult to maneuver through complex pipelines, and they provide only 2D video without spatial localization or

3D reconstruction. CCTV crawlers [6, 20, 21], by contrast, assist navigation through robotic mobility and thus reduce manual intervention; however, they remain bulkier than snake cameras, restricting their use in small-diameter pipes, and they too depend largely on 2D imaging with limited spatial information.

A major shortcoming of existing in-pipe inspection methods is their inability to support reliable condition assessment and system performance analysis. Due to the lack of spatially accurate 3D representations, users face significant challenges in detecting geometric anomalies, quantifying defect dimensions, and conducting effective quality assurance and control (QA/QC). This limitation also hampers broader analysis of pipeline system performance. Moreover, the lack of accurate onboard localization necessitates additional manual processes, such as the deployment of sonde beacons [22] or ground-penetrating radar (GPR), to determine the locations of observed features. This reliance on external localization not only complicates field operations but also increases the cost and effort required to accurately document and remediate defects within the pipeline network.

To address these limitations, robotics-aided inspection systems using multi-sensor fusion and Simultaneous Localization and Mapping (SLAM) techniques offer a promising direction. By automatically localizing within the pipe and reconstructing dense spatial information, such systems have the potential to generate the visual, 3D, and geospatial data needed for detailed condition monitoring, defect quantification, and system-level assessment. However, applying SLAM in pipeline environments introduces unique challenges. Unlike structured indoor and open outdoor environments, which have been the primary focus of most SLAM methods, pipelines often have smooth, cylindrical surfaces with few distinctive geometric features, combined with extreme scale constraints and limited sensing fields of view, all of which limit the effectiveness of conventional SLAM hardware and algorithms.

This thesis presents a novel approach to in-pipe SLAM that addresses the challenges of confined, GPS-denied environments through sensor fusion and tailored algorithmic frameworks. Specifically, it introduces **VILL-SLAM** [23], which integrates a monocular camera (V), inertial measurement unit (I), structured-light laser profilometer (L), and LiDAR (L) into a compact sensor payload. By fusing complementary visual and depth measurements and leveraging the known structural characteristics of pipes as geometric constraints, VILL-SLAM resolves metric scale ambiguity, significantly reduces global drift, and achieves localization error below 1%. Furthermore, it enables photorealistic 3D reconstructions of pipe interiors with sub-millimeter accuracy, supporting high-fidelity inspection and analysis.

Custom robotic systems running these localization and mapping approaches have been deployed in real-world pipeline inspections across gas and stormwater domains. These deployments revealed a gap between laboratory research and field-readiness, especially in under-explored environments such as underground pipes. To address this gap, the thesis introduces a **Design for Deployment (D4D)** framework—a systems engineering approach for bridging the gap between prototype development and real-world use, informed by two field deployment case studies.

In summary, this thesis presents a comprehensive investigation into the design and implementation of in-pipe localization and mapping methods, and the deployment of robotic in-pipe inspection systems. The remaining chapters are organized as follows:

- Chapter 2 reviews the fundamentals of SLAM and state estimation, examines related work in in-pipe robotic sensing and SLAM, and outlines the unique challenges of pipeline environments.
- Chapter 3 details the VILL-SLAM system and its evaluation in large-diameter pipes (12 inches and above), including real-world deployment results.
- Chapter 4 presents the Design for Deployment (D4D) framework and illustrates its application through retrospective case studies of robotic system deployment in challenging field environments.
- Chapter 5 summarizes the key findings, discusses the implications for future research, and outlines potential directions for advancing robust, field-ready in-pipe inspection systems.

Chapter 2

Background and Related Work

2.1 State Estimation and SLAM Preliminaries

Robotic navigation in unknown environments relies heavily on state estimation, which is the foundation for both odometry and SLAM. State estimation involves determining a set of latent variables $\mathbf{x}$, typically including the robot's trajectory and, in the case of SLAM, also the positions of observed landmarks, given a sequence of noisy sensor measurements $\mathbf{z}_{1:k}$ and control inputs $\mathbf{u}_{1:k}$.

The problem can be formulated as a Maximum A Posteriori (MAP) estimation: finding the most probable state trajectory given the available data:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} p(\mathbf{x} \mid \mathbf{z}_{1:k}, \mathbf{u}_{1:k}), \tag{2.1}$$

where $\mathbf{x}$ denotes the entire sequence of states from time 1 to time k, and, in the case of SLAM, also includes the estimated landmark positions.

Using Bayes' theorem, this posterior can be rewritten as:

$$p(\mathbf{x} \mid \mathbf{z}_{1:k}, \mathbf{u}_{1:k}) \propto p(\mathbf{z}_{1:k} \mid \mathbf{x}) p(\mathbf{x} \mid \mathbf{u}_{1:k}), \tag{2.2}$$

where $p(\mathbf{z}_{1:k} \mid \mathbf{x})$ is the likelihood of observing the measurements given the states, and $p(\mathbf{x} \mid \mathbf{u}_{1:k})$ captures the motion model and any prior knowledge over the state evolution.

The distinction between **odometry** and **SLAM** lies in how the state is defined:

- In odometry, the state typically includes only the robot's pose over time, while mapping is handled separately.
- In SLAM, the state vector is augmented to also include the positions of landmarks observed in the environment:

$$\mathbf{x} = \{\mathbf{x}_{\text{robot}}, \mathbf{l}_1, \mathbf{l}_2, \ldots, \mathbf{l}_M\}, \tag{2.3}$$

  where $\mathbf{l}_i$ denotes the position of the i-th landmark. This joint estimation allows SLAM systems to enforce consistency between robot motion and environmental observations.

This formulation provides the foundation for both filtering-based and optimization-based approaches to state estimation, classified according to the structure of the inference process, as discussed next.

2.1.1 Filtering-Based Approaches

Filtering approaches estimate the current state sequentially by maintaining a belief $p(\mathbf{x}_k \mid \mathbf{z}_{1:k}, \mathbf{u}_{1:k})$, updated with new measurements.

Kalman Filter (KF)

The KF [24] assumes linear system dynamics and Gaussian noise, with the state $\mathbf{x}_k$ evolving as:

$$\mathbf{x}_k = \mathbf{F}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_k + \mathbf{w}_k, \quad \mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k, \tag{2.4}$$

where $\mathbf{F}$ is the state transition matrix, $\mathbf{B}$ is the control input matrix, $\mathbf{H}$ is the observation matrix, $\mathbf{u}_k$ is the control input, and $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q})$ and $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R})$ are process and measurement noise. Also denote the state estimation covariance as $\mathbf{P}_k$.

The KF recursively updates the state estimate through:

1. Prediction Step:

$$\hat{\mathbf{x}}_k^- = \mathbf{F}\hat{\mathbf{x}}_{k-1} + \mathbf{B}\mathbf{u}_k \tag{2.5}$$

$$\mathbf{P}_k^- = \mathbf{F}\mathbf{P}_{k-1}\mathbf{F}^\top + \mathbf{Q} \tag{2.6}$$

2. Update Step:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}^\top \left(\mathbf{H} \mathbf{P}_k^- \mathbf{H}^\top + \mathbf{R} \right)^{-1} \tag{2.7}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k \left(\mathbf{z}_k - \mathbf{H} \hat{\mathbf{x}}_k^- \right) \tag{2.8}$$

$$\mathbf{P}_k = \left(\mathbf{I} - \mathbf{K}_k \mathbf{H} \right) \mathbf{P}_k^- \tag{2.9}$$

Extended Kalman Filter (EKF)

For nonlinear systems, the EKF [25] extends the KF by linearizing around the current estimate using a first-order Taylor expansion, but can diverge in highly nonlinear scenarios. This limitation is addressed in the Unscented Kalman Filter (UKF) [26], which avoids direct linearization by approximating the distribution of states with a set of carefully chosen samples called sigma points. These points are propagated through the nonlinear dynamics, capturing the mean and covariance more accurately without requiring explicit Jacobians.

2.1.2 Optimization-Based Approaches

Unlike filtering-based methods, which estimate the state sequentially, optimization-based approaches solve for the entire sequence of states by considering all available measurements together, either as a full batch or within a sliding window.

Taking the negative logarithm and expanding the terms according to (2.2) transform the optimization problem in (2.1) into an equivalent minimization problem:

$$\begin{aligned} \mathbf{x}^* &= \arg\min_{\mathbf{x}} \left[-\log p(\mathbf{x} \mid \mathbf{z}_{1:k}, \mathbf{u}_{1:k}) \right] \\ &= \arg\min_{\mathbf{x}} \left[-\log p(\mathbf{x} \mid \mathbf{u}_{1:k}) - \sum_{k=1}^{K} \log p(\mathbf{z}_k \mid \mathbf{x}_k) \right] \end{aligned} \tag{2.10}$$

Each likelihood term $\log p(\mathbf{z}_k \mid \mathbf{x}_k)$ can be interpreted as enforcing a constraint between the estimated state and the observed data. Under Gaussian noise assumptions, minimizing the negative log-likelihood leads to a nonlinear least-squares formulation [27]:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_i \|\mathbf{e}_i(\mathbf{x})\|^2_{\mathbf{\Sigma}_i}, \tag{2.11}$$

where $\mathbf{e}_i(\mathbf{x})$ is the residual associated with the i-th measurement or constraint, depending on the current state estimate, and $\mathbf{\Sigma}_i$ is the covariance matrix capturing the uncertainty of the corresponding measurement.

In multi-sensor SLAM, each sensor modality (e.g., IMU, LiDAR, camera) contributes different residuals based on its observations, and the overall optimization problem integrates all these constraints into a unified cost function.

2.1.3 Factor Graphs and Solvers

The optimization-based SLAM problem, formulated as minimizing the sum of squared residuals, can also be structured as a factor graph [28]. Factor graphs represent the SLAM problem as a bipartite graph, where variable nodes (e.g., robot poses, landmarks) are connected to factor nodes (e.g., odometry, sensor measurements) that encode probabilistic constraints between variables. A common special case is the *pose graph*, where only robot poses are estimated, and factors represent relative pose constraints — such as those from odometry or loop closures [29]. Pose graph optimization (PGO) is especially useful for trajectory refinement.

Solvers like GTSAM [28], g2o [29], and Ceres [30] provide efficient nonlinear optimization over these graphs, leveraging sparse matrix techniques and incremental methods to handle large-scale SLAM problems.

2.1.4 IMU Preintegration

IMU sensors provide high-rate motion information, which is crucial for odometry in challenging environments. The inertial state evolves according to:

$$\dot{\mathbf{p}} = \mathbf{v}, \quad \dot{\mathbf{v}} = \mathbf{R}(\mathbf{a} - \mathbf{b}_a - \mathbf{n}_a), \quad \dot{\mathbf{R}} = \mathbf{R}\lfloor \boldsymbol{\omega} - \mathbf{b}_g - \mathbf{n}_g \rfloor_\times, \tag{2.12}$$

where $\mathbf{p} \in \mathbb{R}^3, \mathbf{v} \in \mathbb{R}^3, \mathbf{R} \in SO(3)$ are the position, velocity, and orientation; $\mathbf{a}$ is the measured linear acceleration; $\boldsymbol{\omega}$ is the measured angular velocity from the gyroscope; $\mathbf{b}_a, \mathbf{b}_g$ are the accelerometer and gyroscope biases; and $\mathbf{n}_a, \mathbf{n}_g$ are the respective sensor noise terms.

Direct integration of raw IMU measurements between consecutive keyframes would require re-computation during every optimization update, leading to high computational costs. Preintegration methods [31] address this by accumulating IMU measurements into a single preintegrated constraint that depends only on the relative motion between frames. This enables efficient incorporation of high-rate IMU data into optimization-based SLAM without repeated reintegration, improving computational efficiency while maintaining accuracy.

2.1.5 SLAM in General Environments

SLAM techniques developed for general indoor and outdoor environments often serve as a foundation for in-pipe localization and mapping. These methods vary significantly in terms of sensor configurations and algorithmic design.

Single-sensor SLAM methods, such as ORB-SLAM2 [10], LOAM [8], and SSL-SLAM [11], rely on visual or LiDAR inputs alone. While computationally efficient and hardware-light, they are sensitive to perceptual aliasing, motion blur, or sparse geometry—conditions commonly found in confined pipe environments.

Multi-sensor SLAM addresses these challenges by fusing complementary data sources. Combining visual or LiDAR odometry with inertial measurements, as seen in VINS-Fusion [7] and FAST-LIO [9], improves robustness against rapid motion and sensor drift. RGB-D SLAM systems utilize synchronized color (RGB) and depth (D) data to enable 3D reconstruction and metric-scale, colorized mapping [32]. More advanced systems such as Super Odometry [33] and R3LIVE [34] further integrate visual, inertial, and LiDAR data to maintain reliable performance under perceptual degradation or dynamic conditions. These methods achieve higher accuracy and resilience by leveraging the strengths of each sensing modality.

SLAM algorithms can also be categorized as either **conventional** or **learning-based**, according to whether they rely primarily on explicit algorithmic modeling or on data-driven learning. Conventional approaches, including the methods above, formulate SLAM as a state estimation problem using techniques such as nonlinear optimization or filtering. In contrast, learning-based SLAM systems employ deep neural networks for tasks like feature tracking [35], depth prediction [36], pose estimation [37], or loop closure [38]. Although learning-based methods show potential in certain environments, their reliance on large training datasets make them less attractive for applications in underexplored, data-scarce domains like pipelines.

2.2 In-Pipe Robot Sensing, Localization, and Reconstruction

A wide range of pipe inspection robots with diverse mechanisms and sensors have been proposed. Mechanisms, which have been surveyed in [39, 40], include wheeled or tracked vehicles, legged or inchworm robots, as well as snake, screw-driven, and parallel-actuated platforms. These robots have been integrated with diverse sensing modalities to support structural and environmental condition monitoring. Typical sensors include encoders [41], ultrasonic transducers [42], electromagnetic transducers [43], mechanical calipers [44], acoustic [45], structured light lasers [46, 47], visual cameras [48], IMU [49], and 3D sensors [50, 51]. These have been surveyed in [52, 53]

Although this thesis focuses on formal state estimation techniques, the vast majority of deployed industrial in-pipe robots do not perform formal localization and mapping. Instead, localization is often approximated through simplified assumptions. It is common for inspection robots to assume that the pipe is linear, using encoder odometers to track the distance traveled along the pipe [6, 54]. This technique, known as **linear referencing**, associates each sensor measurement (e.g., visual frame or laser scan) with a position on a one-dimensional path based on its timestamp [20, 21, 47]. While this method is simple and inexpensive, it fails to capture important deviations such as curves, branches, elevation changes, and encoder drift, limiting its accuracy in long or complex networks. Furthermore, such assumptions prevent accurate 3D reconstruction, which is increasingly important for predictive maintenance, deformation analysis, and robot re-navigation.

2.2.1 Challenges in Confined-Space Pipe Environments

Localization and mapping in confined spaces like underground pipes present distinct challenges compared to conventional SLAM scenarios. These arise from the physical constraints, limited sensing capabilities, and surface characteristics within pipes.

1. **Geometric Simplicity**: Pipelines typically have smooth, cylindrical geometries with few distinctive features such as edges or planes. In perfectly cylindrical environments, translation along the pipe axis and rotation around the pipe axis are weakly constrained when using 3D sensing modalities like LiDAR, making

geometry-based SLAM methods, such as LOAM [8] and FAST-LIO [9], less effective.

2. **Limited Visual Features**: Many pipelines also lack visually distinctive textures, presenting smooth, repetitive, or reflective surfaces that challenge feature-based SLAM methods. Techniques such as ORB [55] and SIFT [56] often fail to extract stable keypoints, limiting reliable visual tracking and optimization. Although abundant texture can sometimes mitigate geometric simplicity, many pipe environments exhibit both challenges simultaneously.
3. **Scale and Accessibility Constraints**: Pipes may be only a few centimeters in diameter, severely limiting the sensor size and the available sensing field of view. Narrow geometries constrain what can be observed at any given time, leading to partial observations (e.g., truncated point clouds) and reduced spatial context for accurate mapping and localization.
4. **High Sensor Noise and Motion Dynamics**: Pipeline robots often experience jerky motion from friction or irregular surfaces, leading to noisy IMU data and degraded visual tracking. In narrow pipes, even moderate speeds can cause motion blur and parallax effects, compounding these issues.
5. **Lack of Global Localization Cues**: Being GPS-denied, underground pipelines offer no external localization references. This forces reliance on relative measurements, which can accumulate unbounded drift over long distances.
6. **Environmental Hazards and Constraints**: Blockages, water, and non-uniform pipe structures (e.g., joints, bends, or deformations) obstruct sensors and challenge the adaptability of SLAM systems.
7. **Computational and Energy Constraints**: Robots must be compact and power-efficient, limiting onboard compute resources. Restricted processing power and IO bandwidth constrain real-time processing and algorithm complexity.
8. **Dark, Wet, and Reflective Conditions**: Poorly lit pipes require onboard lighting, which may cause inconsistent illumination in visual data. Water and wet surfaces introduce vision distortions through refraction and reflections.

These challenges necessitate specialized sensing strategies, robust sensor fusion, and algorithms tailored to confined, feature-sparse environments.

2.2.2 In-Pipe Localization and Mapping

In contrast to industrial practices, academic research has explored more principled approaches to in-pipe localization and mapping, as surveyed in [57]. These methods generally fall into two categories based on whether the environment structure is estimated jointly with the trajectory: **SLAM-based approaches**, which jointly estimate robot trajectory and 3D environment structure, and **odometry-only approaches**, which estimate the trajectory alone and associate sensor data to the estimated path for post-hoc analysis.

When applying full SLAM in pipe environments, significant challenges arise due to the lack of salient and distinguishable features. As discussed in Section 2.2.1, the repetitive and smooth geometry of pipes limits the performance of feature-based methods. LiDAR odometry suffers from geometric simplicity, while visual SLAM systems often experience scale drift due to weak textures and limited depth cues. To address these issues, some vision-based methods [58, 59] incorporate geometric priors, such as fixed-radius assumptions or known models for straight pipe sections and junctions. While these constraints help resolve scale ambiguity, they limit generalization to pipes with consistent, known geometries. An alternative approach by [60] employs active laser profiling to directly measure local depth, thereby improving metric accuracy and reducing drift.

Our prior work, VLI-SLAM [61], tackles the scale ambiguity challenge by fusing structured-light laser data with visual-inertial odometry. Building on [62] for laser line extraction and [7] for tightly coupled visual-inertial SLAM, this method is agnostic to the pipe's geometric form and enables dense, colorized reconstructions. Although it achieves robust tracking in short-range scenarios, performance degrades over long pipe runs due to the absence of global localization cues, leading to cumulative drift.

To mitigate cumulative drift, several works have explored the use of additional structural or semantic cues. For example, [63] integrates identifiable pipe features—such as junctions and corners—as pose graph constraints for trajectory refinement. Other methods explicitly incorporate cylindrical shape priors to reduce drift. In [64], ORB-SLAM2 [10] is extended with a pose graph optimization step that penalizes deviation from an idealized pipe axis. While effective in reducing drift, this method assumes straight, uniformly shaped pipes, an oversimplification that may not hold in real-world networks.

Odometry-only methods are often preferred when visual or LiDAR sensing is unreliable, particularly in narrow, featureless, or highly reflective pipes. Many of these rely on filtering-based sensor fusion techniques. For instance, [49] uses an EKF to fuse wheel encoder data and IMU measurements for pose estimation in PIG systems, but the absence of global reference makes this methods susceptible to long-term drift. In [41], encoder data is obtained from the tether cable rather than from onboard encoders. A related method in [65] combines encoder and IMU fusion with visual tracking of a sponge element placed in front of the camera to estimate heading. While effective in small-diameter (15 mm) pipes, this approach is highly dependent on external hardware modifications and lacks general applicability.

Chapter 3

VILL-SLAM: Localization and Dense RGB-D Reconstruction for Pipe Environments

In this chapter, we present VILL-SLAM, a dense RGB-D SLAM framework that integrates a monocular camera (V), an inertial measurement unit (I), a ring-shaped laser profiler (L), and a LiDAR sensor (L) into a compact sensor package optimized for in-pipe operations. By fusing complementary visual and depth information from the camera, laser profilometry, and LiDAR measurements, the system addresses the challenges of metric scale estimation inherent in conventional monocular SLAM methods. To further enhance localization accuracy, the known cylindrical geometry of pipes is exploited to formulate two additional factors within a factor graph framework, providing strong geometric constraints to mitigate odometry drift. The proposed method is evaluated through experiments on real pipe samples and benchmarked against existing state-of-the-art algorithms. Compared to our research team's previous system without LiDAR integration, VILL-SLAM achieves a 6.6-fold reduction in odometry drift, with a mean drift of 0.84% over a 22-meter trajectory and a mean pointwise 3D scanning error of 0.88 millimeters in 12-inch diameter pipes. Field deployments in operational pipeline networks further demonstrate the practical utility of the proposed multi-modal sensor fusion approach for inspection and anomaly detection tasks.

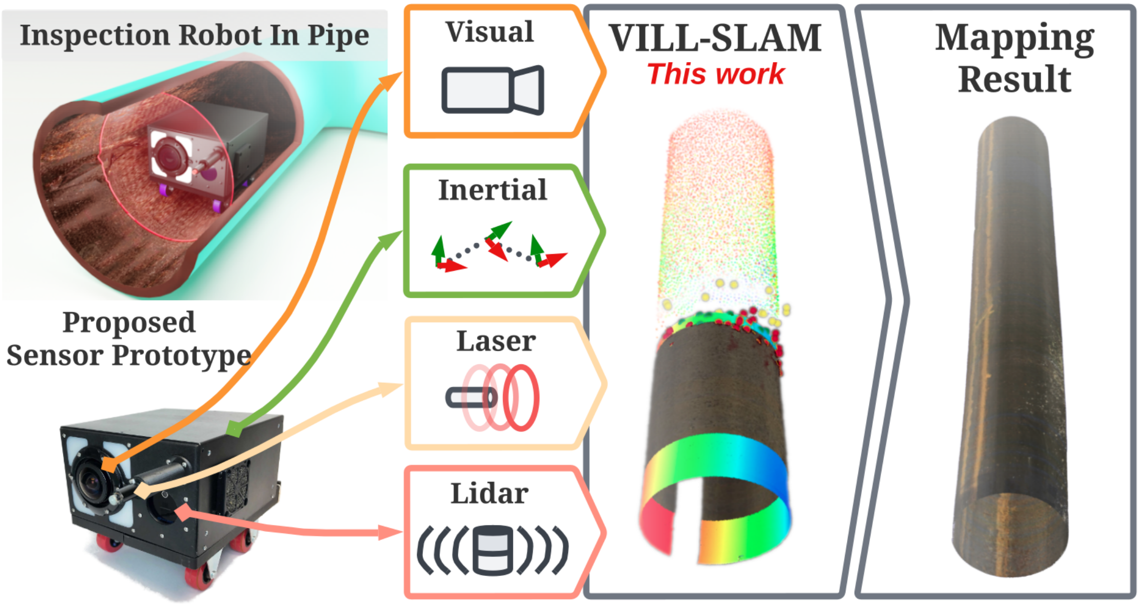

Figure 3.1: The in-pipe mapping sensor hardware prototype using the proposed VILL-SLAM method to fuse multiple sensory information, which produces a photo-realistic dense RGB-D reconstruction of a 12-inch pipe segment's inner surface.

3.1 System Overview

3.1.1 Hardware System Overview

Our proposed sensor suite design (Fig. 3.1) consists of an RGB CMOS camera with a fisheye lens, a MEMS-based 6-axis IMU, a laser projector, and a Realsense L515 LiDAR. The laser projector projects a laser plane orthogonal to the camera's optical axis by emitting a thin laser beam towards a conic mirror, and the laser plane forms a ring on the pipe's inner surface. We adapt the alternating-shutter technique described in [61] to strobe the red laser stripe and the illumination LED in synchronization with the image shutter trigger (Fig. 3.2). This way, we effectively capture the visual frames $\mathcal{I}_v$ containing visual details and the profiling frames $\mathcal{I}_p$ containing a bright laser ring with minimal time gaps. The streaming rate of the visual-laser frame pairs is 30 Hz. We perform camera intrinsic calibration with [66], camera-laser calibration with [62], camera-IMU calibration with [67], and camera-LiDAR calibration using MATLAB's LiDAR Toolbox. The proposed sensor package is compact and attachable to existing actuated in-pipe crawler robots.

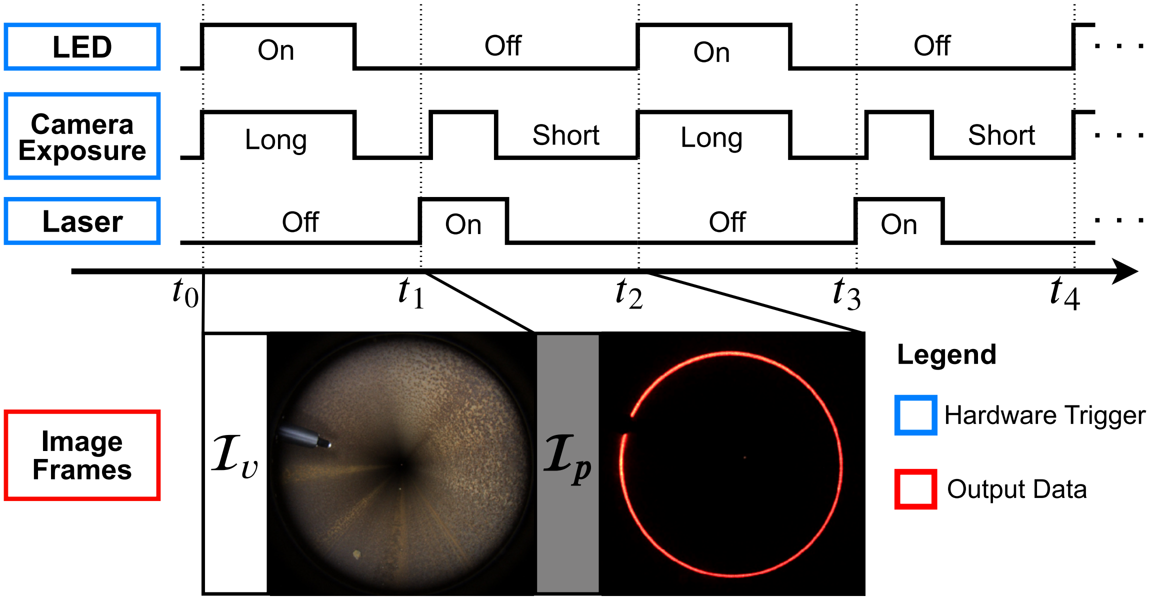

Figure 3.2: We capture visual and profiling image frames using one camera by quickly switching between visual and profiling frames using a microcontroller, measuring RGB and depth information in a near-concurrent fashion.

3.1.2 Software System Overview

The software architecture diagram is illustrated in Fig. 3.3. After acquiring data from each physical sensor, a series of front-end preprocessing processes formulate data frames by tracking the visual features, preintegrating IMU measurements [31], triangulating the laser profiler data, and fitting cylinder primitives using the LiDAR point cloud. Then the software associates the 2D visual features with depths from the triangulated 3D laser points to bootstrap the SLAM process. With the preprocessed sensory data from the front-end, a sliding-window-based nonlinear optimization process is activated to estimate robot odometry and visual feature depths, where the constraints in the cost function are structured with a factor graph. Using the odometry determined from the state estimator, the 3D laser scans are registered into a colored point cloud map. We also feed the odometry back to register a local LiDAR map, which provides global depth information during the visual-depth association.

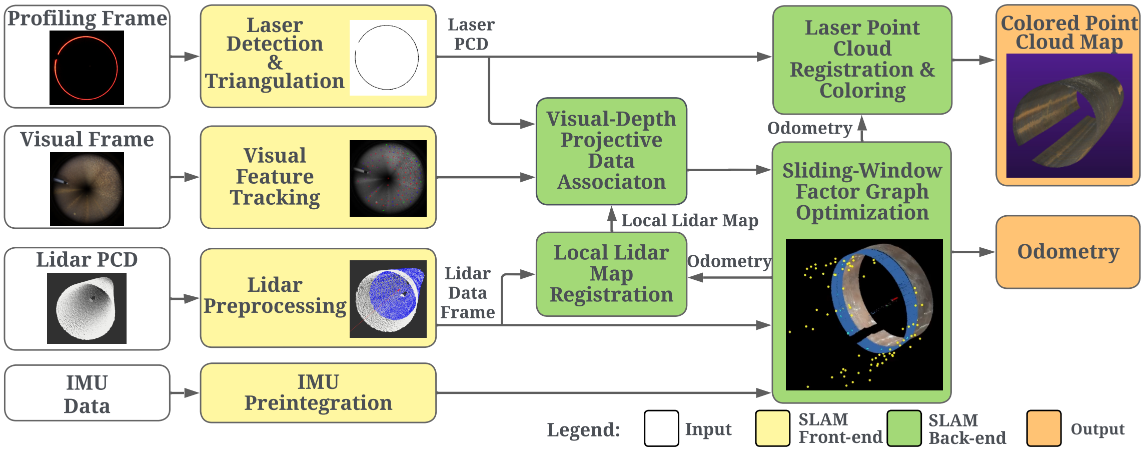

Figure 3.3: VILL-SLAM software architecture and information flow chart.

3.2 Visual-Inertial-Laser-LiDAR SLAM

At the core of this method, is a sliding-window-based nonlinear optimization that performs state estimation using visual, inertial, and laser-and-lidar-induced depth measurements. It also incorporates both the pipe's cylindrical structure and available geometric features to reduce odometry drift. Finally, the output is a dense RGB-D scan of the pipe interior and the sensor package's 6-DoF odometry.

3.2.1 Sensor Data Preprocessing

The initial stage comprises the pre-processing of raw data obtained from the various sensors. Each data stream is processed independently, and is outlined as follows:

1) Laser detection and triangulation: Given the calibrated laser plane parameters with respect to the camera frame, we detect and triangulate the laser points through HSV thresholding in the profiling frames to find the corresponding 3D laser point for each 2D laser pixel following the method described in [61]. In practice, the laser can be robustly extracted even when ambient light is captured in the profiling frames, for example, near pipe entrances and exits.

2) Visual feature tracking: This is done through the Shi-Tomasi corner detector [68] and KLT optical flow [7]. Also, the keyframes are selected in this step. We treat an incoming frame as a new keyframe if the average parallax of tracked features between this frame and the latest keyframe is greater than a certain threshold.

3) IMU preintegration: This is implemented according to [7]. As mentioned in Section 2.1.4, preintegration summarizes raw IMU measurements between keyframes into a compact relative motion constraint, avoiding repeated integration during later optimization.

4) LiDAR preprocessing: LiDAR point clouds streaming at 10 Hz are cropped to a region of interest between 0.4 m and 6 m from the sensor, spatially downsampled by a factor of 20, and transformed to the body frame, where the body frame is aligned to the IMU frame. If the remaining points are sufficient, we then attempt to fit a cylinder to the pipe point cloud using RANSAC-based cylinder fitting [69], regardless of whether the captured point cloud is actually cylindrical or not. The output of LiDAR preprocessing is a LiDAR data frame, which contains the preprocessed LiDAR point cloud $\mathcal{L}$, the set of points-on-cylinder $\mathcal{Y} \subseteq \mathcal{L}$, and the corresponding cylinder parameters if the cylinder is found, including the normalized axis vector $\mathbf{u}$, an axis point $\mathbf{p}$, and the radius r.

3.2.2 Visual-Depth Association

After the preprocessing step, the visual features are associated with the corresponding depth data from laser profiling and LiDAR. Denote the set of visual features as $\mathcal{F}$. A feature is defined to be a feature-on-laser if any of its observations is close to the laser pixels in adjacent $\mathcal{I}_p$. Denote the set of features-on-laser as $\mathcal{F}_l$. Similarly, a feature $f \notin \mathcal{F}_l$ is a feature-on-LiDAR if any of its observations is close to a point in the local LiDAR map projected to $\mathcal{I}_v$, and the set of features-on-LiDAR is denoted as $\mathcal{F}_L$. For a feature $f_i \in \mathcal{F}_l \cup \mathcal{F}_L$, the observation frame in which the feature pixel is the closest to the laser pixel or the projected LiDAR pixel is defined as its primary observation frame c_i^*. If $f_i \notin \mathcal{F}_l \cup \mathcal{F}_L$, its c_i^* is the first observation frame. Note that if a feature $f_i \in \mathcal{F}_l \cup \mathcal{F}_L$, there exists a depth association between the 2D feature and 3D data. Denote its associated depth at c_i^* as $\bar{d}_i$.

3.2.3 Estimator Initialization

To bootstrap the SLAM process, a vision-only structure from motion (SfM) is performed to obtain the up-to-scale camera poses and feature positions, followed by visual-inertial alignment, which aligns metric IMU pre- integration with the SfM results [7]. This way, we can obtain a rough gyroscope bias estimation $\hat{\mathbf{b}}_g$ and a scale estimation $\hat{s}$. Here, the scale accounts for the unknown depth scaling factor of visual features inherent to monocular SfM. At this stage, the gyroscope bias is prone to error due to insufficient rotational excitation, and the scale estimation can be a few orders of magnitude larger than the true scale in an in-pipe scenario. A fine-tuned scale estimation, $\bar{s}$, is computed by multiplying the rough scale by a factor determined by the associated depth $\bar{d}_i$ of each visual feature $f_i \in \mathcal{F}_l \cup \mathcal{F}_L$:

$$\bar{s} = \alpha \frac{1}{|\mathcal{F}_l|} \sum_{f_i \in \mathcal{F}_l} \frac{\bar{d}_i}{\hat{d}_i} \hat{s} + (1 - \alpha) \frac{1}{|\mathcal{F}_L|} \sum_{f_i \in \mathcal{F}_L} \frac{\bar{d}_i}{\hat{d}_i} \hat{s}, \tag{3.1}$$

where $|\cdot|$ denotes the cardinality of a set, $\hat{d}_i$ is the estimated depth of each feature from triangulation, and α balances the contribution of laser and LiDAR depths to the scale correction and it is set to $|\mathcal{F}_L|/(|\mathcal{F}_l| + |\mathcal{F}_L|)$ if $|\mathcal{F}_L| > 0$ or 1 otherwise.

Using the new scale, we correct each keyframe's position, velocity, and feature positions estimated during visual-inertial alignment and conclude the initialization process. Note that we do not fine-tune the gyroscope bias estimation here since it will be further corrected online during the sliding-window-based optimization.

3.2.4 Sliding-Window-Based Factor Graph Optimization

After estimator initialization, we proceed with a sliding window-based tightly-coupled monocular VIO. The full state vector, consisting of robot states $\mathbf{x}$ and landmark states λ in the sliding window, is defined as:

$$\begin{aligned} \mathcal{X} &= [\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_n, \lambda_0, \lambda_1, ..., \lambda_m] \\ \mathbf{x}_k &= [\mathbf{T}^w_{b_k}, \mathbf{v}^w_{b_k}, \mathbf{b}_a, \mathbf{b}_g], \end{aligned} \tag{3.2}$$

where n, m are the total number of keyframes and visual features in the sliding window, respectively. λ_i is the inverse feature depth of f_i in its primary observation frame c^*_i. $\mathbf{b}_a$ and $\mathbf{b}_g$ are the accelerometer and gyro biases [7], $\mathbf{T}^w_{b_k}$ is the pose of the k^{th} body frame with respect to the world frame.

We perform maximum *a posteriori* estimation of the states $\mathcal{X}$ by minimizing the weighted sum of four factors in a factor graph, alongside the pose prior obtained from the last marginalization [7], as shown in Fig. 3.4.

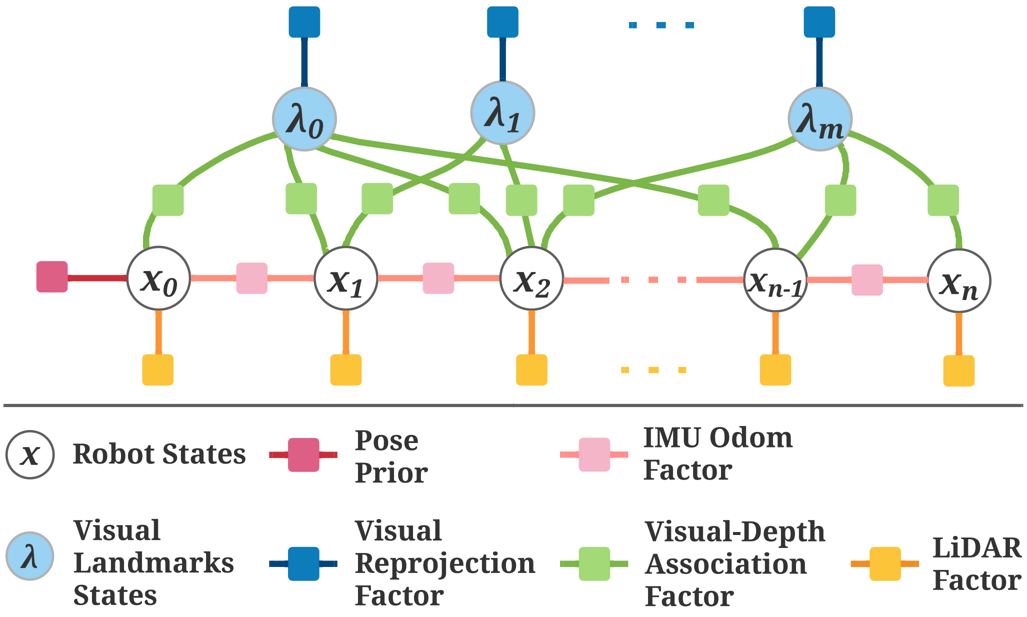
λ0
λ1
λm
x0
x1
x2
xn-1
xn
x Robot States
Pose Prior
IMU Odom Factor
λ Visual Landmarks States
Visual Reprojection Factor
Visual-Depth Association Factor
LiDAR Factor

Figure 3.4: At the core of our method, a factor graph is used to represent various states with their related constraints within a sliding window.

Visual Reprojection Factor: This factor measures the re-projection error of visual features between keyframes. Denote the set of visual frames in the sliding window as $\mathcal{C}$. For every feature f_i, we project its estimated 3D coordinates to keyframe $c_j \in \mathcal{C}$ and compute the 2D pixel difference with the observation in its primary observation frame c_i^*:

$$e_c = \sum_{i\in\mathcal{F}} \sum_{j\in\mathcal{C}} \left\| \pi_c(\mathbf{T}_w^{c_j}\mathbf{T}_{c_i^*}^{w}\frac{1}{\lambda_i}\pi_c^{-1}(\begin{bmatrix} u_i^{c_i^*} \\ v_i^{c_i^*} \end{bmatrix})) - \begin{bmatrix} u_i^{c_j} \\ v_i^{c_j} \end{bmatrix} \right\|^2, \tag{3.3}$$

where $\pi_c(\cdot)$ projects a 3D point onto the 2D image and $\pi_c^{-1}(\cdot)$ back-projects a pixel onto the normalized image plane.

Visual-Depth Association Factor: This factor leverages the depth association between visual features and laser or LiDAR information to further constrain feature depth estimation. For a feature $f_i \in \mathcal{F}_l \cup \mathcal{F}_L$, we compute the residual between the associated depth $\bar{d}_i$ from projective data association and the estimated feature depth:

$$e_d = \sum_{f_i \in \mathcal{F}_l \cup \mathcal{F}_L} ||\frac{1}{\lambda_i} - \bar{d}_i||^2 \tag{3.4}$$

IMU Odometry Factor: We follow the IMU residual definition in [7] to estimate $\mathbf{v}_{b_k}^w$, $\mathbf{b}_a$, $\mathbf{b}_g$, and $\mathbf{T}_{b_k}^w$. The IMU factor is constructed from the preintegrated measurements between keyframes and is used to assist state estimation in low-feature regions [58].

LiDAR Factor: This factor is used to constrain the odometry estimation by aligning LiDAR frames through LiDAR point cloud matching. This alignment is performed between each LiDAR frame $\mathcal{L}_k$ in the current sliding window and a reference LiDAR frame $\mathcal{L}_{ref}$ captured prior to the window and updated periodically with a fixed update period τ_{ref}, and the pose $\mathbf{T}_{b_{ref}}$ of the body frame b_{ref} in which $\mathcal{L}_{ref}$ was captured is known.

We define two candidate LiDAR factors: 1) the LiDAR cylinder factor, inspired by [64], for long and straight pipe segments and 2) the LiDAR iterative-closest-point (ICP) factor when the sensor package approaching geometrically-diverse environments such as pipe with dents or protrusions, fittings, branches, or deformed areas. Depending on the environmental characteristics, the algorithm selects one LiDAR factor from these two candidates. The criterion for the selection is termed as the cylindrical structure regularity $R_{\mathcal{Y}}$, which is the ratio between the number of points-on-cylinder $\mathcal{Y}$ and the total number of points in the LiDAR point cloud $\mathcal{L}$ in this frame. The larger $R_{\mathcal{Y}}$ is, the more cylindrical the environment is:

$$R_{\mathcal{Y}} = \begin{cases} |\mathcal{Y}|/|\mathcal{L}|, & \text{if } |\mathcal{L}| > 0 \\ 0, & \text{otherwise.} \end{cases} \tag{3.5}$$

1) LiDAR Cylinder Factor is selected if $R_{\mathcal{Y}}$ is greater than a threshold $H_{\mathcal{Y}}$ for both $\mathcal{L}_k$ and $\mathcal{L}_{ref}$, which means that the environment is almost perfectly cylindrical. The key idea is that for two sets of points-on-cylinder, if they capture the same physical cylindrical environment, all points are equidistant from one single cylinder axis. This constraint is mathematically formulated as:

$$e_{\mathcal{Y}} = \sum_{k=0}^{n} \sum_{\mathbf{q}_i \in \mathcal{Y}_k} \left\| \left(\mathbf{T}^{w}_{b_{ref}} \mathbf{u}^{b_{ref}} \right) \times \left(\mathbf{T}^{w}_{b_k} \mathbf{q}_i^{b_k} - \mathbf{T}^{w}_{b_{ref}} \mathbf{p}^{b_{ref}} \right) \right\| - \left\| \mathbf{T}^{w}_{b_{ref}} \mathbf{u}^{b_{ref}} \right\| \cdot r, \tag{3.6}$$

where $\mathcal{Y}_k \in \mathcal{L}_k$ and $\mathcal{Y}_{ref} \in \mathcal{L}_{ref}$. These points-on-cylinder are assumed to lie on the same physical cylinder within the update period τ_{ref}, which we empirically set to 10 s based on the maximum curvature along the pipe and the robot's speed. $\mathbf{q}_i$ is any point in $\mathcal{Y}_k$. $\mathbf{u}^{b_k}$, $\mathbf{p}^{b_k}$ and $\mathbf{u}^{b_{ref}}$, $\mathbf{p}^{b_{ref}}$ are the cylinder axis vectors and axis points in frame b_k and b_{ref}, respectively, and r is cylinder radius (Fig. 3.5). Although the two degrees of freedom about the axial direction of the pipe are underconstrained with this cylinder factor alone, the pose is fully constrained when jointly optimized with other factors.

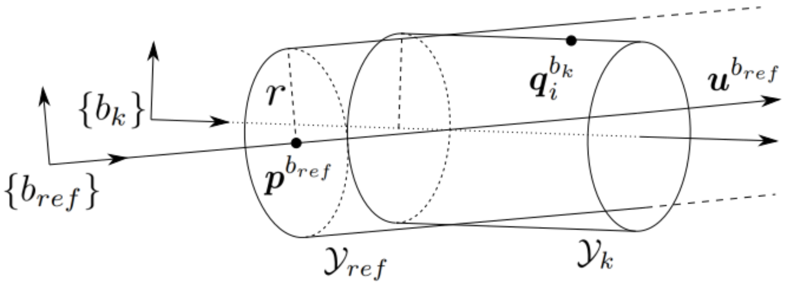

Figure 3.5: LiDAR cylinder factor.

2) LiDAR ICP Factor is selected if $R_{\mathcal{Y}} \in (0, H_{\mathcal{Y}}]$ for both $\mathcal{L}_k$ and $\mathcal{L}_{ref}$. First, point-to-point ICP [70] is performed on $\mathcal{L}_k$ and $\mathcal{L}_{ref}$ to find the matching point pairs in the LiDAR point clouds. Then, we construct the residual by transforming the corresponding points into the world frame and computing the distance between each pair of correspondences:

$$e_{icp} = \sum_{\mathbf{q}_i \in \mathcal{L}_k \cap \mathcal{L}_{ref}} \left\| \mathbf{T}^{w}_{b_k} \mathbf{q}_i^{b_k} - \mathbf{T}^{w}_{b_{ref}} \mathbf{q}_i^{b_{ref}} \right\|^2 \tag{3.7}$$

Note that the ICP can be completed without ambiguity because the threshold check on the cylindrical structure regularity ensures that sufficient geometric features exist to perform the alignment.

3.2.5 Map Registration

With the odometry from the optimization, we register two maps in the world frame: 1) a dense and colored map of the laser scans and 2) a sparse map of LiDAR point clouds. The laser map is used as the final output of the SLAM and is generated in a similar fashion as described in [61], where the color of each laser scan in a profiling frame is estimated using adjacent visual frames. The LiDAR map is an intermediate product needed by the visual-depth association step. To ensure bounded memory usage, only the LiDAR frames captured in the last update period are kept in the LiDAR map, and the points are downsampled using a voxel grid filter [69]. The LiDAR map allows the visual-depth association between a current visual frame and historical LiDAR data, which is desirable in long narrow pipes.

3.3 Evaluation Experiments

Experiments are conducted to functionally validate our method's dense RGB-D mapping capability and verify that it is able to perform low-drift localization and accurate 3D reconstruction. First, we visually compare the mapping quality of our method against the state-of-the-art SLAM methods. Next, we conduct two experiments to quantitatively analyze the localization and mapping performance. All our experiments are performed in 12-inch diameter, empty metal pipe samples that are placed above the ground. No additional lighting besides the LED on the sensor prototype and no fiducials are placed in the pipe.

3.3.1 Qualitative Analysis

We visually compared the mapping result of our VILL-SLAM to various state-of-the-art single or multi-sensor SLAM algorithms, including FAST-LIO2 [9], ORB-SLAM2 [10], LOAM [8], VINS-Mono [7] and dense mapping methods like SSL-SLAM2 [11] and RGBDTAM [12]. Among the evaluated methods, all except our methods and VINS-Mono fail to localize and generate a meaningful map of the pipe. Although VINS-Mono is able to generate a map of the pipe, the map is sparse, noisy, and inaccurate due to the incorrect scale estimation in pipes and the lack of dense mapping functionality. In

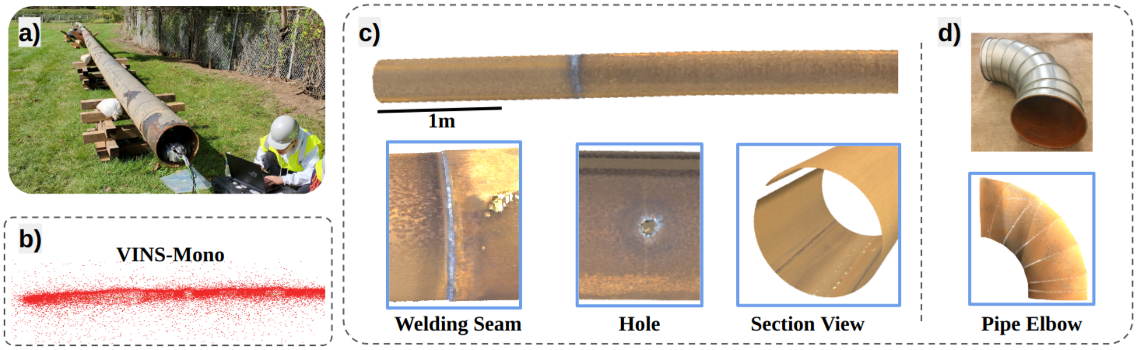

Figure 3.6: Test environment and mapped point cloud visualization. (a) Test site with a total pipe length of 50m. (b) Mapping results from VINS-Mono [7]. Note that other tested methods including LOAM [8], FAST-LIO2 [9], ORB-SLAM2 [10] and dense mapping pipelines like SSL-SLAM2 [11] and RGBDTAM [12] all failed to produce reasonable results under same testing conditions. (c) Dense colorized 3D map created by VILL-SLAM (ours) in the pipe test site and the zoom-in views of some regions of interest. (d) 3D map created by VILL-SLAM in a pipe elbow sample.

contrast, our VILL-SLAM is able to generate denser and smoother maps (Fig. 3.6c) thanks to the accurate metric scale estimation through visual-depth association with laser and LiDAR. Our output map is also colorized using the alternating-shutter laser profiling method (Section 3.1.1). In Fig. 3.6d, we also include a mapping result of a 90-degree metal duct, whose inner surface is painted to prevent reflection. The photo-realistic point cloud maps of the pipe interior built by VILL-SLAM contain both geometric and visual details of the pipe, which is valuable for downstream surface defect or pipe geometry analysis.

3.3.2 Localization Accuracy Evaluation

We collect the ground truth trajectory with a Leica total station (Leica Geosystems AG, Heerbrugg, Switzerland) to track a prism mounted on the top of the sensor package. Since the estimated and the true trajectories are not in the same coordinate frame, we first find the transformation between the world frame set by the state estimator and the total station's frame by aligning the first ten meters of the trajectories using a closed-form method described in [71]. Subsequently, the two trajectories are transformed into the same coordinate frame, and the distance between every two matched points is calculated. We compute the absolute trajectory error (ATE) over the trajectory length, which is clipped to 22 m.

We compare the localization accuracy of VILL-SLAM against VINS-Mono and VLI-SLAM. Table 3.1 shows the performance statistics across six trials, where drift is defined as the maximum error over trajectory length. The mean odometry drift of VILL-SLAM is 0.84% and the mean RMSE is 7.62 cm, which is 6.6 times drift improvement and 2.8 times RMSE improvement compared to VLI-SLAM. VILL-SLAM also shows a smaller error variance. From this experiment, we verify that our proposed method is able to achieve low drift even over long distances in pipes. On the testing PC with AMD Ryzen 5900x CPU, the average mapping frame rate is 27 fps, which is sufficient for real-time mapping. An example of the ATE of VLI-SLAM and VILL-SLAM is shown in Fig. 3.7. The plot for VINS-Mono is not included because the error is too large compared to the rest.

Table 3.1: Mean and Standard Deviation of Localization Errors

Metrics	VILL-SLAM (our current)	VLI-SLAM (our prior)	VINS-Mono
RMSE (cm)	**7.62 ± 4.38**	21.67 ± 10.07	1285.71 ± 251.94
Max (cm)	**18.46 ± 7.73**	121.78 ± 40.08	4205.16 ± 452.07
Drift (%)	**0.84 ± 0.35**	5.53 ± 1.82	189.34 ± 20.55

3.3.3 3D Reconstruction Evaluation

As shown in Fig. 3.8, we first use a high-end 3D printer to print custom evaluation jigs, whose ground-truth point clouds are exported from the CAD software. Since the 3D printer has relatively high printing accuracy, the printing error is considered negligible. After installing the evaluation jig onto the pipe's inner surface, we scan the pipe and the jig using the sensor prototype. Finally, the point cloud map from SLAM and the ground-truth point cloud are aligned using ICP, and the point-to-point L2 distance error is computed. Across 4 trials using 2 jig designs, the average error is 0.88 mm per point, indicating that our scanner can produce maps with sub-millimeter grade local scanning accuracy.

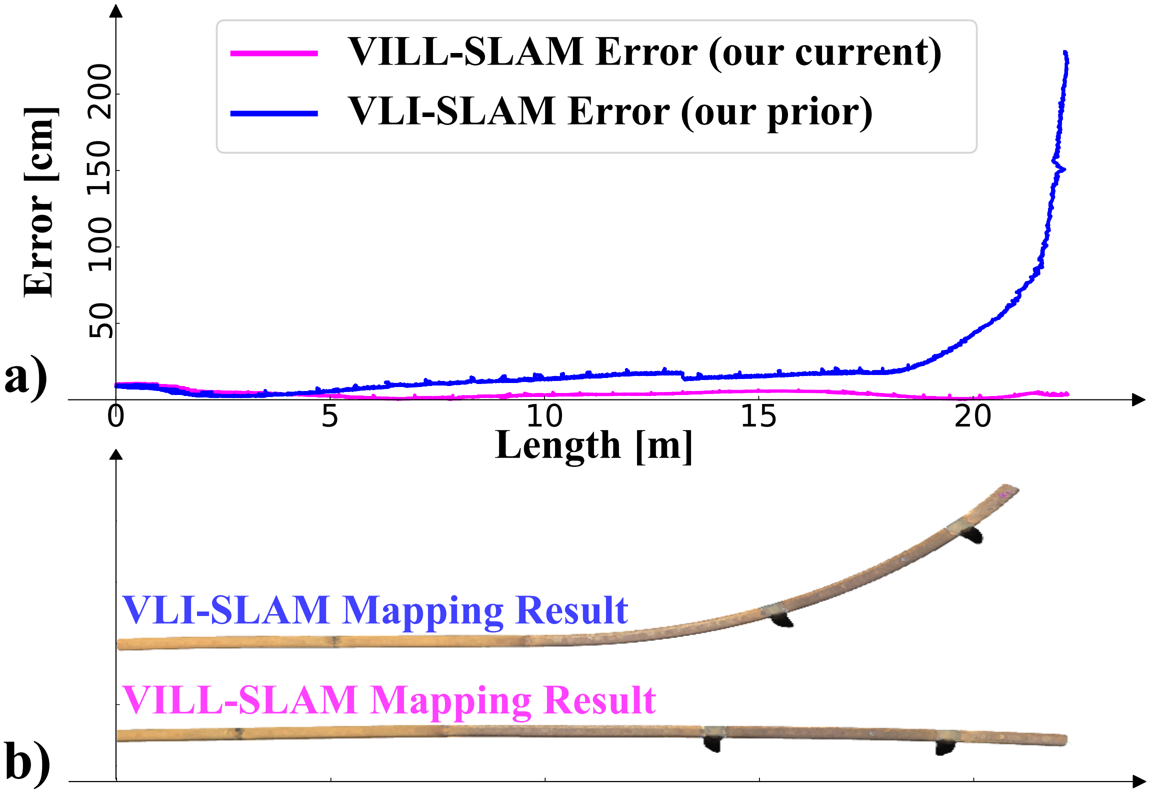

Figure 3.7: Example ATE plots of VLI-SLAM and VILL-SLAM. (a) Visualization of the ATE of our current work VILL-SLAM (with LiDAR) and our prior method VLI-SLAM (without LiDAR) using data collected in one trial. (b) The corresponding 3D maps of the pipe using the two methods.

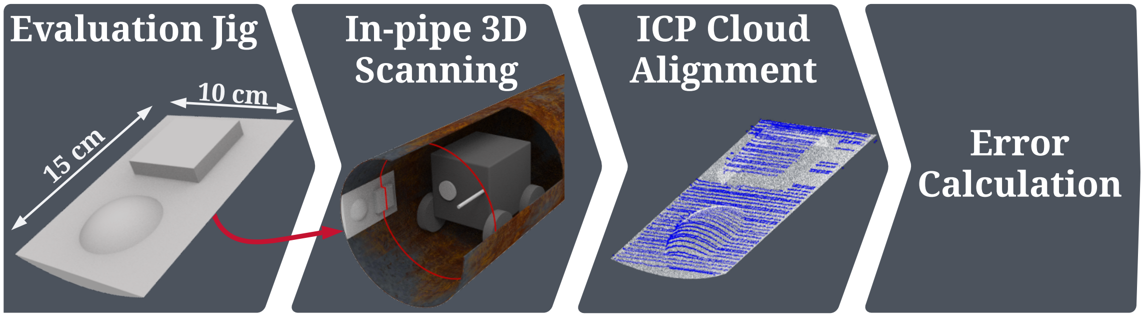

Figure 3.8: Workflow of 3D reconstruction accuracy assessment using an evaluation jig and in-pipe 3D scanning experiments.

3.3.4 Simulation

Based on the visual and texture data captured in the physical pipes in the mock-up test sites, we develop a simulated pipe environment in Epic Games' Unreal Engine 4 (UE4), replicating the conditions of the physical samples. Compared to other robotic simulators such as Gazebo [72] and Mujoco [73], UE4 provides superior graphics and texture rendering, allowing for a more photorealistic simulation of in-pipe environments. The simulation comprises sensor suite simulation and piping system generation, which is achieved by combining basic pipe units and procedural random defect creation. First, we construct basic pipe segments using Blender 4 (Blender Foundation, Amsterdam, Netherlands) and subsequently created random defects using geometry nodes. Next, procedural controls handle model selection, positioning, and connection of segments. A movable "virtual" robot is centered in the pipe for data collection. The simulated sensor suite and data capturing schemes also match the real counterpart.

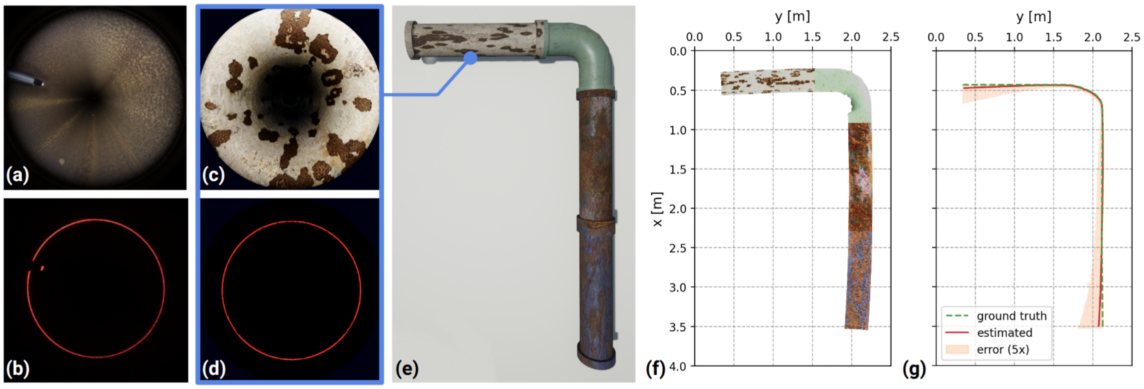

Figure 3.9: In-pipe sensing simulation in UE4. (a,b) Examples of visual and laser profiling frames captured inside a real pipe. (c,d) Corresponding visual and laser profiling frames captured inside a simulated bendy pipe. (e) The simulated pipe environment where frames (c) and (d) are captured. (f) 3D point cloud reconstruction of the simulated pipe interior. (g) Estimated path (red solid curve) versus true path (green dashed curve) within the simulated pipe, with the error (magnified 5 times for visualization) shaded in orange.

The simulation facilitates the verification of the robot's sensing and SLAM software, enabling the testing of various pipeline topographies, such as curved pipes, without the need to rely solely on physical trials. Fig. 3.9 illustrates a simulated 12-inch-diameter curved pipe, alongside the images captured within the pipe and the corresponding localization and mapping results. The localization root-mean-square error (RMSE) absolute trajectory error (ATE) is 0.027 m, reflecting a drift of 0.55% over a 4.886-meter trajectory.

3.4 Field Deployment Outcomes in Natural Gas Pipes

We deployed the multi-sensor payload (described in Section 3.1.1) in an underground 12-inch natural gas distribution system in Louisiana, USA. The century-old cast-iron and steel pipes in this site was decommissioned following Hurricane Katrina. The pipe owners had minimal available documentation about the pipes, which were limited to outdated paper blueprints from a hundred years ago.

We performed a total of six trials in two pipes. During each trial, the robot collected multi-modal data, including camera imagery, 2D laser profiles, and inertial measurements. Two GPS coordinates were recorded to anchor the reconstructed trajectory in geodetic space: one point $\mathbf{p}_0^{\text{GPS}}$ was at start of the pipe, and another point $\mathbf{p}_1^{\text{GPS}}$ along the pipe's axial direction, such as the end point of the pipe. Real-time data display included encoder-based distance tracking and camera streams, and laser-based out-of-roundness (OOR) analysis per American Petroleum Institute (API) Standard 579 [74].

Laser Profile Analysis: Laser pixels were triangulated and turned into cross-sectional profiles in the form of 3D points (as described in Section 3.2.1). Each profile was then fitted with a best-fitting circle [75], and the per-laser-point OOR was calculated as the percentage deviation from the nominal radius.

Data Post-Processing: After SLAM execution, the estimated trajectory and the 3D reconstruction were transformed from the relative coordinates into absolute GPS coordinates, by aligning the starting point of the trajectory with the first GPS marker $\mathbf{p}_0^{\text{GPS}}$ and aligning the trajectory's direction with the vector $\mathbf{p}_1^{\text{GPS}} - \mathbf{p}_0^{\text{GPS}}$. Each image frame was tagged with traveled distance, OOR metrics, and estimated GPS coordinates.

Deployment Insights: Despite one trial lost to camera fogging in high humidity,

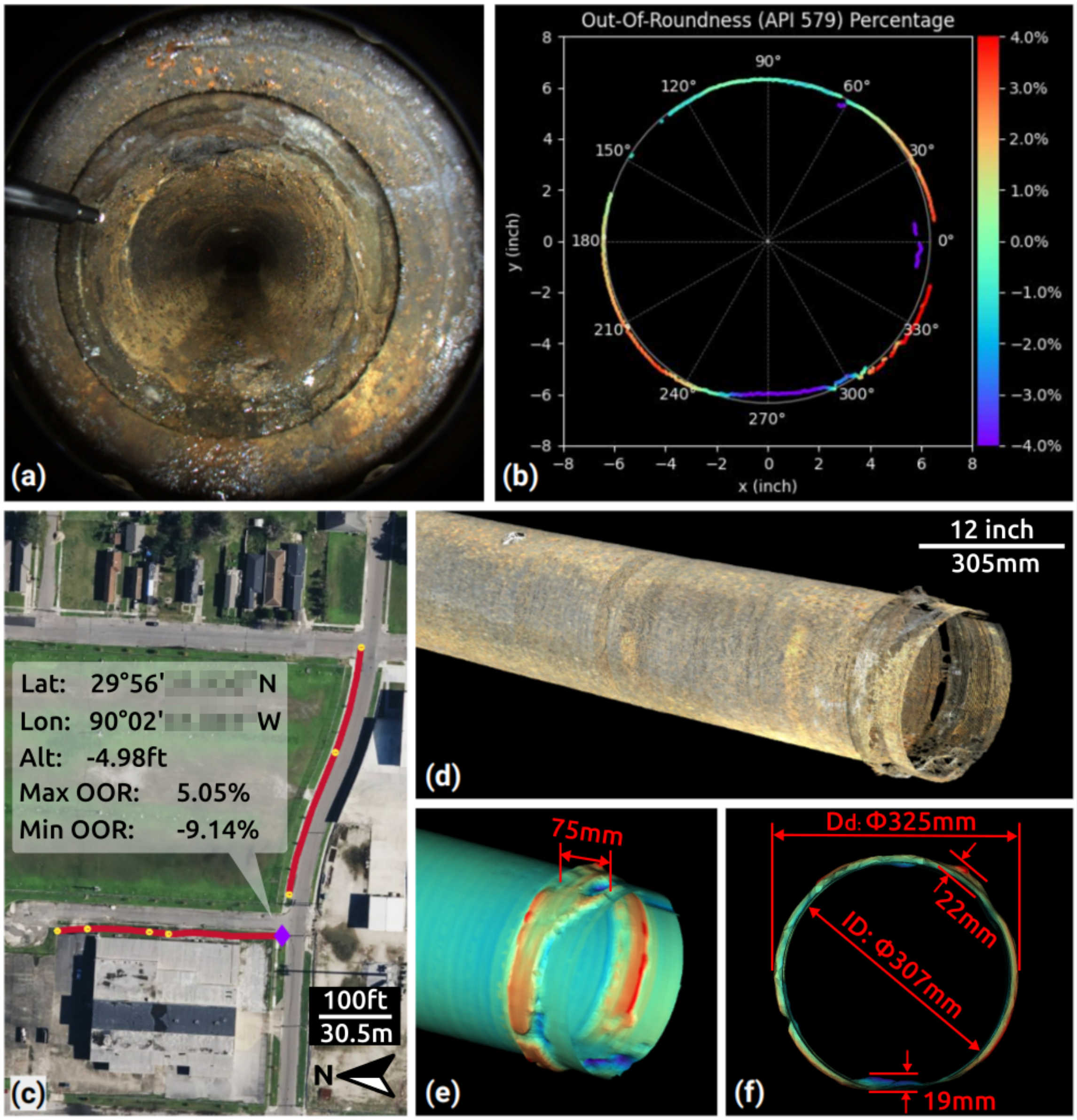

Figure 3.10: Data captured in a field deployment trial near a pipe dislocation spot. (a) Visual frame showing a gap, rust, and signs of water. (b) Out-of-roundness (OOR) plot of the corresponding laser profile, with warm colors indicating outward deformation and cool colors indicating inward deformation. (c) Mapped pipe trajectory overlaid on a satellite map, highlighting the dislocation spot. (d) Reconstructed 3D point cloud of the affected area. (e, f) Corresponding mesh with diameter and bump size analysis.

most trial yielded high-quality data products:

- In the first pipe, the robot identified an unexpected anomaly at 336 ft (Fig. 3.10). Combined visual, laser, and 3D reconstruction data revealed a pipe joint dislocation with an 18 mm diameter increase and an inward bump of 19 mm. These features were not detectable from 2D video footage or single-scan laser profile alone.
- In the second pipe, a 45-degree elbow was encountered 384 ft from the entry point, contradicting the pipe owner's initial map. This discovery helped correct infrastructure records.
- Reconstructed trajectories overlaid on satellite maps and the 3D meshes of localized deformities provided stakeholders with a new level of spatial and structural insight.

To the best of our knowledge, this is the first RGB-D dataset captured from inside real underground natural gas distribution pipelines. The result is a multi-modal, georeferenced dataset combining visual, 3D geometric, and inertial field data. These data products not only support structural assessment, but also provide a foundation for developing learning-based anomaly detection algorithms, which could enhance both real-time and retrospective inspection workflows.

3.5 Conclusion and Discussion

This chapter presented VILL-SLAM, a confined-space mapping algorithm designed for accurate localization and 3D reconstruction inside narrow pipelines. The system tightly integrates a monocular camera, IMU, laser profiler, and LiDAR into a robust multi-sensor fusion framework tailored for GPS-denied, low-feature environments. By fusing two redundant yet complementary sources of depth information—laser profiler and LiDAR—the method effectively mitigates drift during long-distance inspection.

To further improve localization accuracy, we introduced two novel optimization factors: the *LiDAR Cylinder Factor*, which leverages the cylindrical geometry of long, straight pipe segments, and the *LiDAR ICP Factor*, which improves robustness in geometrically diverse regions. These constraints further anchor the state estimation process to the known structure of the environment.

Field deployments validated the system's performance, with VILL-SLAM outperforming conventional SLAM methods in both localization and mapping accuracy. Real-world experiments demonstrated sub-millimeter-level, photorealistic 3D reconstructions, achieving an average localization drift of 0.84% and a local scanning error of 0.88 mm per point over a 22-meter trajectory in 12-inch diameter pipes.

While our method offers many advantages, there are several limitations:

1. **Calibration Sensitivity:** The system is highly sensitive to calibration quality, particularly the extrinsic calibration between the camera and laser. Small perturbations can lead to significant mapping errors. Future work may include implementing online calibration methods to reduce reliance on a single pre-deployment calibration.

2. **Visual Degradation:** The system's reliance on optical flow-based visual tracking makes it vulnerable to failures in low-feature or degraded visual conditions (e.g., fogged lenses). Real-time image quality monitoring and failure detection would help mitigate this risk.

3. **Failure Recovery:** Currently, the tightly coupled sensor fusion framework lacks robust fallback mechanisms. Failures caused by unreliable visual or inertial data—especially during bumpy traversal—could be addressed by incorporating relocalization strategies [76], lightweight fallback odometry, or redundancy through multi-odometry fusion [33]. A more modular software architecture with health checks per sensor would also improve fault tolerance [77].

4. **Generality:** The system currently assumes circular cross-sections. Broader applicability could be achieved by integrating geometric constraints suited to other pipe environments (e.g., elliptical or rectangular ducts).

5. **Loop Closure:** Enabling loop closure detection and correction for closed-loop or networked pipe systems remains an open area for future work.

Given the system's demonstrated field performance and practical utility, we are continuing to deploy and evaluate the VILL-SLAM framework in collaboration with partners in the infrastructure sectors. Ongoing work focuses on expanding the system's applicability across a variety of pipeline types and use cases—such as stormwater—while further hardening the hardware and improving software resilience for long-term, autonomous field use.

Chapter 4

Smoothing the Road from Lab to Field: A Design for Deployment (D4D) Framework for Robotic Systems Engineering

Bringing robotic technology from controlled laboratory settings into complex real-world environments remains a significant challenge. Ideally, engineering projects should begin by identifying a clearly defined problem in the field, followed by the development of a tailored technological solution. However, in practice, the process is often inverted: researchers and engineers frequently start with a novel technology and then seek out applications where it might be useful. More commonly, projects fall somewhere in between—stakeholders may articulate broad pain points or high-level needs, but detailed requirements are often ambiguous or evolving. This ambiguity is especially pronounced in under-explored or unfamiliar environments, where users themselves may not fully understand the operational constraints or workflow implications without first observing how a deployed robot behaves in situ. In such cases, progress is stymied by a classic chicken-and-egg dilemma: stakeholders need data from early deployments to define requirements, yet deployment requires a system that appears sufficiently mature.

Moreover, robotic systems rarely operate in isolation. Effective deployment requires careful consideration of how human operators, stakeholders, and support personnel interact with the robot in dynamic, often unpredictable conditions. This human-in-the-loop aspect introduces additional complexity, as engineers must account not only for technical functionality but also for human factors, user experience, and organizational workflows.

In such scenarios—characterized by unclear requirements, evolving understanding of the operational environment, and complex human-robot interactions—iterative development becomes essential. However, this raises a critical question: **How can we maximize outcomes per development iteration and accelerate the readiness of a robotic system for real-world deployment, particularly in the case where both the environment and the operational specifications are not well understood?**

This chapter explores that question in the context of robotic deployment in under-explored domains, such as infrastructure inspection or urban search and rescue (USAR) [78]. We will discuss the limitations of the existing systems engineering approaches, present a new framework, show its application in two retrospective case studies, and discuss its implications, limitations, and alignment with emerging trends in robotics systems engineering.

4.1 Limitations of Existing Systems Engineering Approaches

Conventional systems engineering frameworks, such as the V-model [13] (Fig. 4.1a), emphasize formal requirement specification, system decomposition, and structured integration. While effective in well-bounded projects with stable requirements, these approaches are less suited to exploratory applications. Their linear and hierarchical structure limits adaptability once development is underway, and they typically provide insufficient mechanisms for integrating evolving user feedback or contextual understanding during the project lifecycle. Human factors, if addressed at all, are often treated as secondary concerns introduced late in the development process.

Agile method [14] (Fig. 4.1b), by contrast, prioritize rapid iteration, continuous feedback, and incremental delivery. These methods are widely adopted in software

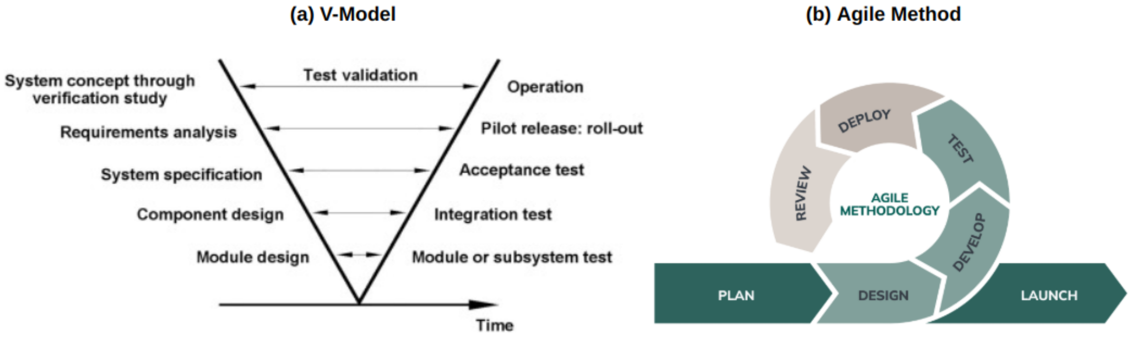

Figure 4.1: Systems engineering methods. (a) V-model [13]. (b) Agile method [14].

engineering, where systems can be modularized, deployed quickly, and updated frequently. However, applying agile principles to robotic systems presents practical challenges due to the interdependence of hardware and software, longer prototyping cycles, and the difficulty of obtaining quick feedback in physical environments. Moreover, agile's emphasis on speed can come at the expense of systems-level integration, safety, and performance robustness—crucial aspects in field robotics.

User-centered design (UCD) [15], originating from human-computer interaction and product design, emphasizes involving users early and iteratively to ensure solutions meet real-world needs. While not a systems engineering methodology, its principles—such as empathy, participatory design, and iterative prototyping—have influenced human-robot interaction research. However, in large-scale or exploratory engineering projects, UCD can be difficult to operationalize. It often requires intensive stakeholder engagement, which may not be feasible when user knowledge is limited or when users themselves are unsure of what they need. UCD also tends to focus on the product or interface level, rather than the full systems engineering process, which limits its utility in orchestrating complex deployments.

4.2 A Design-Inspired Systems Engineering Framework: Design for Deployment (D4D)

To address these limitations, we propose Design for Deployment (D4D), a systems engineering framework inspired by the iterative ethos of design thinking. Although robotic development is fundamentally constrained by systems engineering principles—such as traceability, verification, and performance guarantees—D4D borrows from the design discipline the emphasis on early-stage user involvement and rapid feedback cycles. The framework is intended not to replace existing engineering rigor, but to augment it with mechanisms for contextual responsiveness and deployment-oriented thinking.

D4D rests on two core principles:

- **Deployment-Oriented Development**: The development process should be tailored to maximize deployment efficiency by conceptualizing operational procedures, environmental factors, and system integration points from the outset. These elements should be considered not as downstream tasks, but as guiding parameters throughout the engineering lifecycle.
- **Collaborative Iteration with Stakeholders**: Users and stakeholders should be involved early and continuously—not merely as requirement providers but as co-creators. By engaging with them through iterative testing, engineers can align technical feasibility with practical needs, test contextual assumptions, and jointly manage expectations and project scope. This helps ensure that both functional and non-functional requirements converge toward deployment-readiness as the project matures.

The procedure of D4D is shown in Fig. 4.2. The steps are detailed as follows.

4.2.1 Concept of Operation

The Concept of Operation (ConOps) phase grounds the development process in real-world context before requirements are formalized. Drawing from the top-down logic of the V-model and enriched by principles from human-centered design and mission-oriented planning, this step focuses on answering: What is the system expected to do, and under what conditions?

D4D structures this phase around three perspectives:

- **Research**: Establishes technical and contextual understanding through literature, expert input, and user interviews. It informs early functional requirements by identifying environmental constraints, operational tasks, and deployment logistics.
- **Imagination**: Encourages scenario-based thinking to anticipate how users will

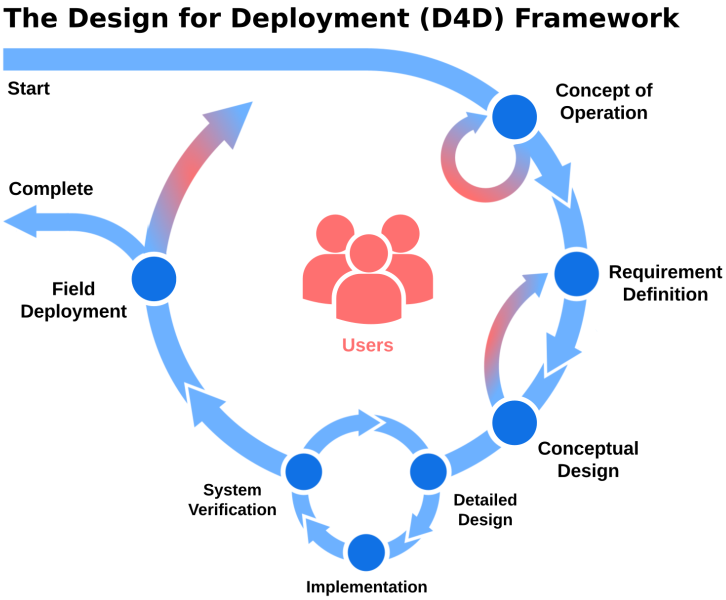

Figure 4.2: The proposed Design for Deployment (D4D) framework, inspired by [13, 14, 15]. The process starts with 'Concept of Operation', followed by requirement definition, conceptual and detailed design, implementation, verification, and field deployment. Our method integrates user input (represented by the red color) from the beginning, which influences both technical development and operational planning. The goal is to align system capabilities with deployment conditions and user needs.

interact with the system, what workflows will emerge, and what failure modes may arise. This helps surface non-functional requirements, such as usability, robustness, and human-system coordination.

- **Empathy**: Introduces user perspectives early, considering physical and cognitive demands on stakeholders. This supports intuitive design and early consideration of human factors, often missed in traditional systems engineering.

To formalize the system's intended use and provide a structured foundation for requirement definition, we adapted the concept of Mission and Environment Vectors—originally introduced as a Mission Description Language (MDL) for mission validation [79]—to clarify deployment goals and contextual constraints. While these vectors are used for post-development validation in the original work, we employed them as a structured way to clarify *system intent* at the beginning stage of development. We also propose an additional Human Context Vector to specify human interaction and operational context.

The **Mission Vector** represents a typical deployment mission's characteristics, such as distance per trial, duration, trajectory shape, payload delivery objectives (if any), and data collection objectives:

$$\mathbf{M} = \{M_{\text{dist}}, M_{\text{time}}, M_{\text{traj}}, M_{\text{pld}}, M_{\text{data}}\} \tag{4.1}$$

The **Environment Vector** captures factors like terrain type, spacial constraints or obstacles, precipitation, humidity, visibility, temperature, pressure, and potential threats or hazards:

$$\mathbf{E} = \{E_{\text{terrain}}, E_{\text{space}}, E_{\text{precip}}, E_{\text{humid}}, E_{\text{vis}}, E_{\text{temp}}, E_{\text{pressure}}, E_{\text{threats}}\} \tag{4.2}$$

The **Human Context Vector** captures expectations or constraints related to human-system interaction:

$$\mathbf{H} = \{H_{\text{ops}}, H_{\text{load}}, H_{\text{comms}}, H_{\text{log}}, H_{\text{safety}}, H_{\text{soc}}\} \tag{4.3}$$

- H_{ops} is the operator profile and skill level, addresses issues with usability and error rates.
- H_{load} is the physical and cognitive load, important for long missions and risk of operator fatigue.
- H_{comms} is the communication structure and protocols between users and between humans and the system. This factor helps define latency tolerances, UI needs, handoff mechanisms.
- H_{log} is the logistical complexity, including transport, setup time, calibration burden, number of personnel required. This factor encourages consideration of portability and deployment overhead.
- H_{safety} is the safety concerns and risk perception, such as human proximity to moving parts, emergency overrides, and fault tolerance.
- H_{soc} is the social, cultural, or organizational factors, including regulatory constraints, local norms, traffic control, or bystander interaction. This factor is useful for deployments in urban or public environments.

Together, these vectors provide a concrete foundation for requirements and ensure early alignment between engineers and users, even when initial needs are vague or evolving.

4.2.2 Requirement Definition and Conceptual Design

These two phases transition from intent to structure. Building on the ConOps, the requirement definition phase defines both functional and non-functional requirements, while conceptual design explores system architecture, modularity, and trade-offs, keeping deployment context in focus. During these phases, user feedback is crucial; early reviews on conceptual design with users help refine and prioritize requirements, thus improving conceptual design by eliminating unnecessary features, mitigating ambiguity and scope drift, and ensuring the expectations between the engineering team and users were well-aligned.

Unlike traditional models such as the V-model that assume fixed requirements, D4D allows for requirement refinement as design and development progresses and as user understanding deepens.

4.2.3 Detailed Design, Implementation, and Verification

In this phase, the system is developed and validated adhering to the hierarchical, top-down bottom-up practice like in the V-model. Specifically, it involves breaking down the system into smaller, manageable modules, implementation each module, and testing the modules and their integration to ensure they meet requirements.

For system verification, D4D recommends using a hybrid testing approach to reduce the risk of late-stage failure. This includes lab testing, mock field trials, and simulation. To avoid the sim-to-real gap, we recommends delaying high-fidelity simulation until real-world insights (even from minimal testing) inform accurate environmental modeling. This ensures that simulation remains grounded and useful as a verification tool.

4.2.4 Field Deployment

Rather than treating deployment as a final milestone as in the V-model, D4D positions it as the culmination of one iteration and the start of the next, as in the Agile method. Field deployment is where operational complexity, user behavior, and environmental variability converge—providing invaluable feedback for refining the system.

The framework emphasizes structured post-deployment reflection, updates to requirements and ConOps according to the evolving user needs and expectations, and renewed alignment with stakeholders. This closes the loop and accelerates progress across development cycles.

4.3 Case Study: Natural Gas Pipe Inspection Robot

This case study presents a retrospective analysis of a robotic inspection system designed to navigate decommissioned underground natural gas pipelines. This project represents a typical example of deployment in an under-explored environment—where user knowledge of deployment conditions is limited, and human involvement is required for successful system setup and control.

Overall, the deployment successfully completed its inspection objectives (see results in Section 3.4). While the project was guided by conventional systems engineering principles, the development process did not fully adhere to the Design for Deployment (D4D) framework. We examine each D4D phase to highlight missed opportunities and lessons learned, particularly around stakeholder engagement and human-system integration.

4.3.1 Concept of Operation and Requirement Definition

The initial task was defined as deploying a robot to inspect 12-inch metal pipes—horizontal, decommissioned, dry, potentially rusted, and up to 500 ft in length. This definition, based on informal user descriptions, physical samples, and online research, enabled the identification of core functional requirements such as in-pipe mobility and data collection for localization and mapping.

However, this early planning lacked structured user engagement and a comprehensive concept of operation, which led to several blind spots:

- Environmental factors such as humidity and temperature were underexplored, and user knowledge of pipe conditions (e.g., elbows, obstructions) was assumed rather than investigated.
- The Human Context Vector was not modeled, resulting in a lack of foresight around communication protocols, task sharing, and physical burden on human operators.
- No formal effort was made to elicit user workflows, operational constraints, or comfort with handling robotic equipment.

As a result, non-functional requirements such as portability, system usability, and operator interaction were underdeveloped, affecting later deployment success.

4.3.2 System Verification

System verification was conducted through mock deployments and simulation. These tests confirmed system functionality under controlled conditions. However, stress testing under realistic field conditions was not emphasized, as user engagement at this stage remained minimal. Additionally, an early simulation setup before acquiring realistic pipe data lacked realistic texture and lighting, leading to a significant sim-to-real gap. The simulation setup in Section 3.3.4 was in fact a rebuilt version. As outlined in the D4D framework, this issue could have been mitigated by delaying simulation until insights from minimal physical testing could inform its parameters.

4.3.3 Field Deployment and On-Site Operations

Direct user engagement occurred relatively late—during pre-deployment preparation—when several critical issues surfaced:

- Field operators highlighted environmental risks (e.g., muddy deployment site, standing water) that had not been previously considered. In response, reactive system hardening (e.g., splash-proofing, ruggedization) had to be implemented on short notice.
- Communication between the teleoperation team and field crew broke down

due to generator noise and physical separation, requiring an improvised human relay—a coordination issue that could have been resolved through early scenario planning and role definition.

- Operators were unfamiliar with handling sensitive components, leading to near-damage of critical tethers and wear on connectors.

These issues reflect the consequences of insufficient and late-stage user engagement. Earlier collaboration—through co-design and scenario walkthroughs—could have surfaced many of these challenges during planning, at lower cost.

4.3.4 System Limitations and Observed Gaps

Post-deployment analysis revealed several issues directly linked to insufficient user input and a lack of structured contextual modeling:

Perception: Vision-based SLAM was disrupted by fogged lenses and blurred images—conditions that could have been anticipated through environmental modeling and user insight.

Manual Operation Burden: Continuous gamepad control caused operator fatigue, underscoring the need for assistive driving systems. A more empathetic understanding of operator workload would have prompted earlier mitigation.

Mobility Mismatch: Unexpected elbows and uneven terrain—despite user claims that pipes were mostly straight—highlight the need for iterative requirement refinement based on real deployment data.

Portability: Though the system was designed to be modular and plug-and-play, all equipment required ten heavy carrying cases, demonstrating insufficient attention to logistical constraints and ease of transport.

Communication and Training: Field staff lacked familiarity with the system, resulting in mishandling. A structured training plan and clearer communication protocols would have improved reliability.

Situational Awareness: Teleoperators lacked feedback on robot orientation and tether status. These usability gaps could have been addressed through user-in-the-loop interface testing or early operator consultation.

4.3.5 Key Takeaways

This case highlights how traditional systems engineering processes—when applied without a structured deployment-oriented framework—can under-address environmental realism, human factors, and operational complexity. While the robot met functional expectations in lab conditions, real-world deployment exposed hidden gaps that impacted usability, robustness, and field coordination.

Had D4D been applied, early and continuous engagement with users, structured modeling of environmental and human context factors (e.g., via Mission, Environment, and Human Context Vectors), and iterative deployment planning could have significantly improved deployment readiness and field performance.

4.4 Case Study: Stormwater Pipe Inspection Robot with Structured Deployment Planning

This case study examines the development and deployment of a robotic system for inspecting underground stormwater pipes. In this case, a different hardware design but essentially the same sensing software was used compared with the previous case. This project followed the D4D framework more closely, particularly in terms of early stakeholder engagement, environmental modeling, and human factor consideration. The project illustrates how systematic use of the D4D principles—especially the Mission, Environment, and Human Context Vectors—helped reduce deployment friction and better align the system with real-world operational needs.

The deployment context was relatively better understood than in the previous case: the users provided detailed information about the operational environment and deployment procedures. However, as the deployment unfolded, certain discrepancies between reported and actual conditions revealed that even expert knowledge can contain gaps, reinforcing the importance of maintaining iterative feedback loops.

4.4.1 Concept of Operation and Requirement Definition

Early engagement with stakeholders enabled a structured and detailed formulation of the Concept of Operation. By explicitly modeling the three vectors of the D4D framework, the team was able to define constraints and requirements with greater clarity:

- The **Mission Vector** was defined through stakeholder input, including the expected mission trajectory shapes, pipe lengths, the range of pipe diameters (18 to 36 inches), and required bend traversal (e.g., 45-degree elbows with 1.5R turning radius).
- The **Environment Vector** included both internal pipe conditions—concrete walls, crack-induced bumps of up to 2 inches, splash risk—and external factors, such as high ambient temperatures and strong sunlight. These insights informed robustness features like splashproofing, thermal protection, and preparation of shade canopies to avoid screen glare.
- The **Human Context Vector** was proactively considered. For example, the robot had to be man-portable to accommodate the field crew's capabilities, and its form factor was constrained by the need to be lowered through standard manholes using a winch.

This early modeling enabled the team to define operational procedures in advance and align the system design with deployment constraints.

4.4.2 System Verification

Informed by the Concept of Operation, the system was verified using a mix of controlled and scenario-driven testing. Specific attention was paid to user interaction. For example, an attitude indicator was added to improve situational awareness and reduce teleoperator perceptual load. Communication between the teleoperation team and field operators was facilitated through walkie-talkies, addressing known pain points from the prior project. Also, operators were briefed on equipment handling, ensuring they were confident in setup and recovery procedures.

These interventions, driven by the Human Context Vector, significantly improved ease of use and reduced the chance of operational errors.

4.4.3 Field Deployment and On-Site Operations

The deployment proceeded smoothly, with the robotic system navigating multiple pipe segments as planned. The early effort invested in operational modeling and user preparation paid off in reduced rework and a coordinated workflow among team members.

However, the field trials also revealed that even informed users may unknowingly provide inaccurate data. For example, in one case, a pipe listed in user records as 24 inches in diameter was measured at 18 inches using the robot's onboard 3D sensing. This discrepancy required a reconfiguration of the robot's hardware, which was handled quickly but highlighted the persistent gap between expected and actual conditions. In another instance, a pipe believed to be drained contained standing water that gradually exceeded the robot's water-handling capability. While the robot was splashproof, it was not designed for submersion, necessitating mission termination.

These cases demonstrate the importance of **treating user input as valuable but imperfect**, and maintaining flexibility in both system design and field operations. While early modeling based on user knowledge is essential, iterative feedback and adaptive design are still necessary—core principles embedded in D4D.

4.4.4 Key Takeaways

This case illustrates the practical benefits of applying the D4D framework in field robotics projects. Early and structured stakeholder engagement enabled a more accurate Concept of Operation, informed by the Mission, Environment, and Human Context Vectors. As a result, the system design was more aligned with real-world constraints, better integrated into user workflows, and easier to operate and deploy.

At the same time, field deployment revealed that even knowledgeable stakeholders may provide incomplete or outdated information. The D4D framework addresses this by encouraging iterative updates to the operational model and continual engagement with users. Such flexibility allowed the team to respond effectively to emerging challenges, while avoiding many of the issues observed in the previous case study.

Overall, this case demonstrates that deployment-oriented planning and user engagement in systems engineering, as embodied in the D4D framework, can significantly improve the field readiness and robustness of robotic systems in uncertain environments.

4.5 Conclusion and Discussion

This chapter proposed the Design for Deployment (D4D) framework to address the challenge of developing robotic systems for under-explored environments, where user requirements are often vague and deployment conditions are not well understood. Through two case studies, we demonstrated how D4D's structured approach—anchored in early modeling of mission, environment, and human context—can help bridge the gap between lab prototypes and robust field-deployable systems.

The advantages of D4D are threefold. First, it promotes early engagement with users and stakeholders, which improves requirement alignment and reduces miscommunication. Second, it introduces formalism into operational modeling through the use of structured vectors, enabling clearer design constraints and more focused system verification. Third, it emphasizes iterative development with deployment as an integral part of the engineering process, rather than an afterthought.

However, D4D also has limitations. It requires additional upfront effort in stakeholder coordination and scenario modeling, which may not be feasible in resource-constrained projects. Moreover, it assumes that users are willing and able to participate meaningfully in early-stage design—an assumption that may not hold in all domains.

Nevertheless, D4D aligns with broader trends in robotics and systems engineering, including increasing attention to human-robot interaction, agile development in hardware systems, and the growing emphasis on deployability and real-world robustness. As robotics increasingly moves from controlled labs to complex field environments, frameworks like D4D are essential for making that transition more reliable, systematic, and user-informed.

Chapter 5

Conclusion

This thesis explored the central question: How to enable accurate localization and mapping inside space-constrained, feature-sparse, and GPS-denied pipelines using multi-sensor fusion? The answer required a combination of sensor system design, algorithmic innovation, deployment-aware system-level thinking that together address the unique challenges of pipeline environments. A core objective was to bridge the gap between theoretical SLAM techniques and practical, field-deployable robotic systems for underground infrastructure inspection. In this research, the journey from theory to deployment unfolded through a series of interlined contributions, each contributing a critical piece of the answer.

The initial foundation was laid by exploring the fundamentals of state estimation and SLAM, grounding the work in established methodologies while critically examining their limitations in the context of underground, confined environments. These environments present unique challenges such as geometric simplicity, a lack of distinctive visual features, and highly constrained sensor views—all of which complicate standard SLAM assumptions. Understanding these constraints was essential for motivating the development of specialized perception strategies.

To address these challenges, the thesis introduced **VILL-SLAM**, a novel fusion framework that integrates monocular vision, inertial sensing, a structured light laser profilometer, and LiDAR. This combination overcomes the metric scale ambiguity inherent in monocular vision and mitigates global drift through geometry-aware depth sensing. VILL-SLAM was shown to achieve sub-millimeter reconstruction accuracy and less than 1% localization error, demonstrating both the theoretical soundness of the approach and its practical performance in 12-inch and larger pipe environments.

Recognizing that real-world impact depends on more than algorithmic performance, the thesis turned toward the challenge of deployment. The transition from lab to field revealed the critical role of engineering practices that support reliability, adaptability, and usability under uncertain and variable conditions. To guide this process, a **Design for Deployment (D4D)** framework was introduced. This systems engineering approach emphasizes formalism in operational modeling, early user engagement, and iterative development informed by field deployments. Through two deployment case studies, the framework demonstrated how considerations beyond algorithmic performance—such as system integration, operator interaction, and environment variability—are essential for successful real-world implementation.

Collectively, this thesis contributes a full-stack solution to the in-pipe SLAM problem. It spans mathematical modeling, sensor suite design, software architecture, fusion algorithms, and deployment methodologies. The proposed methods not only work in controlled settings but have been shown to perform in diverse and unpredictable field conditions across gas and stormwater infrastructure domains.

This work highlights a broader insight: solving the in-pipe SLAM problem is not solely about improving perception algorithms. Success depends equally on selecting appropriate sensing modalities, designing integrated hardware-software systems, and planning for variability in the deployment environment. In this sense, the "how" of localization and mapping is both a technical and engineering question.

Future work will require a combination of continued research and practical deployment efforts to extend the impact of this work. The current localization frameworks provide a strong foundation for full autonomy in pipeline inspection robots. However, additional extensive testing in real-world environments is necessary to validate system robustness, identify unforeseen challenges, and refine performance under operational conditions. Integrating motion planning and teleoperation assistance could enhance navigational capabilities in complex or branching networks. Semantic analysis and anomaly detection could be layered onto the 2D and 3D data outputs to automate structural assessment and prioritize maintenance efforts. Scaling inspection to large networks may involve the development of multi-agent systems, where small robots collaboratively inspect different sections and share map information. Finally, sustained collaboration with infrastructure owners, industry stakeholders, and regulatory bodies will be essential to turn these technologies into deployable standards that shape the future of underground infrastructure monitoring.

In conclusion, this thesis demonstrates a complete pipeline—from theory to deployment—for multi-sensor fusion localization and mapping in pipe inspection. By combining perception algorithms, sensor integration, and deployment-aware system design, it contributes to the future of resilient, intelligent underground infrastructure monitoring. The methods and systems introduced in this thesis have the potential to benefit pipe owners, field operators, and infrastructure engineers by improving operational efficiency, reducing inspection costs, enabling as-built system quality assessment and performance analytics, and improving the safety and sustainability of critical pipeline infrastructure.

Bibliography

[1] Wikipedia contributors. San bruno pipeline explosion – wikipedia, the free encyclopedia, n.d. URL https://en.wikipedia.org/wiki/San_Bruno_pipeline_explosion. (document), 1.1

[2] Peter Coombes, Tom Micevski, and George Kuczera. Deterioration, depreciation and serviceability of stormwater pipes. 01 2002. (document), 1.1

[3] National Institute of Standards and Technology. Electrical duct bank issues, n.d. URL https://www.nist.gov/image/electrical-duct-bank-issues. (document), 1.1

[4] American Pipeline Solutions. Unlocking the benefits of smart pigging, 2023. URL https://www.americanpipelinesolutions.com/blog/2023/3/1/unlocking-the-benefits-of-smart-pigging. (document), 1.2, 1

[5] Spartan Tool. 8 things to consider when buying a sewer inspection camera. URL https://spartantool.com/blog/8-things-to-consider-when-buying-a-sewer-inspection-camera/. (document), 1.2, 1

[6] RapidView IBAK North America. Mainlite hd. URL https://rapidview.com/products/mainlite-hd/. (document), 1.2, 1, 2.2

[7] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018. (document), 2.1.5, 2.2.2, 3.2.1, 3.2.3, 3.2.4, 3.2.4, 3.3.1, 3.6

[8] J. Zhang and S. Sanjiv. Loam: Lidar odometry and mapping in real-time. *Robotics: Science and Systems*, 2, nov 2014. (document), 2.1.5, 1, 3.3.1, 3.6

[9] Wei Xu, Yixi Cai, Dongjiao He, Jiarong Lin, and Fu Zhang. Fast-lio2: Fast direct lidar-inertial odometry. *IEEE Transactions on Robotics*, 38(4):2053–2073, 2022. (document), 2.1.5, 1, 3.3.1, 3.6

[10] Raúl Mur-Artal and Juan D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. (document), 2.1.5, 2.2.2, 3.3.1, 3.6

[11] H. Wang, C. Wang, and L. Xie. Lightweight 3-d localization and mapping for solid-state lidar. *IEEE Robotics and Automation Letters*, 6(2):1801–1807, 2021. (document), 2.1.5, 3.3.1, 3.6

[12] Alejo Concha and Javier Civera. Rgbdtam: A cost-effective and accurate rgb-d tracking and mapping system. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6756–6763, 2017. (document), 3.3.1, 3.6

[13] Peter R. N. Childs. *Mechanical Design Engineering Handbook*, chapter 1, pages 1–47. Butterworth-Heinemann, Oxford, 2nd edition, 2019. Section 1.9.4, Chapter 1 - Design. (document), 4.1, 4.1, 4.2

[14] Pierre-Jean Lajoie Michaud. Agile software development: everything you need to know. https://www.nexapp.ca/en/blog/agile-software-development, March 2024. (document), 4.1, 4.1, 4.2

[15] Ji-Ye Mao, Karel Vredenburg, Paul W. Smith, and Tom Carey. The state of user-centered design practice. *Commun. ACM*, 48(3):105–109, mar 2005. (document), 4.1, 4.2

[16] Pipeline and Hazardous Materials Safety Administration (PHMSA). Pipeline incident 20 year trends, 2022. URL https://www.phmsa.dot.gov/data-and-statistics/pipeline/pipeline-incident-20-year-trends. 1

[17] NOAA National Centers for Environmental Information (NCEI). U.s. billion-dollar weather and climate disasters: State summary, 2025. URL https://www.ncei.noaa.gov/access/billions/state-summary/US. 1

[18] U.S. Congress Joint Economic Committee. Flooding costs the u.s. between $179.8 and $496.0 billion each year. Technical report, U.S. Congress Joint Economic Committee, June 2024. URL https://www.jec.senate.gov/public/_cache/files/bc171a7e-2829-462d-8193-7b7c4d59a6e3/jec-report-on-economic-cost-of-flooding.pdf. 1

[19] Dhanabalan Thangam, Haritha .M, R Ramesh, Ramakrishna Narasimhaiah, N Muddasir, Ahamed Khan, Shabista Booshan, Bharath Booshan, and R Ganesh. *Impact of Data Centers on Power Consumption, Climate Change, and Sustainability*, pages 60–83. 03 2024. 1

[20] W. Russell Kampfer, Ralf Bartzke, and Wolfgang Ziehl. Flexible mobile robot system for smart optical pipe inspection. In *Smart Structures*, 1998. 1, 2.2

[21] Alireza Ahrary. 17 sewer robotics. 2012. 1, 2.2

[22] Schonstedt Instrument Company. *Locating 101 Handbook*, chapter 5, pages 33–34. Schonstedt Instrument Company, 2018. 1

[23] Tina Tian, Luyuan Wang, Xinzhi Yan, Fujun Ruan, G. Jaya Aadityaa, Howie

Choset, and Lu Li. Visual-inertial-laser-lidar (vill) slam: Real-time dense rgb-d mapping for pipe environments. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1525–1531, 2023. 1

[24] R. E. Kalman. A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45, 1960. 2.1.1

[25] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. MIT Press, Cambridge, MA, 2005. ISBN 978-0-262-20162-9. 2.1.1

[26] Simon J. Julier and Jeffrey K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *Proceedings of AeroSense: The 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, volume 3068, pages 182–193, Orlando, FL, 1997. SPIE. 2.1.1

[27] Giorgio Grisetti, Rainer Kümmerle, Cyrill Stachniss, and Wolfram Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010. 2.1.2

[28] Frank Dellaert. Factor graphs and gtsam: A hands-on introduction. *Technical Report GT-RIM-CP__R-2012-002*, 2012. 2.1.3

[29] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g2o: A general framework for graph optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3607–3613, 2011. 2.1.3

[30] Sameer Agarwal, Keir Mierle, and The Ceres Solver Team. Ceres Solver, 10 2023. 2.1.3

[31] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, 33(1):1–21, 2017. 2.1.4, 3.1.2

[32] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, 2011. 2.1.5

[33] Shibo Zhao, Hengrui Zhang, Peng Wang, Lucas Nogueira, and Sebastian Scherer. Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8729–8736, 2021. doi: 10.1109/IROS51168.2021.9635862. 2.1.5, 3

[34] Jiarong Lin and Fu Zhang. R3live: 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), pages 10672–10678, 2022. 2.1.5

[35] Nan Yang, Rui Wang, Jörg Stückler, and Daniel Cremers. Df-vo: Deep feature-based visual odometry. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 707–722, 2020. 2.1.5

[36] Chaoyang Wang, Jose M. M. Montiel, and Juan D. Tardós. Deeptam: Deep tracking and mapping. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 834–849, 2018. 2.1.5

[37] Ronald Clark, Sen Wang, Hongkai Wen, Andrew Markham, and Niki Trigoni. Vinet: Visual-inertial odometry as a sequence-to-sequence learning problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3995–4001, 2017. 2.1.5

[38] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5297–5307, 2016. 2.1.5

[39] Everson Brum Siqueira, Rodrigo Zelir Azzolin, Silvia Silva da Costa Botelho, and Vinícius Menezes de Oliveira. Inside pipe inspection: A review considering the locomotion systems. In *CONTROLO 2016*, pages 449–458, 2017. 2.2

[40] Jalal Taheri Kahnamouei and Mehrdad Moallem. A comprehensive review of in-pipe robots. *Ocean Engineering*, 277:114260, 2023. ISSN 0029-8018. 2.2

[41] Andreu Corominas Murtra and Josep M. Mirats Tur. Imu and cable encoder data fusion for in-pipe mobile robot localization. In *2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, pages 1–6, 2013. 2.2, 2.2.2

[42] Michael Beller. Pipeline inspection utilizing ultrasound technology: On the issue of resolution. In *Proceedings of the Pigging Products and Services Association (PPSA) Seminar*, 2007. 2.2

[43] Yan Shi, Chao Zhang, Rui Li, Maolin Cai, and Guanwei Jia. Theory and application of magnetic flux leakage pipeline detection. *Sensors*, 15(12):31036–31055, 2015. ISSN 1424-8220. 2.2

[44] Kwanghyun Yoo, Dae-Kwang Kim, Jae-Jun Kim, Seung-Ung Yang, Hui-Ryoung Yoo, HongSeok Song, Seungjoon Jang, Chongam Kim, Yutaek Seo, and Dong-Kyu Kim. Digital twin model of in-line inspection systems for low-pressure, low-flow natural gas pipelines. *Gas Science and Engineering*, 134:205518, 2025. ISSN 2949-9089. 2.2

[45] Jie Zhang, Xudong Niu, Anthony J. Croxford, and Bruce W. Drinkwater. Pipe

inspection using guided acoustic wave sensors integrated with mobile robots. *NDT E International*, 139:102929, 2023. ISSN 0963-8695. 2.2

[46] Ting Wu, Shaohui Lu, and Yiping Tang. An in-pipe internal defects inspection system based on the active stereo omnidirectional vision sensor. In *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 2637–2641, 2015. 2.2

[47] Zheng Liu and Dennis Krys. The use of laser range finder on a robotic platform for pipe inspection. *Mechanical Systems and Signal Processing*, 31:246–257, 2012. ISSN 0888-3270. 2.2

[48] Juho Kannala, Sami S. Brandt, and Janne Heikkilä. Measuring and modelling sewer pipes from video. *Machine Vision and Applications*, 19(2):73–83, 2008. 2.2

[49] Wasim M. F. Al-Masri, Mamoun F. Abdel-Hafez, and Mohammad A. Jaradat. Inertial navigation system of pipeline inspection gauge. *IEEE Transactions on Control Systems Technology*, 28(2):609–616, 2020. 2.2, 2.2.2

[50] Chris H. Bahnsen, Anders Skaarup Johansen, Mark Philip Philipsen, Jesper W. Henriksen, Kamal Nasrollahi, and Thomas Baltzer Moeslund. 3d sensors for sewer inspection: A quantitative review and analysis. *Sensors (Basel, Switzerland)*, 21, 2021. 2.2

[51] Zhexiong Shang and Zhigang Shen. Dual-function depth camera array for inline 3d reconstruction of complex pipelines. *Automation in Construction*, 152:104893, 2023. ISSN 0926-5805. 2.2

[52] Zheng Liu and Yehuda Kleiner. State of the art review of inspection technologies for condition assessment of water pipes. *Measurement: Journal of the International Measurement Confederation*, 46:1–15, 2013. 2.2

[53] Qiuping Ma, Guiyun Tian, Yanli Zeng, Rui Li, Huadong Song, Zhen Wang, Bin Gao, and Kun Zeng. Pipeline in-line inspection method, instrumentation and data management. *Sensors*, 21(11), 2021. ISSN 1424-8220. 2.2

[54] Envirosight. ROVVER® X Mainline Crawler. `https://envirosight.com/products/rovver-x/`. 2.2

[55] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011. 2

[56] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. ISSN 1573-1405. 2

[57] Saber Kazeminasab, Neda Sadeghi, Vahid Janfaza, Moein Razavi, Samira Ziyadidegan, and M. Katherine Banks. Localization, mapping, navigation, and

inspection methods in in-pipe robots: A review. *IEEE Access*, 9:162035–162058, 2021. 2.2.2

[58] Rahul Summan, William Jackson, Gordon Dobie, Charles Macleod, Carmelo Mineo, Graeme West, Douglas Offin, Gary Bolton, Stephen Marshall, and Alexandre Lille. A novel visual pipework inspection system. *AIP Conference Proceedings*, 1949, 2018. 2.2.2, 3.2.4

[59] Peter Hansen, Hatem Alismail, Peter Rander, and Brett Browning. Monocular visual odometry for robot localization in lng pipes. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3111–3116, 2011. 2.2.2

[60] Kenki Matsui, Atsushi Yamashita, and Toru Kaneko. 3-d shape measurement of pipe by range finder constructed with omni-directional laser and omni-directional camera. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2537–2542, 2010. 2.2.2

[61] D. Cheng et al. Visual-laser-inertial slam using a compact 3d scanner for confined space. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5699–5705, 2021. 2.2.2, 3.1.1, 3.2.1, 3.2.5

[62] Daqian Cheng, Haowen Shi, Michael Schwerin, Michelle Crivella, Lu Li, and Howie Choset. A compact and infrastructure-free confined space sensor for 3d scanning and slam. In *2020 IEEE SENSORS*, pages 1–4, 2020. 2.2.2, 3.1.1

[63] Rob Worley, Ke Ma, Gavin Sailor, Michele M. Schirru, Rob Dwyer-Joyce, Joby Boxall, Tony Dodd, Richard Collins, and Sean Anderson. Robot localization in water pipes using acoustic signals and pose graph optimization. *Sensors*, 20(19), 2020. ISSN 1424-8220. 2.2.2

[64] R. Zhang, M. H. Evans, R. Worley, S. R. Anderson, and L. Mihaylova. Improving slam in pipe networks by leveraging cylindrical regularity. In Charles Fox, Junfeng Gao, Amir Ghalamzan Esfahani, Mini Saaj, Marc Hanheide, and Simon Parsons, editors, *Towards Autonomous Robotic Systems*, pages 56–65. Springer International Publishing, 2021. 2.2.2, 3.2.4

[65] Hoon Lim, Jae Youn Choi, Young Sik Kwon, Eui-Jung Jung, and Byung-Ju Yi. Slam in indoor pipelines with 15mm diameter. In *2008 IEEE International Conference on Robotics and Automation*, pages 4005–4011, 2008. 2.2.2

[66] Lionel Heng, Bo Li, and Marc Pollefeys. Camodocal: Automatic intrinsic and extrinsic calibration of a rig with multiple generic cameras and odometry. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1793–1800, 2013. 3.1.1

[67] Paul Furgale, Joern Rehder, and Roland Siegwart. Unified temporal and spatial calibration for multi-sensor systems. In *2013 IEEE/RSJ International Conference*

on Intelligent Robots and Systems, pages 1280–1286, 2013. 3.1.1

[68] Jianbo Shi and Carlo Tomasi. Good features to track. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994. 3.2.1

[69] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation*, pages 1–4, 2011. 3.2.1, 3.2.5

[70] P.J. Besl and H.D. McKay. A method for registration of 3d shapes. *IEEE Trans. Pattern Anal*, pages 230–256, 1992. 3.2.4

[71] Berthold KP Horn. Closed-form solution of absolute orientation using unit quaternions, 1987. 3.3.2

[72] N. Koenig and A. Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2149–2154 vol.3, 2004. 3.3.4

[73] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. 3.3.4

[74] American Petroleum Institute. *Fitness-for-Service (API RP 579-1/ASME FFS-1)*. American Petroleum Institute, Washington, DC, 3rd edition, 2016. 3.4

[75] G. Taubin. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13 (11):1115–1138, 1991. 3.4

[76] Hui Zhang, Xieyuanli Chen, Huimin Lu, and Junhao Xiao. Distributed and collaborative monocular simultaneous localization and mapping for multi-robot systems in large-scale environments. *International Journal of Advanced Robotic Systems*, 15(3), 2018. 3

[77] Kamak Ebadi and Lukas Bernreiter et al. Present and future of slam in extreme underground environments. *ArXiv*, abs/2208.01787, 2022. 3

[78] Julian Whitman, Nico Zevallos, Matt Travers, and Howie Choset. Snake robot urban search after the 2017 mexico city earthquake. In *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–6, 2018. 4

[79] Anton Dimov Hristozov. Fast and secure mission description, validation and deployment for safety-critical operations. In *2024 IEEE International Systems Conference (SysCon)*, pages 1–8, 2024. 4.2.1