Efficient Real-Time Calibration and Odometry for Dense Multi-Modal Mapping with UAVs

Dissertation

zur

Erlangung des Doktorgrades (Dr. rer. nat.)

der

Mathematisch-Naturwissenschaftlichen Fakultät

der

Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Jan Quenzel

aus

Schwerin, Deutschland

Bonn, March 2025

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn

Gutachter/Betreuer:	Prof. Dr. rer. nat. Sven Behnke
Gutachter:	Prof. Dr. rer. nat. Andreas Nüchter

Tag der Promotion:	07.11.2025
Erscheinungsjahr:	2025

Jan Quenzel

Efficient Real-Time Calibration and Odometry for Dense Multi-Modal Mapping with UAVs

Abstract

Autonomous robotic systems heavily rely on knowledge about their environment to safely navigate, interact with, and perform search and rescue (SAR) and inspection tasks in real-time. To better understand the robot's surroundings, a flying robot requires fast and robust perception, enabled by complementary sensors.

However, improper sensor calibration degrades the localization accuracy and reconstruction quality, which may lead to failure of the overall system. The common photometric error assumes a constant brightness, which is regularly violated in the real world and impairs the system's robustness. To restore this photometric consistency, we extract small oriented patches at tracked ORB features and jointly estimate the photometric parameters on keyframes including the exposure change. Our approach densely models the radial intensity fall-off due to vignetting and the camera response function with thin plate splines (TPS) from sparse measurements. To further improve runtime, we establish correspondences via direct gradient-based metrics and propose a novel robust combination of gradient orientation and magnitude, applicable for Visual-SLAM, disparity- and depth estimation.

Independent of ambient illumination, LiDARs provide accurate distance measurements around the robot even in texture-less environments. Thus, our LiDAR-inertial odometry MARS jointly aligns multi-resolution surfel maps with a Gaussian Mixture Model (GMM) formulation using a continuous-time B-spline trajectory. We accelerate covariance and GMM computation with Kronecker sums and products. An unscented transform (UT) de-skews surfels at runtime, while a timewise splitting into intra-scan segments facilitates motion compensation during spline optimization. Complementary soft constraints on relative poses from robot odometry and preintegrated IMU pseudo-measurements further improve our system's robustness and accuracy.

For high-level planning in dynamic environments, a signum occupancy function improves the reactivity of our mapping by maintaining a short temporal occupancy window in real-time. In addition, we enrich our dense map with color, thermal signatures, and semantic information using the spline trajectory for accurate and motion-compensated projection. Our semantic fusion further adapts a Bayesian update in logarithmic form for greater numerical stability.

The methods presented throughout this thesis provide state-of-the-art results on various datasets. As such, our created maps facilitate inspection and SAR while improving decision-making for further downstream tasks. Moreover, our methods are applicable for general dense 3D mapping and localization with, e.g., car-, robot-mounted, or handheld sensor suites.

Zusammenfassung

Autonom-agierende robotische Systeme sind auf eine verlässliche Umgebungswahrnehmung angewiesen, um sicher zu navigieren, mit der Umgebung zu interagieren sowie Such- und Rettungsmissionen (SAR) oder Inspektionsaufgaben durchzuführen. Für Flugroboter hat eine schnelle und robuste Wahrnehmung, unterstützt durch Sensorik mit komplementären Modalitäten, höchste Priorität. Allerdings beeinträchtigt eine fehlerhafte Kalibrierung nicht nur die Genauigkeit der Lokalisierung und Kartierung, sondern kann zum Ausfall des Systems führen.

In bildbasierten Systemen wird häufig eine konstante Helligkeit zwischen Bildkorrespondenzen angenommen. Jedoch wird diese Annahme in der Realität oft verletzt und reduziert somit die Robustheit der Verfahren. Um die photometrische Konsistenz wiederherzustellen, extrahieren wir kleine orientierte Bildausschnitte an getrackten ORB Merkmalen und schätzen die photometrischen Parameter gemeinsam inklusive der Belichtungsänderung anhand von Keyframes. Unser Ansatz modelliert sowohl Vignettierungsbedingten radialen Intensitätsabfall als auch die Kameraspezifische Antwortfunktion (CRF) mit Hilfe von Thin Plate Splines (TPS) dicht auf Basis von spärlichen Messungen. Zur weiteren Beschleunigung von visuellem SLAM, Disparitäts- und Tiefenschätzung stellen wir die Korrespondenzen direkt mit einer neuen robusten Metrik durch Kombination von Gradientenorientierung und -magnitude her.

Unabhängig von Lichtverhältnissen oder Textur misst ein LiDAR den Abstand zu Objekten im Sichtfeld sehr genau, womit sich der Sensor gut zur Schätzung der Eigenbewegung eignet. Unsere LiDAR-Inertial-Odometrie MARS registriert mehrere Multiresolutions-Surfel-Karten durch direkte Optimierung einer zeitkontinuierlichen Trajektorie anhand eines Gaußschen Mischmodells (GMM). Hierbei beschleunigen Kronecker Summen und Produkte die GMM- und Kovarianzberechnungen. Eine Unscented Transform (UT) entzerrt Surfel zur Laufzeit während eine temporale Aufteilung in Intra-Scan Segmente die Bewegungskompensation als Teil der Splineoptimierung umsetzt. Komplementäre Nebenbedingungen für relative Posen und vorintegrierte IMU Pseudomessungen erhöhen die Robustheit und Genauigkeit.

Für die Planung in dynamischen Umgebungen verwenden wir für die Belegtheit eine Signumfunktion, um ein kurzes zeitliches Belegtheitsfenster zu realisieren und so die Reaktivität unserer Karte zu verbessern. Darüber hinaus integrieren wir Semantik, Farb- und Thermalinformationen in unserer dichten 3D Karte unter Berücksichtigung der Kamerabewegung anhand der Splinetrajektorie. Für die semantische Fusion adaptieren wir das Bayes'sche Update in logarithmischer Form zur Verbesserung der numerischen Stabilität. Die Resultate der hier vorgestellten Methoden spiegeln den aktuellen Stand der Kunst auf verschiedenen Datensätzen wider. Als solches, vereinfachen die von unseren Verfahren erstellten Karten die Inspektion sowie Such- und Rettungsmaßnahmen, während gleichzeitig die nachfolgende Entscheidungsfindung verbessert wird. Dabei sind unsere Methoden nicht auf diese Einsatzfälle beschränkt, sondern generell für die dichte 3D-Kartierung und Lokalisierung mit Fahrzeugen, Robotern oder tragbaren Sensorsystemen einsetzbar.

Acknowledgments

Ohana means family.
Family means nobody gets left behind, or forgotten.

— Lilo & Stitch

Thank you to my family, friends, colleagues and Prof. Behnke,
for this remarkable time.

Any crash you can walk away from
is a good crash!

— Launchpad McQuack

You can't make an omelette without breaking eggs.
As the saying goes, for all the countless robots I worked with over these years,
only two have been lost beyond repair:

In memory of "Splasher" and "DRZ2".

This work was supported by grant BE 2556/7 as part of the research group FOR 1505 by the German Research Foundation (DFG) and grant 608849 of the European Union's 7th FP as well as under Germany's Excellence Strategy, EXC-2070 - 390732324 (PhenoRob) and by the German Federal Ministry of Education and Research (BMBF) in the projects "Kompetenzzentrum: Aufbau des Deutschen Rettungsrobotik-Zentrums (A-DRZ)", grant 13N14859, "Etablierung des Deutschen Rettungsrobotik-Zentrums (DRZ)", grant 13N16477, and "UMDenken: Supportive monitoring of turntable ladder operations for firefighting using IR images", grant 13N16811.

Contents

Acronyms

AR	augmented reality
APE	absolute position error
ATE	absolute trajectory error
AVX	advanced vector extensions
BIM	building information model
GBA	global bundle adjustment
LBA	local bundle adjustment
BRIEF	binary robust independent elementary features
CG	conjugate gradient
CRF	conditional random field
CRF	camera response function
CPU	central processing unit
CNN	convolutional neural network
CT-ICP	Continuous-Time ICP
deque	double-ended queue
DoF	degrees of freedom
DS	double sphere camera model
DLO	direct LiDAR odometry
DLIO	direct LiDAR-inertial odometry
DSO	direct sparse odometry
KF	Kalman filter
EKF	extended Kalman filter (KF)
EM	expectation maximization
EMoR	empirical model of response
EWMA	exponentially weighted moving average
FAST	features from accelerated segment test
FMA	Fused Multiply-Add
FoV	field-of-view
FPN	feature pyramid network
GD	gradient descent
GM	gradient magnitude
GGCM	generalized gamma curve model

ICP	iterative closest point
GICP	generalized ICP
GMM	Gaussian mixture model
GN	Gauss-Newton
GP	Gaussian process
GPS	global positioning system
GPU	graphics processing unit
GNSS	global navigation satellite system
GS	Gaussian splatting
HDR	high dynamic range
HSV	hue, saturation, value
IMU	inertial measurement unit
IEKF	iterated error-state Kalman filter
IoU	intersection over union
IRLS	iterative reweighted least squares
KLT	Kanade–Lucas–Tomasi tracker
KLD	Kullback-Leibler divergence
LM	Levenberg-Marquardt
LP	label propagation
LRM	Large Reconstruction Model
LO	LiDAR odometry
LIO	LiDAR-inertial odometry
LIO-MARS	LiDAR-inertial odometry with MARS maps
LIC	LiDAR-inertial-camera odometry
LiDAR	Light Detection and Ranging
LOAM	localization and mapping
LSTM	long short-term memory
LSD-SLAM	Large-Scale Direct Monocular SLAM
LUT	look-up-table
MAD	mean absolute deviation
MARS	Multi-Adaptive-Resolution-Surfel
MAE	mean absolute error
MLE	maximum likelihood estimator
MLP	multi-layer perceptron
MoCap	Motion Capture
MVS	multi-view stereo

NCC	normalized cross-correlation
NDT	Normal distribution transform
NeRF	neural radiance field
NID	normalized information distance
NN	nearest neighbor
ORB	Oriented FAST and Rotated BRIEF
OKVIS	open keyframe-based visual-inertial SLAM
PDE	partial differential equation
PGO	pose graph optimization
PTAM	Parallel Tracking and Mapping
RAM	random access memory
RANSAC	random sample consensus
RMS	root-mean-squared
RMSE	RMS error
RGB	red, green, blue
RGB-D	RGB and depth
RBF	radial basis function
ROS	robot operating system
RP	radial polynomial
RTK-GPS	real-time kinematic GPS
SAR	search and rescue
SDE	stochastic differential equation
SGF	scaled normalized gradient field
SIFT	scale-invariant feature transform
SIMD	single instruction, multiple data
SfM	structure-from-motion
SLAM	simultaneous localization and mapping
slerp	spherical linear interpolation
s.p.d.	symmetric positive semidefinite
SDF	signed distance function
ESDF	Euclidean SDF
SE-LIO	Semi-Elastic-LIO
SVO	semi-direct visual odometry
SSD	sum of squared differences
SSIM	structured similarity index measure
surfel	surface element

SuMa	surfel mapping
TLS	terrestrial LiDAR scanner
TSDF	truncated SDF
TPS	thin plate spline
UAV	unmanned aerial vehicle
UGV	unmanned ground vehicle
UT	unscented transform
ViT	vision transformer
VIO	visual-inertial odometry
VO	visual odometry
VSLAM	Visual-SLAM
voxel	volume element

1 Introduction

Robots promise unique opportunities and novel applications in our everyday lives to speed up routine tasks, reduce risks for human operators, or prevent endangering people in the first place.

Over the last two decades, first responders (Ollero et al., 2006; Murphy et al., 2021; Lattimer et al., 2023) and inspectors (Lottes et al., 2017; Jordan et al., 2018; Rakha and Gorodetsky, 2018; Quenzel et al., 2019; Meribout et al., 2023) have been quick to adopt multi-rotor drones with color and thermal cameras as an effective tool for remote reconnaissance and surveying in unstructured real-world environments, as shown in Fig. 1.1. As technology matures and availability increases, piloting unmanned aerial vehicles (UAVs) became easily accessible without extensive training and, thus, enables everyone to see the world from an entirely new perspective.

Nowadays, UAVs deliver crucial overviews or close-ups of difficult-to-reach or hostile environments, e.g., after incidents or natural disasters (Alon et al., 2021; Ray et al., 2022; Manzini et al., 2023; Surmann et al., 2024). Such imagery was previously restricted to larger aircraft or completely impossible due to space and safety concerns.

However, in practice, current applications have various severe limitations. Drones overly rely on global navigation satellite systems (GNSSs) and the availability of free space all around them to ensure safe navigation. Preprogrammed flight paths need to be collision-free and require GNSS.

As a consequence, manual operation via a live video stream with a limited field-of-view (FoV) is the standard for UAV-based search and rescue (SAR) missions (Murphy et al., 2021; Ray et al., 2022). For example, during SAR in an unstable or partially collapsed building (Murphy et al., 2021), the operator should focus on their assigned task, such as looking for injured inhabitants, finding entrance points, or identifying possible hazards, and not on piloting the drone. Assistance functions like waypoint navigation (Schleich, Beul, Quenzel, and Behnke, 2021) reduce the burden on the operator in high-stress situations.

In the vicinity of structures, problems mutually intensify due to limited free space, obstacles blocking the operator's direct line of sight, an impaired wireless data connection behind barriers (Patchou et al., 2022), and reduced GNSS precision and availability (USSF 2022). At the same time, operators have to consider the UAV's surroundings and cope with accompanying risks. Since a single image provides only little structural information, the situation becomes more challenging and exhausting for the pilot. Under these circumstances, knowing the UAVs' position and orientation within a 3D model (Quenzel and Behnke, 2021) of the immediate environment improves the situational awareness tremendously (Surmann et al., 2024). Modern obstacle avoidance (Schleich and Behnke, 2022) subsequently reduces the risk of losing the drone.

Processing of gathered UAV footage is a common practice for inspection tasks (Rakha and Gorodetsky, 2018). However, it remains the exception in

Figure 1.1: Applications: a) Industrial inspection inside a chimney. b) Targeted examination and localization of facade fires. c) Screening for clusters of embers during wildfire exercise. d) Exploration during a natural disaster without endangering first responders.

SAR (Manzini et al., 2023). Aggravating the matter in both cases, data processing only happens after landing and not at flight time. To harness the vast and previously untapped potential for autonomous assistance functions and automatic scene reconstruction of metric-scale multi-modal 3D maps in real-time, UAVs (Beul et al., 2015; Burri et al., 2015) are equipped with additional compute capabilities and multi-modal sensors (Beul et al., 2018; Schleich et al., 2021), as shown in Fig. 1.2.

In order to benefit from onboard processing, the deployed methods require being:

- fast, as we need the result at runtime with low latency,
- accurate, such that the map depicts the actual environment, and
- robust, to work in various environments.

Throughout this thesis, we present novel approaches for calibration, odometry, and mapping that fulfill these requirements and are designed to work in close proximity to structures in GNSS-denied environments. Furthermore, our solutions lay the foundation for assistance functions such as obstacle avoidance, navigation, and exploration that foster safe manual and enable autonomous drone operations (Quenzel et al., 2019; Schleich et al., 2021).

The multi-modal 3D map contains the observed structures with spatial dimensions, color (Rosu et al., 2019b), thermal signatures (Rosu et al., 2019a), and the semantics to categorize objects and surfaces (Behley et al., 2019; Bultmann et al., 2023). Additionally, it depicts the whereabouts of the UAV (Quenzel and Behnke, 2021) at any point in time in relation to its vicinity. As a consequence, these maps assist in making more informed decisions on downstream tasks.

Our methods are not limited to inspection and SAR with UAVs but are also applicable for general dense 3D mapping and localization with, e.g., car-, robot-

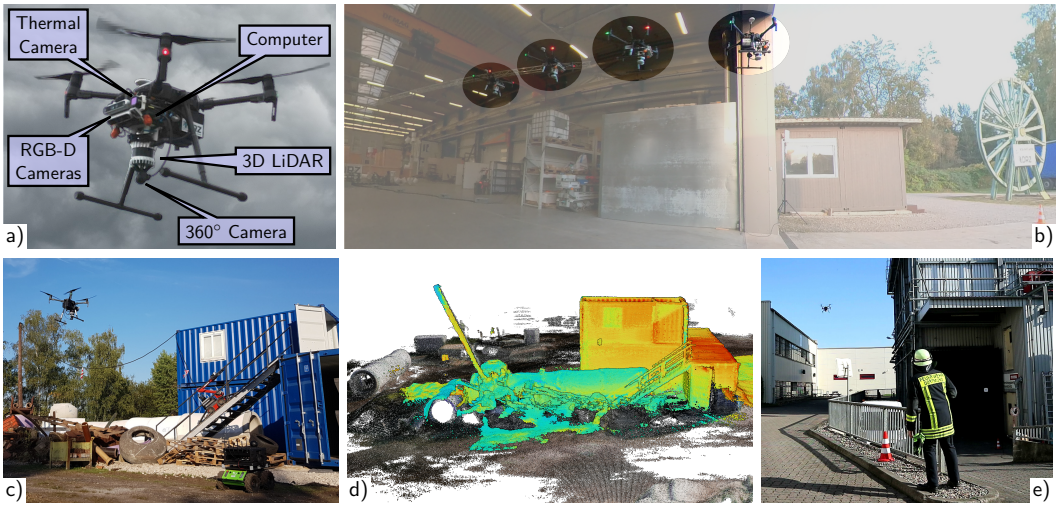

Figure 1.2: Enabled Autonomy: a) UAV with multi-modal sensors and onboard computing capability. b) Autonomous flight in GNSS-denied indoor environment (from left to right). c) Multi-robot cooperation with teleoperated ground vehicles. d) Automatic scene reconstruction of a disaster site. e) Autonomous exploration for firefighters.

mounted (Kim et al., 2020; Wen et al., 2020), or handheld sensor suites (Zhang et al., 2021) and do not require apriori knowledge about the environment.

1.1 Tasks and Key Contributions

In this thesis, we present novel approaches for robust perception to tackle the following three tasks:

T1) on-the-fly photometric calibration using general environment geometry without the need for specific calibration targets,

T2) real-time state estimation to enable higher levels of autonomy for aerial and ground robots,

T3) dense 3D mapping with motion compensation to create accurate environment maps enriched by sensors of different modalities.

The four key contributions to handle these tasks are:

1. dense correction factors from 2D thin plate splines (TPSs) (Ch. 3) for vignetting calibration (addressing T1),

2. a novel gradient-based dissimilarity metric (Ch. 4) to robustify direct image alignment (addressing T2),

3. a continuous-time LiDAR-inertial odometry (LIO) (Ch. 5) with joint registration of multiple surfel maps (addressing T2, T3), and

4. a Bayesian fusion (Ch. 7) in logarithmic form for semantic mapping (addressing T3).

The first key contribution utilizes 2D TPSs to approximate dense pixelwise correction factors for vignetting estimation from sparsely distributed correspondences without requiring multiple per-pixel measurements. Oriented patches around ORB features (Rublee et al., 2011) are the basis for the joint photometric calibration (Ch. 3) and for map point radiance estimation using just keyframes. A sixth-order polynomial captures the general shape of the vignetting, whilst the TPS captures local deformations. A 1D TPS with border conditions models the camera response function (CRF) in combination with a Gamma curve. The radiance allows the estimation of the exposure ratio w.r.t. the tracked map for the current frame. Our calibration handles natural scenes without uniform illumination or known calibration targets and is even usable at runtime on a laptop central processing unit (CPU).

The second contribution addresses the problem of robustifying the direct alignment of image pairs by replacing the photometric error with a new gradient-based dissimilarity metric (Ch. 4). Our approach combines a gradient orientation-based metric with a magnitude-dependent scaling term, which is easy to integrate into existing visual odometry (VO) systems and increases their robustness while running at the frame rate of a typical camera. Our evaluation shows that our metric leads to more robust and more accurate estimates of the scene depth in typical disparity tasks as well as camera trajectories from direct image registration.

The third contribution concerns continuous-time LIO running in real-time onboard a UAV. Our LiDAR-inertial odometry with MARS maps (LIO-MARS) jointly registers multiple consecutive scans using a Gaussian mixture model (GMM) formulation by directly optimizing the timewise non-uniformly spaced B-spline knots that represent the trajectory. Prior to registration, embedding scans into multi-resolution surfel maps with sparse permutohedral lattices (Adams et al., 2010) greatly reduces the number of residuals during optimization. Adaptive selection of the appropriate surfel resolution further improves efficiency (Ch. 5). Rephrasing the GMM and surfel covariances with Kronecker sum and products improves parallelization (Ch. 6). A timewise splitting into intra-scan segments facilitates motion compensation at optimization time, while an unscented transform (UT) enables the de-skewing of individual surfels at runtime without costly pointwise reintegration. The continuous-time trajectory further allows us to leverage relative motion constraints to increase resilience, e.g., from robot odometry or preintegrated IMU measurements (Ch. 6). Our LIO-MARS provides reliable pose estimates with state-of-the-art accuracy on various datasets.

The fourth contribution adapts a Bayesian fusion (Ch. 7) for semantic mapping to logarithmic form for higher precision and greater numerical stability. Furthermore, our continuous-time trajectory allows seamless integration of multiple sensor modalities with differing time offsets for more consistent poses, enabling dense mapping with sparse volumes for real-time fusion.

We demonstrate the merits of our methods on real-world datasets and in real-robot experiments. Our contributions w.r.t. the state of the art are presented and discussed in detail in chapters 3 to 7.

1.2 Publications

Parts of this thesis have been published in journals and peer-reviewed conference proceedings. The publications are provided in chronological order:

J. Quenzel, J. Horn, S. Houben, and S. Behnke (2018). "Keyframe-based Photometric Online Calibration and Color Correction." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593595

J. Quenzel, R. A. Rosu, T. Läbe, C. Stachniss, and S. Behnke (2020). "Beyond Photometric Consistency: Gradient-based Dissimilarity for Improving Visual Odometry and Stereo Matching." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9197483

D. Schleich, M. Beul, J. Quenzel, and S. Behnke (2021). "Autonomous Flight in Unknown GNSS-denied Environments for Disaster Examination." In: *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*. DOI: 10.1109/ICUAS51884.2021.9476790

S. Bultmann*, J. Quenzel*, and S. Behnke (2021). "Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles." In: *Proceedings of the European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR50962.2021.9568812

J. Quenzel and S. Behnke (2021). "Real-time Multi-Adaptive-Resolution-Surfel 6D LiDAR Odometry using Continuous-time Trajectory Optimization." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS51168.2021.9636763

S. Bultmann, J. Quenzel, and S. Behnke (2023). "Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles with Label Propagation for Cross-Domain Adaptation." In: *Journal of Robotics and Autonomous Systems* 159. DOI: 10.1016/j.robot.2022.104286

An * denotes equal contribution.

Further contributed peer-reviewed papers are relevant to this thesis and cited as external literature in chronological order:

S. Houben, J. Quenzel, N. Krombach, and S. Behnke (2016). "Efficient multi-camera visual-inertial SLAM for micro aerial vehicles." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2016.7759261

J. Quenzel, R. A. Rosu, S. Houben, and S. Behnke (2017). "Online Depth Calibration for RGB-D Cameras using Visual SLAM." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2017.8206043

J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall (2019). "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2019.00939

J. Quenzel, M. Nieuwenhuisen, D. Droeschel, M. Beul, S. Houben, and S. Behnke (2019). "Autonomous MAV-based Indoor Chimney Inspection with 3D Laser Localization and Textured Surface Reconstruction." In: *Journal of Intelligent & Robotic Systems (JINT)* 93.1. DOI: 10.1007/s10846-018-0791-y

R. A. Rosu, J. Quenzel, and S. Behnke (2019b). "Semi-supervised Semantic Mapping Through Label Propagation with Semantic Texture Meshes." In: *International Journal of Computer Vision (IJCV)* 128.5. DOI: 10.1007/s11263-019-01187-z

R. A. Rosu, J. Quenzel, and S. Behnke (2019a). "Reconstruction of Textured Meshes for Fire and Heat Source Detection." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR.2019.8848943

R. A. Rosu, P. Schütt, J. Quenzel, and S. Behnke (2020). "LatticeNet: Fast Point Cloud Segmentation Using Permutohedral Lattices." In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2020.XVI.006

I. Kruijff-Korbayova, R. Grafe, N. Heidemann, A. Berrang, C. Hussung, C. Willms, P. Fettke, M. Beul, J. Quenzel, D. Schleich, S. Behnke, J. Tiemann, J. Güldenring, M. Patchou, C. Arendt, C. Wietfeld, K. Daun, M. Schnaubelt, O. von Stryk, A. Lel, A. Miller, C. Roehrig, T. Straßmann, T. Barz, S. Soltau, F. Kremer, S. Rilling, R. Haseloff, S. Grobelny, A. Leinweber, G. Senkowski, M. Thurow, D. Slomma, and H. Surmann (2021). "German Rescue Robotics Center (DRZ): A Holistic Approach for Robotic Systems Assisting in Emergency Response." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR53300.2021.9597869

H. Surmann, K. Daun, M. Schnaubelt, O. von Stryk, M. Patchou, S. Böcker, C. Wietfeld, J. Quenzel, D. Schleich, S. Behnke, R. Grafe, N. Heidemann, D. Slomma, and I. Kruijff-Korbayova (2024). "Lessons from robot-assisted disaster response deployments by the German Rescue Robotics Center task force." In: *Journal of Field Robotics (JFR)* 41.3. DOI: 10.1002/rob.22275

J. Quenzel, L. T. Mallwitz, B. T. Arnold, and S. Behnke (2024). "LiDAR-Based Registration Against Georeferenced Models for Globally Consistent Allocentric Maps." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR62954.2024.10770000

1.3 Thesis Outline

This thesis is structured as follows:

Chapter 2 provides the theoretical background on unconstrained optimization, transformations, camera projection, image distortion, and typical error measures used for evaluation and method comparison.

Chapter 3 details our real-time method for online photometric calibration of RGB cameras using sparse image feature correspondences for dense vignetting correction with TPS, CRF, and relative exposure estimation with application to color correction to improve the accuracy of upstream tasks like structure-from-motion (SfM) and dense multi-view stereo (MVS).

Chapter 4 introduces a robust gradient-based dissimilarity metric for direct pixelwise matching required by depth estimation and direct image alignment without explicit color correction.

Chapter 5 proposes a real-time LiDAR odometry by joint alignment of a sliding scan window against local keyframes using multi-resolution surfel maps with a local continuous-time B-spline trajectory.

Chapter 6 extends the approach of Ch. 5 with a non-uniform continuous-time trajectory over the whole sequence and integrates inertial and complementary relative motion estimates as well as surfel-based motion compensation.

Chapter 7 presents approaches for dense mapping of occupancy, color, thermal, and semantic information.

Finally, Ch. 8 discusses the main findings, summarizes, and concludes this thesis.

2

Preliminaries

2.1 Notation

Throughout this work, we denote sets with capital calligraphic ($\mathcal{A}$), matrices with capital (A), and vectors with bold lowercase letters ($\boldsymbol{a}$).

When referencing an entry within a vector, we use the subscript to denote its position with zero-based indexing. Hence, the second entry in a column vector $\boldsymbol{a}$ is a_1. For 2D-matrices, we add both dimensions as subscripts. For example, A_{21} is the entry in the third row and second column. Similarly, $A_{2\times 3}$ denotes the left upper 2-by-3 block matrix.

With $\operatorname{diag}(\boldsymbol{a})$, we denote a diagonal matrix with the elements of the vector $\boldsymbol{a}$ along its diagonal.

2.2 Unconstrained Optimization

Given a function $f(\boldsymbol{x})$, one seeks to obtain an optimal solution through minimization:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}\in\mathbb{R}^N}{\arg\min}\, f(\boldsymbol{x}). \tag{2.1}$$

Evaluation of all parameter combinations in $\boldsymbol{x}$ is, in general, infeasible, especially for continuous and multi-dimensional $\boldsymbol{x} \in \mathbb{R}^N$.

Similarly, $f(\boldsymbol{x})$ is often highly non-linear, such that a small change in $\boldsymbol{x}$ may change the value of f tremendously. We generally describe a change in $\boldsymbol{x}$ by either an update or a downdate with the $\boxplus$ and $\boxminus$-operator, respectively. For $\mathbb{R}^N$, these are simple vector addition ($+$) and subtraction ($-$). As we will see later, these definitions differ for transformations and manifolds (Ch. 6).

Furthermore, the optimal solution cannot be obtained analytically for most functions $f(\boldsymbol{x})$. Nevertheless, a first-order Taylor series expansion $g(\boldsymbol{x})$ of $f(\boldsymbol{x})$ at an initial estimate $\boldsymbol{x}_0$:

$$g_1(\boldsymbol{x}) = f(\boldsymbol{x}_0) + (\boldsymbol{x} \boxminus \boldsymbol{x}_0)^\intercal \nabla f(\boldsymbol{x}_0), \tag{2.2}$$

allows approximating $f(\boldsymbol{x})$ with a much simpler and easier-to-optimize function $g_1(\boldsymbol{x})$. Locally — close to the linearization point $\boldsymbol{x}_0$ — the difference between g_1 and f is small. From $g_1(\boldsymbol{x})$ we calculate a descent direction $\boldsymbol{d}(\boldsymbol{x}_i)$ that, once applied to the current estimate, yields an estimate $\boldsymbol{x}_{i+1}$ with a lower value:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i \boxplus \gamma_i \boldsymbol{d}_i(\boldsymbol{x}_i). \tag{2.3}$$

Here, $\gamma_i \in \mathbb{R}_{>0}$ is the current step size, e.g., $\gamma = 1$, and has to be chosen such that $f(\boldsymbol{x}) \geq f(\boldsymbol{x}_{i+1})$. For the gradient descent (GD) method, this descent direction is the negative gradient ∇g_1:

$$\boldsymbol{d}_i(\boldsymbol{x}_i) = -\nabla g_1(\boldsymbol{x}_i). \tag{2.4}$$

The GD method leads to a local optimum, which is not necessarily a unique or global minimum. Furthermore, GD converges slowly and may zigzag towards the local optima. The momentum method (Polyak, 1964) reduces zigzagging through a linear combination of the last and the current gradient directions; in analogy to a heavy ball rolling down a hill due to momentum, the gradient direction does not change instantaneously. More advanced methods like Adam (Kingma and Ba, 2015) and AdamW (Loshchilov and Hutter, 2019) are established to optimize neural network weights.

A second-order Taylor series expansion further helps to develop methods with faster convergence:

$$g_2(\boldsymbol{x}) = f(\boldsymbol{x}_0) + (\boldsymbol{x} \boxminus \boldsymbol{x}_0)^\intercal \nabla f(\boldsymbol{x}_0) + \frac{1}{2} (\boldsymbol{x} \boxminus \boldsymbol{x_0})^\intercal \nabla^2 f(\boldsymbol{x}_0) (\boldsymbol{x} \boxminus \boldsymbol{x}_0) . \tag{2.5}$$

The quadratic function $f(\boldsymbol{x})$:

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^\intercal A \boldsymbol{x} + \boldsymbol{b}^\intercal \boldsymbol{x} + c, \tag{2.6}$$

is convex if A is symmetric positive semidefinite (s.p.d.) and its minimizer $\boldsymbol{x}^*$ is unique (Nocedal and Wright, 2006). Setting the gradient ∇f to zero gives rise to a linear system of equations:

$$A\boldsymbol{x} = \boldsymbol{b}, \tag{2.7}$$

The solution of Eq. 2.7 is the minimizer $\boldsymbol{x}^*$ of $f(\boldsymbol{x})$. Equation 2.5 is evidently a quadratic function $f(\boldsymbol{r})$ with:

$$\boldsymbol{r} = \boldsymbol{x} - \boldsymbol{x}_0, \tag{2.8}$$
$$A = \nabla^2 f(\boldsymbol{x}_0), \tag{2.9}$$
$$\boldsymbol{b} = \nabla f(\boldsymbol{x}_0)^\intercal, \tag{2.10}$$
$$c = f(\boldsymbol{x}_0). \tag{2.11}$$

Hence, we obtain for an s.p.d. matrix A the minimizer of Eq. 2.5 via solving Eq. 2.7. The solution is the descent direction of the Gauss-Newton (GN) method:

$$\boldsymbol{d}_i(\boldsymbol{x}_i) = -\left(\nabla^2 g_2(\boldsymbol{x}_i)\right)^{-1} \nabla g_2(\boldsymbol{x}_i)^\intercal. \tag{2.12}$$

When A is negative definite or $f(\boldsymbol{x}) < f(\boldsymbol{x}_{i+1})$, a modification to the Hessian Eq. 2.9 is helpful with the s.p.d. matrix D:

$$A = \nabla^2 f(\boldsymbol{x}_0) + \lambda D. \tag{2.13}$$

Examples for D include the identity matrix I or a diagonal matrix scaled with the maximal value of the Hessian's diagonal:

$$D = \max(\mathrm{diag}(\nabla^2 f)) \cdot I. \tag{2.14}$$

λ allows seemlessly interpolating between GD ($\lambda \to \infty$) and GN ($\lambda \to 0$). The adjustment of λ has the same effect as changing the step size γ (Transtrum and

Sethna, 2012). This results in the descent direction for the Levenberg-Marquardt (LM) method:

$$\boldsymbol{d}_i\left(\boldsymbol{x}_i\right) = -\left(\nabla^2 g_2\left(\boldsymbol{x}_i\right) + \lambda D\right)^{-1} \nabla g_2\left(\boldsymbol{x}_i\right)^\intercal . \tag{2.15}$$

In the following chapters, we routinely minimize the quadratic error of an error function $e(\cdot)$ over N measurements:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \sum_i^N \left\| e_i\left(\boldsymbol{x}\right)\right\|^2 . \tag{2.16}$$

Wrapping the error in a robust cost function ρ (Barron, 2019), like the Huber norm (Huber, 1964), further reduces the effect of outliers during optimization:

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} \sum_i^N \rho\left(\left\| e_i\left(\boldsymbol{x}\right)\right\|^2\right) . \tag{2.17}$$

We stack all errors $e_i\left(\boldsymbol{x}\right)$ into the residual vector $\boldsymbol{r}$, and the quadratic error and its gradient become:

$$\sum_i^N \left\| e_i\left(\boldsymbol{x}\right)\right\|^2 = \boldsymbol{r}^\intercal \boldsymbol{r}, \tag{2.18}$$

$$\nabla\left(\boldsymbol{r}^\intercal \boldsymbol{r}\right) = 2\mathbf{J}^\intercal \boldsymbol{r}, \tag{2.19}$$

$$\mathrm{J}_{ij} = \frac{\partial e_i(\boldsymbol{x})}{\partial x_j}. \tag{2.20}$$

It can be difficult or infeasible to compute the Hessian $\nabla^2(\boldsymbol{r}^\intercal \boldsymbol{r})$ explicitly due to e.g., memory, runtime limitations or when the function itself is unknown. Thus, GN uses the following symmetric approximation:

$$2\mathbf{J}^\intercal \mathbf{J} \approx \nabla^2(\boldsymbol{r}^\intercal \boldsymbol{r}). \tag{2.21}$$

This gives rise to the Normal equations:

$$\mathbf{J}^\intercal \mathbf{J} \Delta\boldsymbol{x} = -\mathbf{J}^\intercal \boldsymbol{r}, \tag{2.22}$$

and $-\Delta\boldsymbol{x}$ becomes the descent direction for GN. Here, solving Eq. 2.22 directly, e.g., via QR decomposition (Higham, 2002), is more numerically stable than the inversion of $\mathbf{J}^\intercal \mathbf{J}$ followed by multiplication with the right-hand side.

For LM, Eq. 2.22 becomes:

$$\left(\mathbf{J}^\intercal \mathbf{J} + \lambda D\right) \Delta\boldsymbol{x} = -\mathbf{J}^\intercal \boldsymbol{r}. \tag{2.23}$$

For D, we use a diagonal matrix with the diagonal entries of $\mathbf{J}^\intercal \mathbf{J}$.

The choice of λ depends on the step quality β (Madsen et al., 2004) of the current update $\Delta\boldsymbol{x}$:

$$f_d = f(\boldsymbol{x}_i) - f\left(\boldsymbol{x}_i \boxplus \Delta\boldsymbol{x}\right), \tag{2.24}$$

$$l_d = \frac{1}{2}\Delta\boldsymbol{x}^\intercal \left(\lambda\Delta\boldsymbol{x} - \boldsymbol{r}\right), \tag{2.25}$$

$$\beta = \frac{f_d}{l_d}. \tag{2.26}$$

λ is initially set to 1×10^{-4} and kept between 1×10^{-16} and 1×10^{16}.

We update λ according to Eq. 2.21 in Madsen et al. (2004). When $\beta < 0$, the update increases the error. Hence, we increase λ:

$$\lambda = \nu\lambda \text{ with } \nu > 1, \tag{2.27}$$

$$\nu = 2\nu, \tag{2.28}$$

to take a shorter step and approach a gradient descent step. Otherwise, we reduce λ:

$$\lambda = \lambda \max\left(\frac{1}{3}, 1 - (2s-1)^3\right), \tag{2.29}$$

$$\nu = 2, \tag{2.30}$$

and accept the current update:

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i \boxplus \Delta\boldsymbol{x}. \tag{2.31}$$

After each update, the non-linearity of $e(\boldsymbol{x}_i)$ requires relinearization at $\boldsymbol{x}_i$ before a new update $\Delta\boldsymbol{x}$ can be computed. We alternate between these two steps for a fixed number of iterations or until one of the following convergence criteria is met:

$$|f(\boldsymbol{x}_i) - f(\boldsymbol{x}_i \boxplus \Delta\boldsymbol{x})| < \theta_e, \tag{2.32}$$

$$|f(\boldsymbol{x}_{i-1}) - f(\boldsymbol{x}_i)| < \theta_e, \tag{2.33}$$

$$\max(\Delta\boldsymbol{x}) < \theta_{\Delta x}. \tag{2.34}$$

The error threshold θ_e is set to 1×10^{-6}, while the maximal coefficient of $\Delta\boldsymbol{x}$ should be above $\theta_{\Delta x} = 1 \times 10^{-8}$.

We often encounter situations where we want to incorporate uncertainty estimates for individual terms. The quadratic cost function becomes a Mahalanobis distance (Kim, 2000) with mean $\boldsymbol{\mu}$ and covariance Σ:

$$f(\boldsymbol{x}) = \|\boldsymbol{x}\|^2_\Sigma = (\boldsymbol{x} - \boldsymbol{\mu})^\intercal \Sigma^{-1} (\boldsymbol{x} - \boldsymbol{\mu}). \tag{2.35}$$

Minimization of Eq. 2.35 is equivalent to solving the weighted least squares:

$$\mathrm{J}^\intercal W \mathrm{J} \Delta\boldsymbol{x} = -\mathrm{J}^\intercal W \boldsymbol{r}, \tag{2.36}$$

if $W = \Sigma^{-1}$ exists. This becomes apparent after applying a Cholesky decomposition to W:

$$W = LL^\intercal, \tag{2.37}$$

$$\mathrm{J}^\intercal \underbrace{LL^\intercal}_{W} \mathrm{J} \Delta\boldsymbol{x} = -\mathrm{J}^\intercal \underbrace{LL^\intercal}_{W} \boldsymbol{r}, \tag{2.38}$$

$$\underbrace{(L^\intercal \mathrm{J})^\intercal}_{\mathrm{J}_d} \underbrace{(L^\intercal \mathrm{J})}_{\mathrm{J}_d} \Delta\boldsymbol{x} = -\underbrace{(L^\intercal \mathrm{J})^\intercal}_{\mathrm{J}_d} \underbrace{(L^\intercal \boldsymbol{r})}_{\boldsymbol{d}}, \tag{2.39}$$

which are the normal equations for:

$$f(\boldsymbol{x}) = \left\| \overbrace{L^\intercal \underbrace{(\boldsymbol{x} - \boldsymbol{\mu})}_{\boldsymbol{r}}}^{\boldsymbol{d}} \right\|^2 = (\boldsymbol{x} - \boldsymbol{\mu})^\intercal \underbrace{LL^\intercal}_{\Sigma^{-1}} (\boldsymbol{x} - \boldsymbol{\mu}). \tag{2.40}$$

2.3 Condition Number

In optimization, we are often confronted with a linear system of equations akin to the normal equations (Eq. 2.22). If the system is sensitive to small perturbations in the input data or due to numerical precision, the computed result may differ strongly from the exact result. In that case, the system is ill-conditioned. To quantify this sensitivity, the matrix condition number is defined in terms of the matrix norm $\|A\|$ (Eq. 6.5 and 6.8 in Higham (2002)):

$$\|A\| = \max_{\|\boldsymbol{x}\|=1} \|A\boldsymbol{x}\|\,, \tag{2.41}$$

$$\kappa(A) = \|A\|\,\|A^{-1}\|\,. \tag{2.42}$$

For a square matrix[1] $A \in \mathbb{R}^{n\times n}$, the Euclidean distance ℓ_2 induces the spectral norm $\|A\|_2$ with spectral radius $\rho(B)$ as the matrix norm $\|A\|$:

$$\|A\|_2 = \rho(A) \text{ with } \rho(B) = |\lambda_{\max}(B)|. \tag{2.43}$$

Thus, it follows from Eq. 2.42 and Eq. 2.43:

$$\kappa(A) = \frac{|\lambda_{\max}(A)|}{|\lambda_{\min}(A)|}. \tag{2.44}$$

Ideally, the condition number $\kappa(A)$ should equal or close to 1, and then we refer to A as well-conditioned. For a s.p.d. matrix $\boldsymbol{\Sigma}$, all Eigenvalues are positive, and the $|\cdot|$ is unnecessary. In some instances, we want to compute the Eigenvalues or κ for the sum of two s.p.d. matrices. For the simple case of $\boldsymbol{\Sigma} + \alpha I$, we directly use:

$$\lambda(\boldsymbol{\Sigma} + \alpha I) = \lambda(\boldsymbol{\Sigma}) + \alpha, \tag{2.45}$$

$$\kappa(\boldsymbol{\Sigma} + \alpha I) = \frac{\lambda_{\max}(\boldsymbol{\Sigma}) + \alpha}{\lambda_{\min}(\boldsymbol{\Sigma}) + \alpha}. \tag{2.46}$$

The first equality follows from the definition of Eigenvectors and Eigenvalues ($A\boldsymbol{v} = \lambda\boldsymbol{v}$, or equivalently $(A - \lambda I)\boldsymbol{v} = \mathbf{0}$):

$$\left((\boldsymbol{\Sigma} + \alpha I) - \overline{\lambda} I\right)\boldsymbol{v} = \mathbf{0}, \tag{2.47}$$

$$\left(\boldsymbol{\Sigma} - \underbrace{\left(\overline{\lambda} - \alpha\right)}_{\lambda_\Sigma} I\right)\boldsymbol{v} = \mathbf{0}, \tag{2.48}$$

$$\overline{\lambda} = \lambda_\Sigma + \alpha. \tag{2.49}$$

Knutson and Tao (2001) provide a more general analysis on the Eigenvalues of the sum of two Hermitian matrices.

1 Higham defines $\|A\|_2$ in a more general setting for a complex matrix $A \in \mathbb{C}^{m\times n}$ and thus requires $\|A\|_2 = \sqrt{\rho(A^*A)}$.

2.4 TRANSFORMATIONS

We define a 3D-point in reference frame F as $\boldsymbol{p}_F = (x, y, z)^\intercal \in \mathbb{R}^3$.

In order to express the point in a different frame, we apply a rigid transform $T_{F_2F_1} \in SE(3)$, which maps the point $\boldsymbol{p}_{F_1}$ from frame F_1 into frame F_2. Rigid transformations (Blanco, 2010) form the Special Euclidean group $SE(n)$:

$$SE(n) = \{(R, \boldsymbol{t}) : R \in SO(n), \boldsymbol{t} \in \mathbb{R}^n\}, \tag{2.50}$$

$$SO(n) = \{R \in \mathbb{R}^{n \times n} : R^\intercal R = I, \det(R) = +1\}. \tag{2.51}$$

The rotation matrix R describes the orientation and vector $\boldsymbol{t}$ the translation. For us, the relevant case is $n = 3$, which has 6 degrees of freedom (DoF), three each for rotation and translation. Without loss of generality, we assume that the reference frame has the identity orientation and its origin at zero. We identify $T_{F_2F_1}$ with its 4×4 matrix operating on homogeneous coordinates, denoted by $[\cdot]$:

$$[\boldsymbol{p}_{F_2}] = T_{F_2F_1}[\boldsymbol{p}_{F_1}], \tag{2.52}$$

$$T_{F_1F_2} = \begin{pmatrix} R & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{pmatrix}. \tag{2.53}$$

This allows multiple transformations to be chained together via matrix multiplication:

$$T_{F_1F_3} = T_{F_1F_2}T_{F_2F_3}. \tag{2.54}$$

Equation 2.52 can be rewritten as:

$$\boldsymbol{p}_{F_2} = R_{F_2F_1}\boldsymbol{p}_{F_1} + \boldsymbol{t}_{F_2F_1}. \tag{2.55}$$

During optimization, the non-linearity of rotations in $SO(3)$ requires linearization. This tangential space $T_R(SO(3))$ for a rotation $R \in SO(3)$ is its corresponding Lie algebra $\mathfrak{so}(3)$:

$$\mathfrak{so}(3) = \{W = -W^\intercal\}, \tag{2.56}$$

$$T_R(SO(3)) = \{RW : W \in \mathfrak{so}(3)\}, \tag{2.57}$$

where W is a skew-symmetric matrix $\in \mathbb{R}^{3\times3}$. The conversion from group to algebra and vice versa uses the logarithmic map at R and its inverse, the exponential map:

$$SO(3) \underset{\exp}{\overset{\log}{\rightleftarrows}} \mathfrak{so}(3), \tag{2.58}$$

$$\log_R : SO(3) \Rightarrow T_R(SO(3)), \tag{2.59}$$

$$\exp_R : T_R(SO(3)) \Rightarrow SO(3). \tag{2.60}$$

The R subscript is generally omitted if R is the identity rotation.

Using the $^\vee$-operator to extract the three unique entries from a skew-symmetric matrix $A \in \mathbb{R}^{3\times3}$ and its inverse, the $^\wedge$-operator $([\cdot]_\times)$:

$$\boldsymbol{a} = A^\vee = [A_{21}, A_{02}, A_{10}]^\intercal, \tag{2.61}$$

$$A = \boldsymbol{a}^\wedge = [\boldsymbol{a}]_\times = \begin{bmatrix} 0 & -a_2 & a_1 \\ a_2 & 0 & -a_0 \\ -a_1 & a_0 & 0 \end{bmatrix}. \tag{2.62}$$

With the $^\vee$- and $^\wedge$ operators, the direct mapping between manifold $SO(3)$ and vector space $\mathbb{R}^3$ and vice versa is given by $\mathrm{Log}(\cdot)$ and its inverse $\mathrm{Exp}(\cdot)$:

$$\mathrm{Log}(R) = \log(R)^\vee = \boldsymbol{\tau} \in \mathbb{R}^3, \tag{2.63}$$
$$\mathrm{Exp}(\boldsymbol{\tau}) = \exp(\boldsymbol{\tau}^\wedge) = R \in SO(3). \tag{2.64}$$

For a rotation matrix R, the $\mathrm{Log}(\cdot)$- and $\mathrm{Exp}(\cdot)$-map are given by Solà et al. (2018):

$$\theta = \arccos\left(\frac{1}{2}(tr(R) - 1)\right), \tag{2.65}$$
$$\boldsymbol{\omega} = \mathrm{Log}(R) = \left[\frac{\theta}{2\sin\theta}(R - R^\intercal)\right]^\vee, \tag{2.66}$$
$$R = \mathrm{Exp}(\boldsymbol{\omega}) = I + \frac{\sin(\|\boldsymbol{\omega}\|)}{\|\boldsymbol{\omega}\|}[\boldsymbol{\omega}]_\times + \frac{1 - \cos(\|\boldsymbol{\omega}\|)}{\|\boldsymbol{\omega}\|^2}[\boldsymbol{\omega}]_\times^2. \tag{2.67}$$

In general, our rotations are expressed by unit-quaternions (Blanco, 2010):

$$\boldsymbol{q} = (q_w, q_x, q_y, q_z)^\intercal \in \mathbb{R}^4 \text{ s.t. } \|\boldsymbol{q}\| = 1. \tag{2.68}$$

Shoemake (1985) provides the corresponding rotation matrix as:

$$R = \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_xq_y + 2q_wq_z & 2q_xq_z - 2q_wq_y \\ 2q_xq_y - 2q_wq_z & 1 - 2q_x^2 - 2q_z^2 & 2q_yq_z + 2q_wq_x \\ 2q_xq_z + 2q_wq_y & 2q_yq_z - 2q_wq_x & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}. \tag{2.69}$$

App. 1.2 of Shoemake (1985) details the conversion from rotation matrix to quaternion. Compared to rotation matrices, quaternions require less parameters and fewer computations for multiplication. To ensure that a quaternion is a valid rotation, only normalization to unit length is necessary:

$$\overline{\boldsymbol{q}} = \frac{1}{\|\boldsymbol{q}\|}\boldsymbol{q}. \tag{2.70}$$

In contrast, for a matrix $A \in \mathbb{R}^{3\times3}$, this involves an orthonormalization with an SVD (Higham, 1989):

$$A = U\Sigma V, \tag{2.71}$$
$$S = \begin{cases} \mathrm{diag}([1, 1, -1]) & \text{if } \det(UV) = -1, \\ I & \text{else}, \end{cases} \tag{2.72}$$
$$R = USV. \tag{2.73}$$

Furthermore, inverting a unit-quaternion rotation is a simple conjugation, which negates the vector component $\boldsymbol{q}_{xyz}$. The $\mathrm{Log}(\cdot)$- and $\mathrm{Exp}(\cdot)$-maps for $\boldsymbol{q}$ are given by:

$$\boldsymbol{\omega} = \mathrm{Log}(\boldsymbol{q}) = \frac{2\arccos(q_w)}{\|\boldsymbol{q}_{xyz}\|}\boldsymbol{q}_{xyz}, \tag{2.74}$$
$$\boldsymbol{q} = \mathrm{Exp}(\boldsymbol{\omega}) = \begin{cases} (1, 0, 0, 0) & \text{if } \boldsymbol{\omega} = \mathbf{0}, \\ \left(\cos\left(\frac{\|\boldsymbol{\omega}\|}{2}\right), \frac{\sin(0.5\|\boldsymbol{\omega}\|)}{\|\boldsymbol{\omega}\|}\boldsymbol{\omega}\right) & \text{else} \end{cases}. \tag{2.75}$$

Here, we omit the corresponding maps for $SE(3)$. An alternative to $SE(3)$ is to separately linearize rotation and translation during optimization (Sommer et al., 2020). Although this split representation ($SO(3) \times \mathbb{R}^3$) does not optimize on the manifold of $SE(3)$ itself, it forms a composite manifold (Solà et al., 2018) and generates valid $SE(3)$ transformations. Multiple studies (Haarbach et al., 2018; Ovrén and Forssén, 2019; Sommer et al., 2020) found improved real-time performance with no significant differences w.r.t. solution quality.

In odometry and simultaneous localization and mapping (SLAM), we regularly need to estimate the transformation $T_{F_2F_1}$ between two sets of points $(\mathcal{P}_{F_1}, \mathcal{P}_{F_2})$ or the projection into an image (Sec. 2.5). Hence, we linearize Eq. 2.52 w.r.t. the 6 DoF transformation parameters $\boldsymbol{\xi} \in \mathbb{R}^6$:

$$\mathrm{J}_T = \frac{\partial \left(T_{F_2F_1} \boldsymbol{p}_{F_1}\right)}{\partial \boldsymbol{\xi}} = \left[I, \left[-\boldsymbol{p}_{F_1}\right]_\times\right], \tag{2.76}$$

where the first three entries of $\boldsymbol{\xi}$ correspond to the translation $\boldsymbol{t}$ and the last three to the rotation R.

2.5 Pinhole Projection

Throughout this work, we assume a pinhole camera model to project 3D points from the camera frame c to the image. Usenko et al. (2018) provide a more comprehensive overview of different camera models. Here, the projection of a point $\boldsymbol{p}$ into a pinhole camera c yields the image coordinates $\boldsymbol{u} = (u_x, u_y)_c^\intercal$ in the image domain $\Omega \subset \mathbb{R}^2$:

$$\pi_c(\boldsymbol{p}_c) : \boldsymbol{p}_c \Rightarrow \boldsymbol{u}_c, \tag{2.77}$$

$$\pi(\boldsymbol{p}) = \frac{K_{2\times3} \cdot \boldsymbol{p}}{p_z}. \tag{2.78}$$

$K_{2\times3}$ are the two upper rows of the pinhole camera matrix K:

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}, \tag{2.79}$$

where f_x and f_y denote the focal length along each image axis, while $\boldsymbol{c} = [c_x, c_y]^\intercal$ defines the principal point. Equation 2.78 can be rewritten as:

$$\boldsymbol{u} = \pi(\boldsymbol{p}) = \begin{pmatrix} \frac{f_x p_x}{p_z} + cx \\ \frac{f_y p_y}{p_z} + c_y \end{pmatrix} = \begin{pmatrix} f_x \overline{u}_x + cx \\ f_y \overline{u}_y + c_y \end{pmatrix}, \tag{2.80}$$

with normalized image coordinates $\overline{\boldsymbol{u}} = [\overline{u}_x, \overline{u}_y]^\intercal = [p_x/p_z, p_y/p_z]^\intercal$. When $p_z < 0$, the point $\boldsymbol{p}$ is behind the image plane. Hence, it is not visible in the image and should be discarded.

The corresponding Jacobian J_π is:

$$J_\pi = \begin{pmatrix} \frac{f_x}{p_z} & 0 & -\frac{f_x p_x}{p_z^2} \\ 0 & \frac{f_y}{p_z} & -\frac{f_y p_y}{p_z^2} \end{pmatrix}. \tag{2.81}$$

Figure 2.1: Camera distortion: straight edges in 3D appear as curved lines in the image. The curvature typically increases with distance to the image center.

Instead of first transforming and then projecting, we can directly combine the K-matrix with a rigid transform $T = [R, \boldsymbol{t}] \in SE(3)$ to obtain the projection matrix $P \in \mathbb{R}^{3\times4}$:

$$P = [KR|K\boldsymbol{t}]\,. \tag{2.82}$$

We then need only one matrix-vector multiplication instead of two. More importantly, this allows efficient computation of the projection for N points simultaneously via a matrix-matrix product after concatenating the points in homogeneous coordinates into a $4 \times N$ matrix. Afterwards, a column-wise division by the last coordinate gives the result for Eq. 2.78. Furthermore, the Jacobian for P directly follows from the chain rule with Eq. 2.81 and Eq. 2.76.

2.6 Distortion

Real camera systems have imperfections and do not follow the standard pinhole model (Furgale et al., 2013; Usenko et al., 2018), as visualized in Fig. 2.1. A commonly adopted solution is to model a radial and tangential distortion with low-order polynomials assuming that the distortion center and the principal point coincide.

Using normalized image coordinates $\overline{\boldsymbol{u}}$, the pinhole projection with radial-tangential distortion becomes:

$$\boldsymbol{u}_d = \text{radtan}\left(\pi\left(\boldsymbol{p}\right)\right), \tag{2.83}$$

$$\boldsymbol{u}_d = \begin{pmatrix} f_x & 0 \\ 0 & f_y \end{pmatrix} \left(\overline{\boldsymbol{u}}_{rad} + \overline{\boldsymbol{u}}_{tan}\right) + \boldsymbol{c}, \tag{2.84}$$

$$\overline{\boldsymbol{u}}_{rad} = \begin{pmatrix} \overline{u}_x \left(1 + k_1 r^2 + k_2 r^4\right) \\ \overline{u}_y \left(1 + k_1 r^2 + k_2 r^4\right) \end{pmatrix}, \tag{2.85}$$

$$\overline{\boldsymbol{u}}_{tan} = \begin{pmatrix} 2p_1 \overline{u}_x \overline{u}_y + p_2 \left(r^2 + 2\overline{u}_x^2\right) \\ p_1 \left(r^2 + 2\overline{u}_y^2\right) + 2p_2 \overline{u}_x \overline{u}_y \end{pmatrix}, \tag{2.86}$$

$$r = \left\|\overline{\boldsymbol{u}}\right\|_2. \tag{2.87}$$

The two parameters k_1, k_2 correspond to the radial while p_1 and p_2 model tangential distortion due to e.g., non-parallel placement of lens and camera sensor.

2.7 ERROR MEASURES

For N error terms stacked into a vector $\boldsymbol{e} \in \mathbb{R}^N$, we define the root-mean-squared (RMS) error (RMSE) and mean absolute error (MAE) as follows (Hodson, 2022):

$$\text{RMSE}\left(\boldsymbol{e}\right) = \sqrt{\frac{1}{N}\boldsymbol{e}^\intercal\boldsymbol{e}} = \sqrt{\frac{1}{N}\sum_{i=0}^{N} e_i^2}, \tag{2.88}$$

$$\text{MAE}\left(\boldsymbol{e}\right) = \frac{1}{N}\sum_{i=0}^{N}|e_i|. \tag{2.89}$$

To evaluate the accuracy of a trajectory $T(\tau) \in SE(3)$ with $\tau \in \mathbb{R}$ w.r.t. some reference, the RMS absolute trajectory error (ATE) (Sturm et al., 2012; Zhang and Scaramuzza, 2018), sometimes called absolute position error (APE) (Grupp, 2017), computes the Euclidean distances for all corresponding pairs $(\boldsymbol{p}_{\text{ref}}, \boldsymbol{p}_\tau)_i$ of reference and trajectory poses after alignment (Umeyama, 1991) with a transformation $X \in SE(3)$:

$$e_{\text{ATE},i} = \left\| X \cdot \boldsymbol{p}_{\tau,i} - \boldsymbol{p}_{\text{ref},i} \right\|_2 . \tag{2.90}$$

We often rely on the RMS-ATE to evaluate the trajectory accuracy.

3

Photometric Online Calibration and Color Correction

Parameter estimation of a camera's vignetting function involves the acquisition of several images in a given scene under very controlled lighting conditions. This is a cumbersome and error-prone task where the result can only be visually confirmed. Many computer vision algorithms assume photoconsistency, i.e., constant intensity between scene points in different images, and tend to perform poorly if this assumption is violated.

We present a real-time online vignetting and response calibration (Quenzel et al., 2018) with additional exposure estimation for global-shutter color cameras. Our method does not require uniformly illuminated surfaces, known texture, or specific geometry. The only assumptions are that the camera is moving, the illumination is static, and the reflections are Lambertian. Our method estimates the camera view poses by sparse Visual-SLAM (VSLAM) and models the vignetting function by a small number of thin plate spline (TPS) together with a sixth-order polynomial to provide a dense estimation of attenuation from sparsely sampled scene points. A TPS models the camera response function (CRF) jointly with a Gamma curve (Ng et al., 2007). We evaluate our approach on synthetic datasets and in real-world scenarios with reference data from a structure-from-motion (SfM) system. We show clear visual improvement on textured meshes without the need for extensive meshing algorithms. A few keyframes are enough to obtain a useful calibration, which makes an on-the-fly deployment conceivable.

Vignetting (Goldman and Chen, 2005; Lauterbach et al., 2017), i.e., the difference in intensity for equally bright scene points in different parts of the image, is an undesirable property of most dioptric camera systems. The aperture, or seldome another set of lenses, blocks a fraction of the incoming light passing through the lens and causes a non-uniform exposure of the photoelectric chip, often increasing in severity towards the outer rim. The effect differs substantially between different lens systems or cameras and may—even if clearly present—be neglected for many applications.

Modeling the relation (Goldman and Chen, 2005; Lauterbach et al., 2017) between the radiance $L_{\boldsymbol{p}}$ of a scene point $\boldsymbol{p} \in \mathbb{R}^3$ with the measured intensity $I_{\boldsymbol{u}}$ of the corresponding image pixel $\boldsymbol{u} \in \mathbb{R}^2$ requires the exposure time k, the position-dependent vignetting $V(\boldsymbol{u})$ and the CRF $f(\cdot)$:

$$I_{\boldsymbol{u}} = f\left(k \cdot V\left(\boldsymbol{u}\right) \cdot L_{\boldsymbol{p}}\right), \tag{3.1}$$

with $V : \mathbb{R}^2 \mapsto [0; 1]$. The CRF $f(\cdot)$ maps between the amount of light reaching the chip and its corresponding measurement. For simplicity, $V(\cdot)$ is often considered radially symmetric around the optical center of the image. We do not restrict ourselves to this assumption. In this work, our objective is to estimate all involved photometric parameters.

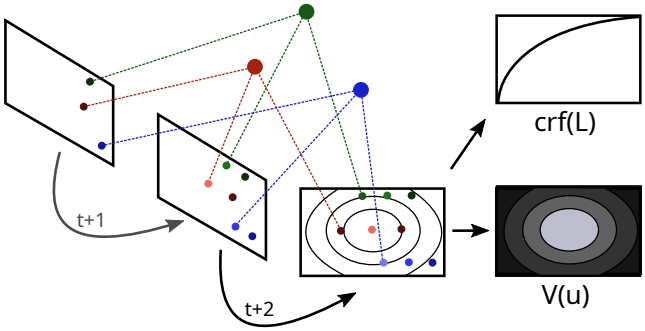

Figure 3.1: Calibration principle: Triangulation of scene points (shown in red, blue and green) minimizes the backprojection error over several frames. Due to vignetting $V(u)$, the measured intensity of the corresponding image points will vary depending on the position $\boldsymbol{u}$ within the frame while the brightness of the entire image is affected by different exposure times k and the camera response function $\mathrm{f}(L)$. The observed attenuations give rise to a camera response and vignetting function.

To this end, we use a robust VSLAM procedure and examine the recorded intensity of well-established map points. Stable triangulation requires the points to be recorded from several camera positions with sufficient parallax. Thus, the corresponding image points are spread over some portion of the camera frames. The intensity of a map point in different regions of the image yields samples for the vignetting function $V(\cdot)$ that is extrapolated with the help of a thin plate spline and a sixth-order polynomial. This approach, illustrated in Fig. 3.1, allows us to obtain a reliable estimate of $V(\cdot)$ quickly based on only a few map points. It does not rely on any known illumination pattern or scene appearance and can be performed by recording natural, albeit textured, environments with static illumination. We demonstrate our method on an image sequence taken with a unmanned aerial vehicle (UAV) to reconstruct a chimney wall structure as well as synthetic sequences with artificial photometric disturbances and show that it significantly improves the results.
In summary, we thoroughly evaluate our photometric online calibration to support our key claims, which are:

- First, combining Gamma curves with TPS improves CRF modeling quality.
- Second, TPS are well suited to capture local deformations within the vignetting that are unhandled by sixth-order radial polynomials (RPs).
- Third, our system enables accurate joint optimization of radiance, vignetting, and camera response in real-time.

Preface

This chapter is adapted from Quenzel et al. (2018), previously published by IEEE and presented at the International Conference on Intelligent Robots and Systems (IROS 2018).

Statement of Personal Contribution

> "The author of this thesis [Jan Quenzel] substantially contributed to all aspects of the previous publication (Quenzel et al., 2018), including the conception, literature survey, design, and implementation of the proposed methods, the preparation of the used data and evaluation of the proposed approach, conducting the experimental evaluation, the analysis and interpretation of the experimental results, drafting the manuscript, as well as the revision and final approval of the version to be published."

3.1 Related Work

There are two main approaches for vignetting correction: Estimating the correction from a single image or from a sequence of images at different view poses with overlapping field-of-view (FoV). The first approach (Zheng et al., 2009; Lopez-Fuentes et al., 2015) requires several necessary assumptions, e.g., uniformly illuminated surfaces, to obtain a viable solution. As such, we prefer the latter approach since it is more robust and enables us to obtain all photometric parameters. Historically, image mosaicking and panoramic photography were some of the first applications. These require stitching multiple images together so that no seam is visible when transitioning between images. Simple vignetting functions, like a sixth-degree polynomial (Goldman and Chen, 2005; Lauterbach et al., 2017), are quite effective in reducing attenuation towards the image borders. In overlapping regions, blending (Zhu et al., 2018) helps to minimize visible seams.

The same challenges occur in mesh texturing. Using graph cuts, Waechter et al. (2014) select the best observing views per mesh face with minimal seams towards neighboring faces. Afterwards, a color adjustment reduces color differences between patches, first globally and later locally, via Poisson Editing to remove remaining visible seams close to the seam itself. However, the reconstructed mesh itself may slightly deviate from the SfM and multi-view stereo (MVS) reconstruction or RGB and depth (RGB-D) point cloud. That is why Zhou and Koltun (2014) optimize the camera poses w.r.t. the mesh such that vertices observed in multiple images exhibit similar intensities. Augmenting the optimization with a non-rigidly deformable grid per image allows us to deal with more complex distortions. Coloring mesh faces then requires subsampling and combining observed colors via a weighted mean.

The photometric or brightness constancy assumption (Park et al., 2017) is central to most direct visual odometry (VO) systems, e.g., semi-direct visual odometry (SVO) (Forster et al., 2014) or direct sparse odometry (DSO) (Engel et al., 2018). Since vignetting and auto-exposure violate the constancy assumption, incorporating photometric calibration improves the accuracy of direct methods, as Zheng et al. (2017) report. Complementary to DSO, Engel et al. (2016) created a monocular camera benchmark, including the photometric calibration of an industry-grade camera. The CRF was calibrated from 1000 images taken by a statically placed camera while manually changing the exposure time in between. Afterwards, vignetting was calculated from images captured while moving the camera along a bright-colored wall with approximate Lambertian reflectance. An attached augmented reality (AR)

marker allowed Engel et al. to estimate the camera pose w.r.t. the planar wall. Given the camera pose, the wall is projected into the images to obtain corresponding intensities at specific points on the wall. The authors then calibrate vignetting and unknown irradiance in an alternating fashion using a maximum likelihood estimator (MLE).

A similar approach by Alexandrov et al. (2016) uses a single white sheet of uniformly illuminated paper as the calibration target for consumer RGB-D sensors. After the CRF estimation (Debevec and Malik, 1997), the camera recorded the paper with fixed exposure and disabled automatic white balance from different vantage points. Then, a floodfill segmentation detects the paper directly without the need for pose estimation or projection. After applying the inverse CRF, observed intensity differences on the sheet should stem purely from vignetting under the assumption of constant illumination throughout the sheet. Alexandrov et al. (2016) then obtain dense vignetting correction factors that outperformed the sixth-order polynomial of Goldman and Chen (2005).

During deployment, the conditions often differ from laboratory settings. Adjusting focus or aperture requires tedious recalibration in the field where a sheet of paper on a planar surface or a tagged wall may not be easily accessible. Online calibration simplifies procedures under such circumstances. For this, Bergmann et al. (2018) employ a Kanade–Lucas–Tomasi tracker (KLT) tracker to find corresponding patches between consecutive images. The correspondences are the basis for optimizing a photometric model as well as the patches' radiance and relative exposure times. The model combines the empirical model of response (EMoR) (Grossberg and Nayar, 2004) as its CRF and the sixth-order RP (Goldman and Chen, 2005) for vignetting. The authors separate the parameter optimization into fast exposure, photometric model, and radiance estimation. Exposure estimation uses a window of ten images compensated for CRF and vignetting. The mean corrected intensity approximates the initial radiance within the window. In parallel, a non-linear optimizer alternates between updating the model and radiance estimates on a window of the last 200 images. The method showed a significant accuracy gain for DSO. However, the authors rely on a completely independent keypoint and motion tracking that does not directly benefit the subsequent VO except for modifying its input. On the other hand, our approach optimizes all parameters jointly and integrates the photometric calibration more straightforwardly and, in particular, avoids two redundant pose estimation and point tracking procedures.

In this thesis, we model the basic shape of vignetting by a sixth-order RP together with a smooth TPS to capture local deformations. The TPS allows for dense approximation from sparsely distributed correction factors derived from corresponding oriented patches around Oriented FAST and Rotated BRIEF (ORB) features. We complement the photometric model with a CRF consisting of a Gamma curve and another TPS with border conditions. Keyframes enable joint optimization of all photometric parameters at runtime while we estimate the current frame's exposure w.r.t. the tracked map directly on arrival. Our algorithm runs online in real-time on a modern laptop central processing unit (CPU) and handles natural and partially dynamic scenes without uniform illumination.

After the original publication (Quenzel et al., 2018), (semi-direct) VO methods increasingly incorporated photometric correction with precalibrated photometric models. Liu et al. (2021) optimize ego-motion and exposure ratio in logarithmic form while optimizing their photometric model in parallel for a fixed number of keyframes. Similarly, Miao et al. (2022) combine the affine brightness model Engel et al. (2018) used with normalized intensities after a logarithmic transform to reduce the difference between bright and dark pixels of the same image. In HSO, Luo et al. (2022) mix two vignetting functions for their correction and adaptively select the best KLT tracker variant based on the image gradients. Miao and Yamaguchi (2021) estimate the exposure ratio from pixel correspondences around matched ORB features with similar radii within the image, where radial vignetting is equally strong. Lin and Zhang (2024) integrate exposure estimation in an iterated error-state Kalman filter (IEKF) for a combined LIO + VIO system and maintain the pointwise radiance within the maps.

If a camera provides its actual exposure time, parameter estimation of CRF and vignetting becomes easier (Haidar et al., 2024). Abboud et al. (2024) integrate such an approach into a VSLAM pipeline.

Unlike previous approaches, Zhu et al. (2021) learn to adapt the image brightness pixel-wise w.r.t. a reference image using a small CNN. However, the CNN does not cope well with occlusion and strong viewpoint variations.

For thermal cameras, Das et al. (2021) investigate online photometric calibration, addressing modality-specific challenges such as temporal drift. Importantly, their correction simplifies odometry (Polizzi et al., 2022) since the standard KLT tracker remains usable.

Another application for photometric models is automatic exposure control. As such, Li et al. (2024) propose to learn exposure based on multi-scale image histograms, whereas Gamache et al. (2024) compare different automatic exposure methods on high dynamic range (HDR) images w.r.t. their influence on VO. Instead, Wang et al. (2022) use precalibrated vignetting and CRF to develop explicit exposure control.

Reconstruction losses in methods like neural radiance field (NeRF) (Mildenhall et al., 2021) or Gaussian splatting (GS) (Kerbl et al., 2023) try to replicate the image color with little deviation. Martin-Brualla et al. (2021) acknowledge that this is insufficient for large-scale datasets recorded with various cameras and variable lighting. Hence, Martin-Brualla et al. (2021) include a low-dimensional appearance embedding within their NeRF that explains image-dependent radiance. More recently, VastGaussian (Lin et al., 2024a) models the appearance variation within an image with a pixelwise multiplier map to prevent floaters in GS. This multiplier map stems from a small per-scene optimized CNN applied to the subsampled image and their pixelwise appearance embedding. Kerbl et al. (2024) instead counteract exposure changes by optimizing a per-image affine transform to modify the color directly . However, NeRF and GS usually require posed images, e.g., from a SfM system like COLMAP (Schönberger and Frahm, 2016) or VGGSfM (Wang et al., 2024b). To reduce the reliance on accurate poses, methods like BARF (Lin et al., 2021) or L2G-NeRF (Chen et al., 2023b) treat the poses as initializations for coarse-to-fine or local-to-global image registration within their respective reconstruction framework. In contrast, “pose-free” approaches (Fan et al., 2024; Fu et al., 2024; Hong et al.,

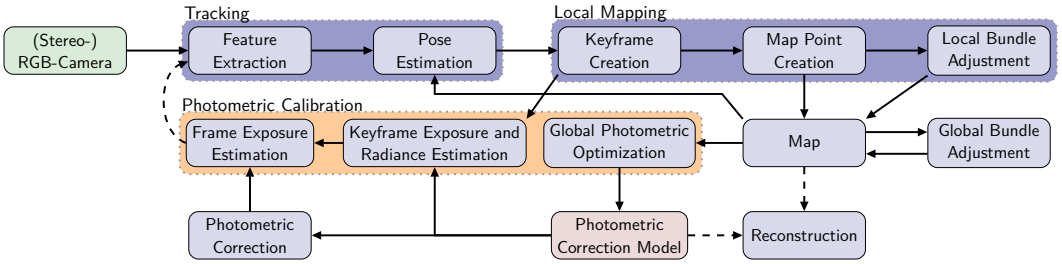

Figure 3.2: Method overview: Visual features are tracked from (Stereo-) RGB images in order to estimate the camera pose w.r.t. the map points. Keyframes and triangulated map points are computed by the local mapping and stored in the map. The photometric calibration (orange dashed box) estimates the exposure time for each frame given the current photometric model $\boldsymbol{\psi}$ (red box) and the tracked map. The radiance estimate of the map is updated on keyframe creation. The photometric model (CRF and vignetting), keyframe exposure times and mappoint radiance are refined given all matched keyframe observations during global photometric optimization. Sparsely, distributed features are used to estimate a pixelwise vignetting correction based on a TPS that can be used to correct future images and reconstruction results.

2025) use feed-forward image matching (Lindenberger et al., 2023; Leroy et al., 2024) or depth estimation (Piccinelli et al., 2024) components to initialize scene geometry and camera poses.

However, as long as correspondences between images are available, our photometric correction remains a possible precursor or intermediate step in the full reconstruction pipeline.

3.2 Our Method

We extend the method from Quenzel et al. (2017) that builds upon a keyframe-based Graph-simultaneous localization and mapping (SLAM) system (Houben et al., 2016) to obtain the color camera trajectory $\mathcal{T}$ and triangulate a sparse feature map $\mathcal{M}$ from ORB features (Rublee et al., 2011). We prefer to use a calibrated and synchronized stereo camera rig or an RGB-D camera in order to avoid monocular scale drift. However, the proposed photometric method also applies to monocular cameras and may be used in conjunction with (inverse-) depth estimates from a (semi-)dense VSLAM system.

Our photometric calibration uses established feature-to-map-point correspondences with additional samples extracted from patches around each feature on their respective scale. As a result, we obtain sparsely distributed samples over the image domain Ω. Hence, we model the vignetting function $V(\cdot)$ as a combination of an even sixth-order polynomial and a smooth TPS, which allows us to estimate the dense attenuation factors for each pixel and color channel. Another one-dimensional TPS with border conditions represents the CRF $f(\cdot)$. Subsequently, we estimate the frame's exposure time k given the current local map $\mathcal{M}_\ell$. As this requires a fast update to the radiance $L_{\boldsymbol{p}}$, we directly refine the radiance of updated map points in $\mathcal{M}_\ell$ after keyframe creation, as shown in Fig. 3.2. In parallel, a global photometric optimization computes

the photometric model $\boldsymbol{\psi} = \{V(\cdot), f(\cdot)\}$ and exposure times asynchronously on all keyframes within the map $\mathcal{M}$.

We will now describe in detail the VSLAM system (Houben et al., 2016; Quenzel et al., 2017), how we establish photometric correspondences, and which correction models use TPS.

In the following, we assume that we observe a static scene with Lambertian reflectance such that the amount of reflected light is independent of the viewing angle. The illumination within the scene should not change over time but may differ locally in the observed scene; we do not assume a uniformly lit scene.

Since we can only obtain the attenuation correction factor for each pixel up to scale, we assume the values to be within $[0, 1]$. We further assume similar attenuation between neighboring pixels.

Furthermore, we assume a rough factory calibration for the intrinsic camera matrices K_c (Eq. 2.79), lens distortion (Sec. 2.6), and the extrinsic transformation $T_{c_i c_j}$ between the cameras $c_i, c_j \in \mathcal{C}$.

3.2.1 *Multi-Camera ORB-SLAM*

The VSLAM system by Houben et al. (2016) adapts the original ORB-SLAM (Mur-Artal et al., 2015), including tracking, local mapping, and loop closing, to operate on images from multiple synchronized cameras $\mathcal{C}$.

Tracking consists of parallel feature extraction for the current simultaneously captured images and subsequent pose estimation, including feature matching w.r.t. the world frame w, e.g., the initial color camera frame. Given the matched features $\mathcal{F}$, each feature with pixel coordinate $\boldsymbol{u}_c \in \mathcal{F}$ provides a constraint in the form of a projection residual $e_{\boldsymbol{u}_c}$ between an observed map point $\boldsymbol{p}_w$ of the local map $\mathcal{M}_\ell$ and the camera pose T_{cw} using Eq. 2.78 with the robust Huber norm (Huber, 1964):

$$e_{\boldsymbol{u}_c} = \rho\left(\left\|\boldsymbol{u}_c - \pi\left(T_{cw}\boldsymbol{p}_w\right)\right\|^2\right). \tag{3.2}$$

A rigid edge $\mathcal{R}$ connects the poses $T_{c_i w}, T_{c_j w}$ of cameras $c_i, c_j \in \mathcal{C}$ with their extrinsic calibration $T_{c_i c_j}$ using the poses' log-map (see Sec. 2.4):

$$T_{c_i c_j} \approx T_{c_i w} T_{c_j w}^{-1}, \tag{3.3}$$

$$\boldsymbol{d}_{\mathcal{R}_{ij}} = \operatorname{Log}\left(T_{c_i w}^{-1} T_{c_i c_j} T_{c_j w}\right), \tag{3.4}$$

$$e_{\mathcal{R}_{ij}} = \boldsymbol{d}_{\mathcal{R}_{ij}}^{\intercal} \Sigma^{-1} \boldsymbol{d}_{\mathcal{R}_{ij}}. \tag{3.5}$$

Such edge enforces the relative pose to remain close to the initial calibration with low uncertainty Σ. However, it is neither fixed nor shared between different images to give the optimization some leeway and flexibility.

Given these residuals, a non-linear optimization scheme updates the current pose estimate for all cameras:

$$\underset{\{T_{cw}\}, c\in\mathcal{C}}{\arg\min} = \left(\sum_{c\in\mathcal{C}} \sum_{\boldsymbol{u}_c\in\mathcal{F}} e_{\boldsymbol{u}_c}\left(T_{cw}\right)\right) + \left(\sum_{c_i\in\mathcal{C}} \sum_{c_j\in\mathcal{C}\setminus c_i} e_{\mathcal{R}_{ij}}\left(T_{c_i w}, T_{c_j w}\right)\right). \tag{3.6}$$

Afterwards, the local mapping decides which images become new keyframes, triangulates new map points and triggers optimization of the map $\mathcal{M}$. After each new

keyframe, a local bundle adjustment (LBA) recomputes the poses for all keyframes and the positions of all map points within the local map $\mathcal{M}_\ell$. For more details, we refer the reader to the corresponding publication (Houben et al., 2016).

After partial exploration of the scene and initial camera tracking, a global bundle adjustment (GBA) refines the keyframe camera poses $T_{cw} \in \mathcal{T}$ and triangulated map points $\boldsymbol{p}_w \in \mathcal{M}$ from undistorted feature observations. In a second optimization step, we include the refinement of lens distortion and intrinsic parameters given the original feature observations while fixing the extrinsic transformation between a stereo camera pair or keeping the first two poses fixed. Thereby, we obtain a more accurate estimate from a factory calibration, allowing us to establish further correspondences between keyframes which have been previously discarded due to high reprojection errors.

3.2.2 *Photometric Correspondences*

For an adequate calibration, we require sufficient pairs of measured $I_{\boldsymbol{u}}$ and expected intensity $\tilde{I}_{\boldsymbol{u}}$, as visualized in Fig. 3.1. Accordingly, we make the following two assumptions. First, all correspondences within an image have the same exposure ratio k. Second, images of a single camera $c \in \mathcal{C}$ share the same CRF and vignetting $V(\cdot)$.

For our measurements, we follow the suggestion of Bergmann et al. (2018) to use a patch centered at the feature location. The feature orientation allows us to extract an unrotated $5 \times 5\,\mathrm{px}$ patch directly at the matched keypoint scale. Alternatively, projection enables the extraction for each map point $\boldsymbol{p}_w \in \mathcal{M}$, e.g., for use with (semi-)direct methods.

To obtain the corresponding expected intensities, each map point maintains a patch of estimated radiance values $\tilde{L}_{\boldsymbol{p}}$. Its initial radiance stems from inverting Eq. 3.1 given the current photometric model $\boldsymbol{\psi}$. Blank and saturated pixels are considered invalid for both intensities, and the pair is subsequently discarded.

In general, this leads to acceptable numbers of constraints on the exposure ratio and CRF. However, vignetting depends on the pixel position, and having multiple pairs for a single pixel rarely happens, especially for sparsely distributed map points. Relying on gathering a large number of pixel correspondences would further prevent our method from running online. Hence, the TPS is a convenient method to interpolate vignetting in between.

3.2.3 *Attenuation Model*

Following Eq. 3.1, knowing the exposure ratio k and CRF $f(\cdot)$ allows us to calculate an attenuation factor $s_{\boldsymbol{u}}$ for the observed pixel in the current image:

$$s_{\boldsymbol{u}} = \tilde{L}_{\boldsymbol{p}} / \left(k f^{-1}\left(I_{\boldsymbol{u}}\right)\right). \tag{3.7}$$

This simplifies with constant exposure and identity mapping ($f(I) = I$) to:

$$s_{\boldsymbol{u}} = \tilde{I}_{\boldsymbol{u}} / I_{\boldsymbol{u}} \propto V\left(\boldsymbol{u}\right). \tag{3.8}$$

Intuitively, a factor greater than one means the pixel is brighter; smaller than one means darker.

We use a TPS to model local attenuation factors w.r.t. the normalized color image coordinates from $[0,1]^2 \in \Omega$. Due to the excellent fill-in property and the minimal bending energy of these splines, this works even with scattered, sparsely distributed data—in our case, the correction factors and corresponding image positions—while giving smooth function approximations with a small number of coefficients.

We use N radial basis functions (RBFs) $\phi(r)$ placed at control points $\boldsymbol{d}_i \in \Omega$ and the polynomial $q(\boldsymbol{u})$ with coefficients $\boldsymbol{v} \in \mathbb{R}^3$:

$$\phi(r) = r^2 \cdot \ln(r), \tag{3.9}$$

$$q(\boldsymbol{u}) = \boldsymbol{v}^\intercal \cdot \begin{pmatrix} 1 \\ \boldsymbol{u} \end{pmatrix}, \tag{3.10}$$

to define the following two-dimensional polyharmonic TPS with coefficients $\boldsymbol{c} \in \mathbb{R}^N$:

$$h(\boldsymbol{u}) = q(\boldsymbol{u}) + \sum_{i=1}^{N} c_i \cdot \phi(\|\boldsymbol{u} - \boldsymbol{d}_i\|). \tag{3.11}$$

Here, $\boldsymbol{u}$ is the data point—a pixel coordinate. The parameters $\boldsymbol{c}$ control the influence of the RBF, while $\boldsymbol{v}$ aids the approximation as a polynomial. One advantage of the TPS is the lack of parameters that have to be tuned since $\boldsymbol{c}, \boldsymbol{v}$ are calculated from the given image positions $\boldsymbol{u}$ and the desired function values, the correction factors $s_{\boldsymbol{u}}$. Furthermore, a TPS is far more flexible than a polynomial with the same number of coefficients.

In the case of interpolation, one seeks to find the coefficients $[\boldsymbol{c}, \boldsymbol{v}]^\intercal$ s.t. the following equations are satisfied:

$$s_i = h(\boldsymbol{u}_i), 1 \leq i \leq M. \tag{3.12}$$

Since the interpolation would require as many RBFs (N) as there are data points (M), this cannot be used efficiently online. Instead, we approximate the underlying function using a grid with a small fixed number of $N = J \times K$ control points:

$$\arg\min_{\boldsymbol{c},\boldsymbol{v}} \sum_{i}^{M} \|h(\boldsymbol{u}_i) - s_i\|^2. \tag{3.13}$$

On each control point $\boldsymbol{d}_i$, one RBF is placed statically. We typically choose $Q_\mathrm{a}, Q_\mathrm{b} \in \{3, \ldots, 7\}$, but other choices and different grids are possible as well. Adding the following conditions:

$$\sum_i^N c_i = 0, \sum_i^N c_i \cdot d_{i,x} = 0, \sum_i^N c_i \cdot d_{i,y} = 0, \tag{3.14}$$

ensures the approximation of the polynomial $q(\cdot)$.

To minimize the least-squares error function Eq. 3.13 such that Eq. 3.14 is satisfied, we have to solve the augmented system of equations, stated in matrix form as:

$$\begin{pmatrix} A & X \\ D & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{c} \\ \boldsymbol{v} \end{pmatrix} = \begin{pmatrix} \boldsymbol{s} \\ \boldsymbol{0} \end{pmatrix}, \tag{3.15}$$

$$A = \begin{bmatrix} \phi\left(|\boldsymbol{u}_1 - \boldsymbol{d}_1|\right) & \dots & \phi\left(|\boldsymbol{u}_1 - \boldsymbol{d}_N|\right) \\ \vdots & \ddots & \vdots \\ \phi\left(|\boldsymbol{u}_M - \boldsymbol{d}_1|\right) & \dots & \phi\left(|\boldsymbol{u}_M - \boldsymbol{d}_N|\right) \end{bmatrix}, \tag{3.16}$$

$$X = \begin{pmatrix} 1 & \boldsymbol{u}_1 \\ \vdots & \vdots \\ 1 & \boldsymbol{u}_M \end{pmatrix}, D = \begin{pmatrix} 1 & \dots & 1 \\ \boldsymbol{d}_1 & \dots & \boldsymbol{d}_N \end{pmatrix}. \tag{3.17}$$

This over-determined system ($M \gg N$) may be solved with the conjugate gradient (CG) method. In contrast to a QR-decomposition, CG allows the use of an initial solution, e.g., from a previous run, speeding up online computation. This simplifies the usage of iterative reweighted least squares (IRLS) (Holland and Welsch, 1977) for outlier rejection with sufficient data points.

Regarding the vignetting correction, we replace the polynomial in Eq. 3.11 with the sixth-order even RP of Goldman and Chen (2005) placed at the center of the unit square $\boldsymbol{d}_m$ and normalization factor $\sqrt{2}$:

$$q\left(r\right) = 1 + \sum\nolimits_{i=1}^{3} v_i \cdot r^{2i}, \tag{3.18}$$

$$h\left(\boldsymbol{u}\right) = q\left(\sqrt{2}\left\|\boldsymbol{u} - \boldsymbol{d}_m\right\|\right) + \sum_{i=1}^{N} c_i \cdot \phi\left(\left\|\boldsymbol{u} - \boldsymbol{d}_i\right\|\right). \tag{3.19}$$

The polynomial $q(\cdot)$ matches the general form of the vignetting function, whilst the RBFs cover higher-order and local deformations.

3.2.4 *Camera Response Function*

We employ a one-dimensional TPS with a linear function (Eq. 3.10) for the camera response. Since the CRF $f(\cdot)$ needs to interpolate from zero to one, we add constraints that enforce $f(0) = 0$ and $f(1) = 1$ and distribute the Q control points equidistantly over the whole domain $[0, 1]$. This corresponds to a partial differential equation (PDE) with Laplace Equation ($\nabla^2 f = 0$) and Dirichlet boundary conditions (Chen, 1992), the solution of which is obtained by solving a linear system of equations.

A problem-specific function for $q(\cdot)$ increases the TPSs' fitting accuracy further compared to the polynomial (Eq. 3.10). Hence, we employ the nth-order generalized gamma curve model (GGCM) by Ng et al. (2007):

$$p_f\left(L\right) = \sum\nolimits_{i=0}^{n} v_i \cdot L^i, \tag{3.20}$$

$$f_{\mathrm{GGCM}}\left(L\right) = L^{p_f(L)}, \tag{3.21}$$

$$g_{\mathrm{GGCM}}\left(I\right) = I^{1/p_f(I)}. \tag{3.22}$$

We propose two new CRF model functions based on the GGCM. Firstly, we use the model $f_{\text{GGCM}}(\cdot)$ and add several polyharmonic TPSs. We term this model GGCM + TPS. Secondly, we use the GGCM model with a more variable TPS instead of the polynomial in Eq. 3.20 and name this model GGCM$^{\text{TPS}}$.

This results in the following functions:

$$f_{\text{TPS}}(L) = \sum_{i=1}^{Q} c_i \cdot \phi(|L - d_i|), \tag{3.23}$$

$$f_{\text{GGCM+TPS}}(L) = f_{\text{TPS}}(L) + L^{p_f(L)}, \tag{3.24}$$

$$f_{\text{GGCM}^{\text{TPS}}}(L) = L^{(p_f(L)+f_{\text{TPS}}(L))}. \tag{3.25}$$

We do not choose the function $g_{\text{GGCM}}(\cdot)$ due to the inversion of the polynomial. The reader may note that $g_{\text{GGCM}}(\cdot)$ is not the correct inverse of $f_{\text{GGCM}}(\cdot)$, except for a pure gamma curve ($n = 0$). Nevertheless, given an estimate of the polynomial, it is easy to calculate a corresponding $g_{\text{GGCM}}(\cdot)$. Once an updated CRF model is available, we sample the CRF equidistantly and optimize for the parameters of the inverse model $f^{-1}(\cdot)$ using $g_{\text{GGCM}}(\cdot)$. So far, we have not seen the necessity of explicitly enforcing monotonicity, as the results were monotone with $Q \in [5, 20]$ control points. However, Utreras and Varas (1991) give an alternative method for monotone TPS.

3.2.5 *Image Correction*

Given the solution to Eq. 3.13, we obtain the fitted TPS by evaluating Eq. 3.19 for each pixel. To remove the vignetting, the inverse CRF needs to be applied to the pixel intensity, followed by multiplication with the inverse attenuation factor and exposure time:

$$\tilde{I}_{\boldsymbol{u}} = f^{-1}(I_{\boldsymbol{u}}) \,/\, [V(\boldsymbol{u}) \cdot k]. \tag{3.26}$$

Here, $\tilde{I}_{\boldsymbol{u}}$ denotes the corrected pixel intensity. For 8-bit images, a look-up-table (LUT) for $f^{-1} \in [0, 255]$ can be easily obtained due to the strict monotonicity of the CRF. The inverse attenuation factors $V(\boldsymbol{u})^{-1}$ require pixelwise evaluation of the TPS only once. Since vignetting is only estimated up to scale, the division of pixelwise attenuation factors by the maximal attenuation ensures the factor's unit range.

3.2.6 *Keyframe-based Photometric Calibration*

Incremental refinement optimizes the photometric calibration from a number of keyframes. Here, we update the entire parameter set in a way similar to Bergmann et al. (2018). Given an initial guess for the radiance $L_{\boldsymbol{p}}$, the vignetting $V(\boldsymbol{u})$, the exposure time k, and the CRF $f(\cdot)$, the global photometric optimization jointly minimizes the difference between the measured image intensity $I_{\boldsymbol{u}}$ and the righthand side of Eq. 3.1 all involved parameters:

$$\underset{\boldsymbol{\psi}, \boldsymbol{k}, \boldsymbol{L_p}}{\arg\min} \sum_{i,j,o=1}^{N,M,O} \rho\left(\left\| w_j \left[I(\boldsymbol{u}_j) - f_{\boldsymbol{\psi}}\left(k_i V_{\boldsymbol{\psi}}(\boldsymbol{u}_j) L_{\boldsymbol{p}_o}\right)\right]\right\|^2\right). \tag{3.27}$$

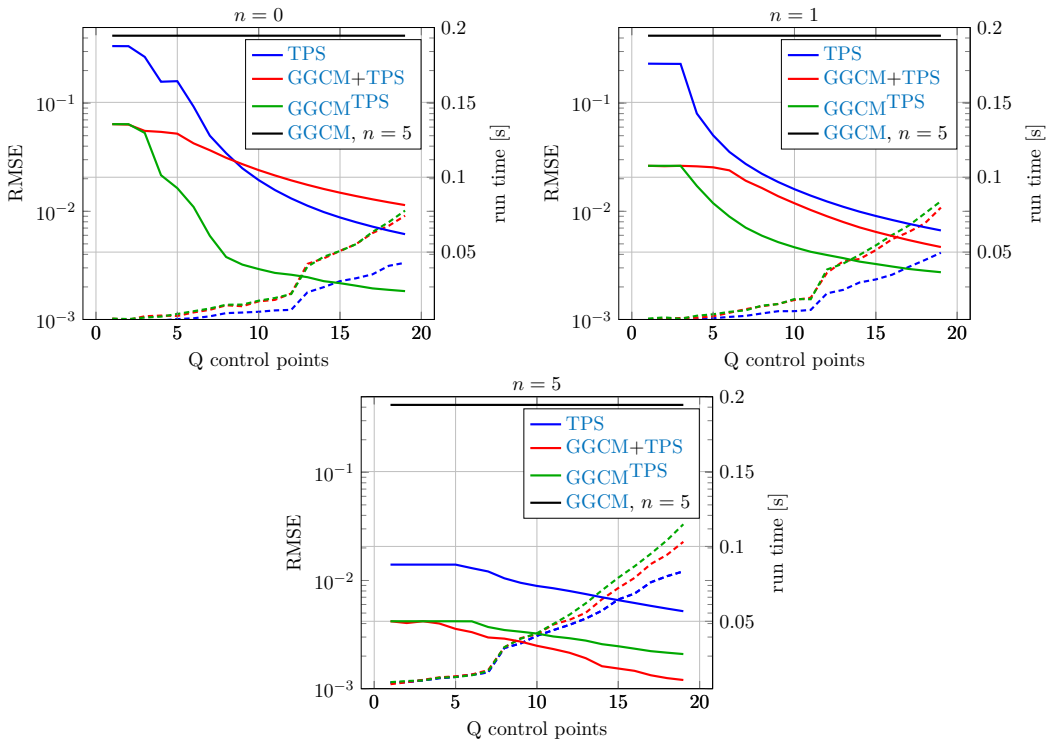

Figure 3.3: Results of the proposed CRF-models for varying polynomial degrees $n = \{0, 1, 5\}$. Dashed curves show runtime according to right-hand axis label. Lower values are better (↓).

Here, N denotes the number of keyframes, M is the number of observations, e.g., 25 pixels per patch, and O is the number of map points. The robust Huber loss ρ with $\alpha = 0.2/255$ reduces the influence of outliers (Triggs et al., 2000). An additional weighting term w_j with $\eta = 1$:

$$w_j = \frac{\eta}{\eta + \|\nabla I\left(\boldsymbol{u}_j\right)\|^2} \tag{3.28}$$

downweighs high gradient pixels. Removal of blank (0) and saturated pixels (255) prevents biasing the optimization towards the start and end of the value range where pixel values don't allow for fine gradation. The remaining pixel values are scaled to the floating-point unit range for further processing.

Exposure time evaluation for individual frames requires the current radiance estimate of the $\tilde{O}$ tracked map points:

$$\arg\min_{k} \sum_{j,o=1}^{M,\tilde{O}} \rho_h \left(\left\| w_j \left[k \cdot L_{\boldsymbol{p}_o} - \frac{f_{\boldsymbol{\psi}}^{-1}\left(I_{\boldsymbol{u}_j}\right)}{V_{\boldsymbol{\psi}}\left(\boldsymbol{u}_j\right)} \right] \right\|^2 \right). \tag{3.29}$$

As global photometric optimization is infeasible under real-time constraints, we refine the exposure time k of a keyframe and the radiance of all its tracked and newly created map points $L_{\boldsymbol{p}}$ on their creation. The refinement (Eq. 3.27) runs asynchronously upon keyframe creation once a certain number of keyframes exist.

In contrast to the depth calibration of Quenzel et al. (2017), we do not rescale keyframes to remove vignetting within our SLAM system due to the properties of

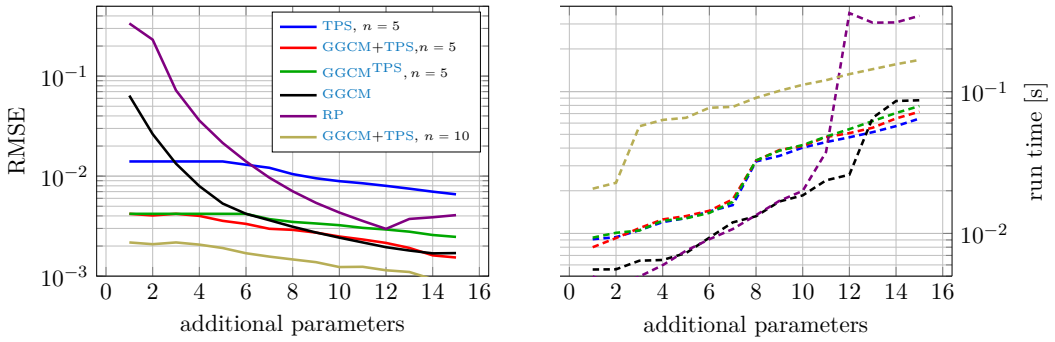

Figure 3.4: The RMS error (RMSE) and runtime results of the proposed CRF-models for varying number of additional parameters. Lower values are better (↓).

features like ORB (Rublee et al., 2011). Such features are designed to be robust against small brightness variations. Hence, we assume the influence for feature-based SLAM to be small and do not perform the costly feature extraction a second time. However, the recomputation of feature descriptors on corrected images is a viable solution. For (semi-)direct methods, where the fundamental assumption is constant brightness between corresponding pixels, vignetting removal improves the accuracy (Engel et al., 2016), and rescaling is suggested.

3.3 Evaluation

First, we test our approach on the RGB-D sequences of the synthetic ICL-NIUM (Handa et al., 2014) dataset with modified vignetting and sinusoidally varying exposure times. Afterwards, our evaluation uses real-world sequences captured by a stereo rig attached to a UAV. The rig consists of two synchronized FLIR Blackfly S BFS-U3-51S5 color cameras with a resolution of 2448×2048 px and calibrated with Kalibr (Furgale et al., 2013). During our experiments, we activated auto exposure without gain and recorded the images at 22 Hz as well as the cameras' exposure times for ground-truth evaluation.

We compare our results on real-world sequences against the methods by Engel et al. (2016), Alexandrov et al. (2016), and Bergmann et al. (2018). These sequences were captured in a lab and a hallway with stonework. All algorithms were running on an Intel Core i7-6700 HQ with 32 GB RAM with Ubuntu 16.04. Vignetting and CRF calibration start after the tenth keyframe becomes available. The control points are set on a 4×5 regular grid to incorporate the image's aspect ratio.

3.3.1 *Camera Response Function*

The DoRF dataset (Grossberg and Nayar, 2004) provides 201 response curves of real cameras. To evaluate our TPS-CRF, we perform a least squares fit for each camera using the Ceres-Solver (Agarwal et al., 2022) and compute the RMSE for each CRF. For selected configurations, we report the RMSE and the running time in Fig. 3.3 and Fig. 3.4. We limit the number of tested TPS parameters to 20 and the polynomial order to 15 and evaluate three different CRFs: TPS with polynomial $q(\cdot)$ as well as

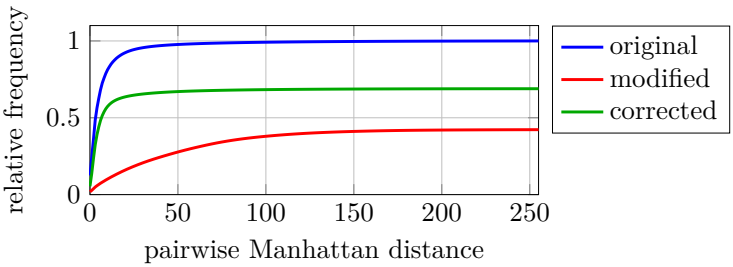

Figure 3.5: Cumulative histogram of the pairwise Manhattan distances between fused points after dense reconstruction using COLMAP (Schönberger et al., 2016) on the ICL-NUIM dataset. A higher curve is better.

with the GGCM (Eq. 3.21), denoted as TPS + GGCM, and a modified GGCM in which a TPS replaces the polynomial, denoted as GGCM^{TPS}. Prior to optimization, the initialization sets all functions to interpolate between zero and one linearly.

We showed that all the presented models can successfully fit real-world camera response functions—even though the total number of parameters increases with additional TPSs. Nevertheless, a trade-off exists between the number of TPS parameters and the time required to fit a higher-order polynomial model. By adding a number of TPS, we can lower the polynomial degree and obtain a better RMSE while using less time to fit the model.

The combination of TPS + GGCM provides the best results, followed by GGCM^{TPS}, while the classical TPS performs worst and takes the longest to optimize. We attribute the accuracy of TPS + GGCM compared to only GGCM to the additional flexibility of the TPSs. The deficit of the original TPS stems from the polynomial, which is not an appropriate model for Gamma-like curves. Still, it reaches the same error as GGCM ($n = 15$) with 60 control points while taking three times longer.

3.3.2 *Vignetting Correction*

We perform multiple experiments with different vignetting masks. The first vignetting mask is the ideal case with a pure sixth-order RP (Eq. 3.18) originating from the image's center. The second vignetting mask has a randomly shifted origin while the third mask contains slight deformations from locally consistent noise. Here, we compare a pure polynomial against a TPS (Quenzel et al., 2019) and our combination (Eq. 3.19).

We employ the MVS pipeline of COLMAP (Schönberger et al., 2016) to create a dense reconstruction given previously selected keyframes. The reconstruction runs separately on the *original*, the *modified*, and the *corrected* image sets. The modified images exhibit alterations for vignetting and synthetic sinusoidal exposure changes as described above. We applied our estimated correction on these images to obtain the corrected set. Figure 3.5 visualizes the difference in mean Manhattan distance between fused points of the reconstruction. A smaller distance is preferable since correspondences in multiple images should be similar. The reconstruction from

Table 3.1: RMSE of vignetting, exposure and mean improvement of consistent feature matches on Monte Carlo sampled synthetically deteriorated sequence of ICL-NUIM (Handa et al., 2014) without loop closure. Second and **best** are highlighted.

Model		RP	TPS	TPS + RP
Vignetting RMSE	(↓)	0.0527	0.0473	**0.0462**
Exposure RMSE	(↓)	0.0419	0.0347	**0.0335**
Improvement [%]	(↑)	11.60	**13.25**	11.98

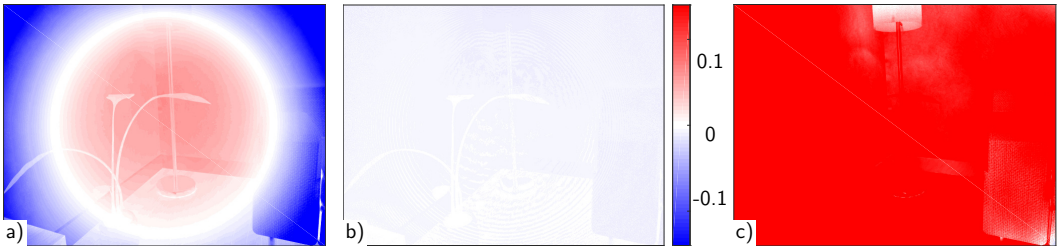

Figure 3.6: Difference between original and corrected image for image 858 from ICL-NUIM living room 2 (Handa et al., 2014). a) Vignetting is evident in the modified image. b) Our correction successfully reduces vignetting, exposure change, and removes the response function. c) The estimated exposure is too high using the method of Bergmann et al. (2018).

corrected images follows the original graph closely for small differences (< 10), whereas the modified sequence exhibits significant differences.

We further deteriorated the vignetting by moving its origin away from the image center and added low spatial-frequency noise. We jointly optimized the vignetting and the exposure time on the same keyframes and computed the RMSE. We repeated this procedure one hundred times for the sixth-order RP (Eq. 3.18), the original TPS (Eq. 3.11), and our radial TPS (Eq. 3.19). The results are shown in Tab. 3.1. As expected, our radial TPS performs best, followed by the original TPS and the radial polynomial. An improved vignetting estimate simultaneously reduces the difference between estimated and correct exposure time. However, increasing the number of TPSs reduces the RMSE at the expense of increased optimization and run time. After optimization, we correct all keyframes and recompute feature descriptors and matches. The ground-truth poses allow checking the matches' consistency. Surprisingly, the number of correct correspondences increases after our compensation by around 12 %, improving the overall system accuracy. Most newly found correspondences were previously slightly above the matching threshold due to the image deterioration.

3.3.3 *Synthetic Datasets*

A drawback of the previously mentioned TPS-CRF is its missing closed-form invertibility. Hence, we choose to use the $GGCM^{\mathrm{TPS}}$ as our CRF model for the integrated

Table 3.2: RMSE of all optimized parameters on synthetically deteriorated sequence of ICL-NUIM (Handa et al., 2014) w/o loop closure. Lower values are better (↓) with second and **best** highlighted.

Model	CRF	$GGCM^{TPS}$			Bergmann et al.
	Vignetting	RP	TPS	TPS + RP	
RMSE	Exposure	0.0749	0.0331	**0.0292**	0.1761
	Vignetting	0.0559	0.0381	**0.0367**	0.1029
	CRF	0.0309	0.0268	**0.0210**	0.1468
$RMSE_{10}$	Exposure	0.0186	**0.0127**	0.0141	0.0567

tests on the synthetic and real-world datasets, which is also relatively fast to optimize. The TPS uses five control points and a second-degree polynomial. We evaluate all three vignetting models and use a grid size of 4×5 for the TPS. Table 3.2 reports the corresponding RMSE for the exposure ratio, CRF, and vignetting. Additionally, we compare the approach of Bergmann et al. (2018) with default parameters and the number of active frames set to the sequence length. As evident in the visualization of the difference between original and corrected estimates in Fig. 3.6 (right), the result exhibits strong drift in the exposure estimate. Hence, we also report the $RMSE_{10}$ over a smaller window of ten frames. We attribute our improvement to the joint optimization, in contrast to alternating between radiance and photometric parameters.

3.3.4 *Real-World Datasets*

We followed the prescribed calibration procedures for the methods of Engel et al. (2016) and Alexandrov et al. (2016). Figure 3.7 visualizes the corresponding inverse CRF curves. During our tests, we found the white-paper method to be sensible to lighting conditions, such as mixtures of artificial and natural light. The method by Engel et al. may produce non-monotonic camera response functions.

Without knowledge about the absolute radiance, exposure time estimation is only up to scale (Bergmann et al., 2018). Hence, we show the exposure ratio, e.g., relative to the first image $\frac{k_i}{k_0}$, in Fig. 3.8 for the first stereo camera on the lab sequence. We align the estimates from our method (green dots) and the approach by Bergmann et al. (2018) (red dots) with least-square fits considering the exponential ambiguity for unknown CRF and a multiplicative factor for the exposure ratio as proposed by Bergmann et al. The results are similar for the second camera but excluded here for brevity.

The sample texture in Fig. 3.9 extracted from the wall sequence shows clear visual improvements. The seams disappear, and the colors become more uniform and consistent.

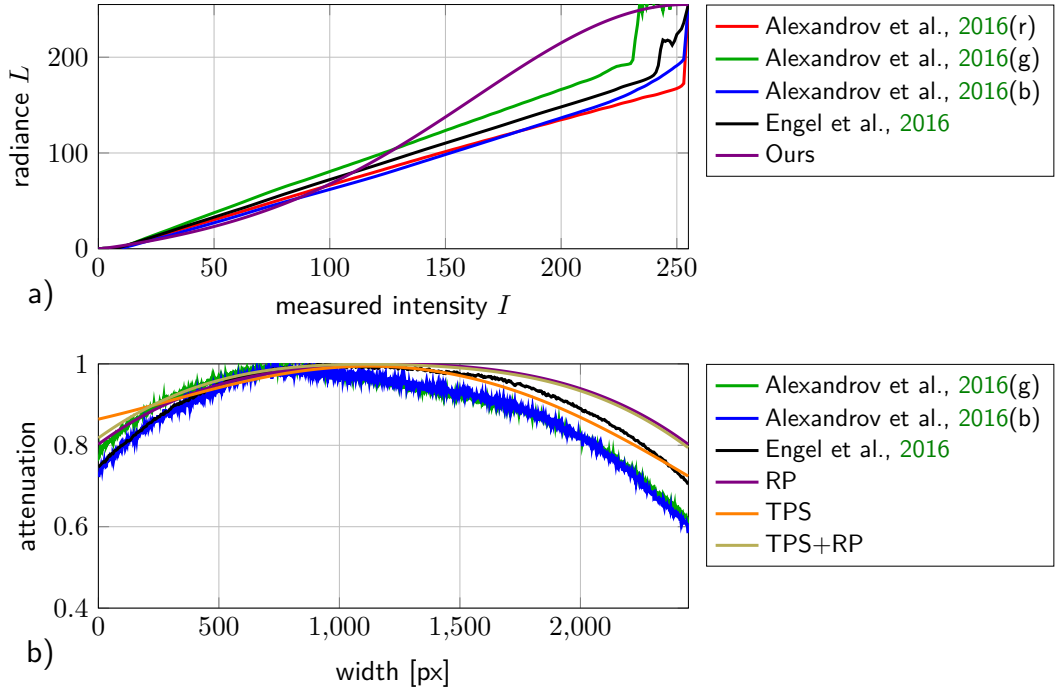

Figure 3.7: Comparison on inverse CRFs and vignetting. a) Estimated inverse CRFs. b) Cross section of vignetting masks along the central row. Results using the method by Alexandrov et al. (2016) are reported for each color channel (r,g,b). The vignetting estimates for radial polynomial (RP), TPS and the combination (TPS+RP) and our CRF were calculated on the lab sequence.

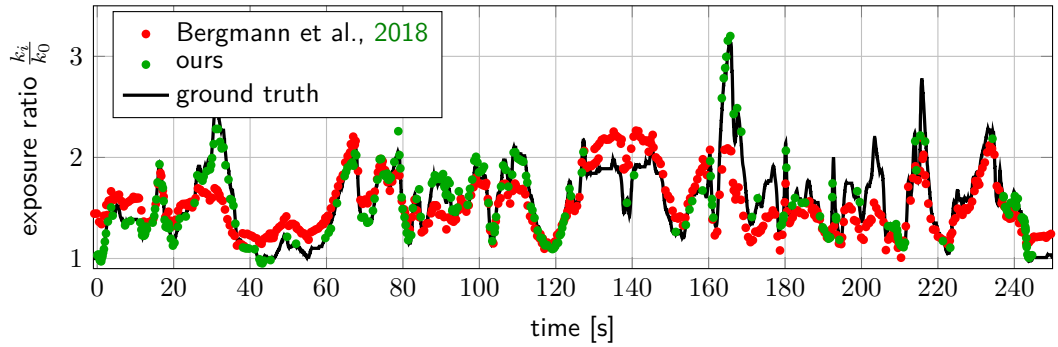

Figure 3.8: Exposure ratios on the lab sequence for one camera. The estimated ratio of the keyframes (dots) follows accurately the real exposure ratio of the camera. Shown are the optimized keyframe estimates.

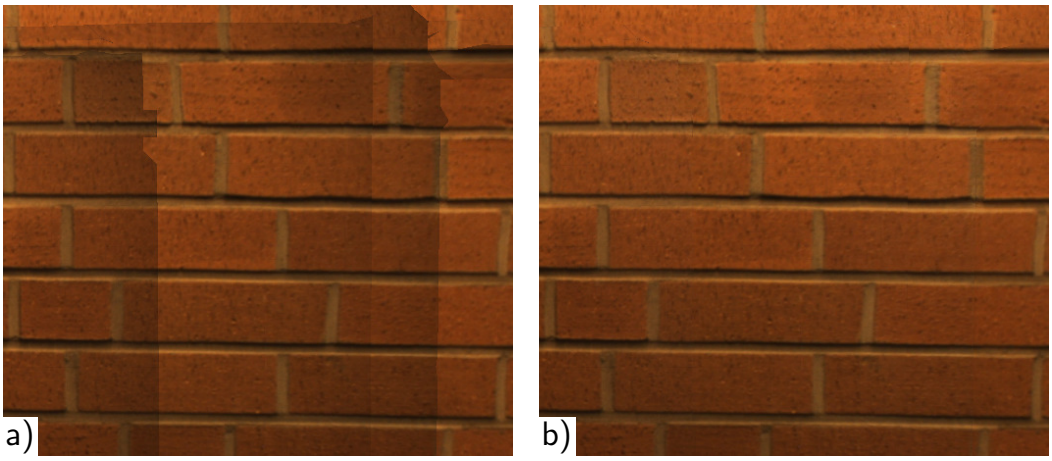

Figure 3.9: Reconstructed texture of a brick wall. The uncorrected approach exhibits brightness differences. Our corrected approach shows substantial improvement.

3.4 Summary

In this chapter, we presented a fast and easy-to-use photometric calibration method and verified our key claims. Our system enables accurate joint optimization of radiance, vignetting, and camera response in real-time without the need for white or evenly illuminated surfaces, calibration targets, or known scene geometry. For this, we obtain oriented patches centered at the keypoint's location from matched ORB features. These sparsely distributed samples are sufficient to optimize all photometric parameters on a set of keyframes. Our fast exposure estimation further computes the exposure ratio for every other frame. We employ thin plate splines with a sixth-order radial polynomial to approximate the attenuation factors w.r.t. the image position and to obtain pixelwise vignetting correction factors. The TPS captures local deformations within the vignetting that are unhandled by the sixth-order radial polynomial. Similarly, we improve the CRF modeling quality of Gamma curve models with a TPS as verified on the DoRF CRF dataset (Grossberg and Nayar, 2004).

Our method outperformed competitors on publicly available synthetic and our real-world sequences. Moreover, feature matching quality and image similarity improve after compensation, reducing seams and borders between fused images and enabling more accurate reconstruction in MVS applications. The experimental results further substantiate that the calibration converges quickly and effectively corrects vignetting and likewise estimates the camera response function, exposure times, and scene radiance. The fitting approach works well with different models of varying complexity and, thus, allows us to cover non-standard camera configurations as well. Due to its straightforward implementation and fast convergence, our contribution can serve as a general initialization stage for robot vision algorithms on mobile platforms, which can then quickly adapt to the current camera setup.

4

Gradient-based Dissimilarity

Pose estimation and map building are key ingredients for autonomous mobile robots or other intelligent vehicles and require the registration of sensor data, e.g., of camera images or LiDAR point clouds. For sequential images, visual odometry (VO) refers to the estimation of the camera's trajectory with several approaches present in the literature (Kerl et al., 2013b; Engel et al., 2014; Forster et al., 2014; Mur-Artal and Tardós, 2017; Engel et al., 2018). The gold standard for computing the relative orientation between two monocular images of a calibrated camera is Nister's 5-point algorithm (Nistér, 2004), which calculates the 5-DoF transformation from five feature correspondences. In practice, it is preferable (Moulon et al., 2013, 2016) to use more points and combine random sample consensus (RANSAC) with the 5-point algorithm and a subsequent least-squares inlier refinement step. Although features are by design resilient against changes in the intensity values of the images, for example, caused by variations in illumination, their extraction can be a time-consuming operation while images commonly contain only a sparsely distributed set of features.

Direct alignment is an alternative approach using comparisons of the pixel intensity values within the image pair instead of explicit feature correspondences since the intensity values of each pixel are directly accessible raw measurements. One often distinguishes depending on the amount of compared pixels between semi-dense (Forster et al., 2017; Park et al., 2017; Engel et al., 2018) and dense methods (Newcombe et al., 2011b; Kerl et al., 2013a). Most direct methods consider the so-called photometric consistency of the image as the objective function to optimize. Similarly, the direct approach has applications in stereo matching or multi-view stereo (MVS) to estimate the scene depth from pairs or multiple images. Achieving robustness is a key challenge for direct approaches because slight variations in illumination, of the camera exposure, due to vignetting, or motion blur directly affect the intensity measurements, as visualized in Fig. 4.1. Reliable and accurate estimates of VO and depth estimation are important for real-world applications especially in diverse environments.

In this chapter, we address the problem of robustifying the direct alignment of image pairs through a new dissimilarity metric for registering images that builds up on the idea of the photometric error and directly improves depth estimates and alignment of image sequences. Our approach (Quenzel et al., 2020) combines a gradient orientation-based metric proposed by Haber and Modersitzki (2006) with a magnitude-dependent scaling term. We furthermore integrate our metric into four different stereo estimation and VO systems [OpenCV, MeshStereo (Zhang et al., 2015), DSO (Engel et al., 2018), and Basalt (Usenko et al., 2020)] to show that our metric leads to improvements.

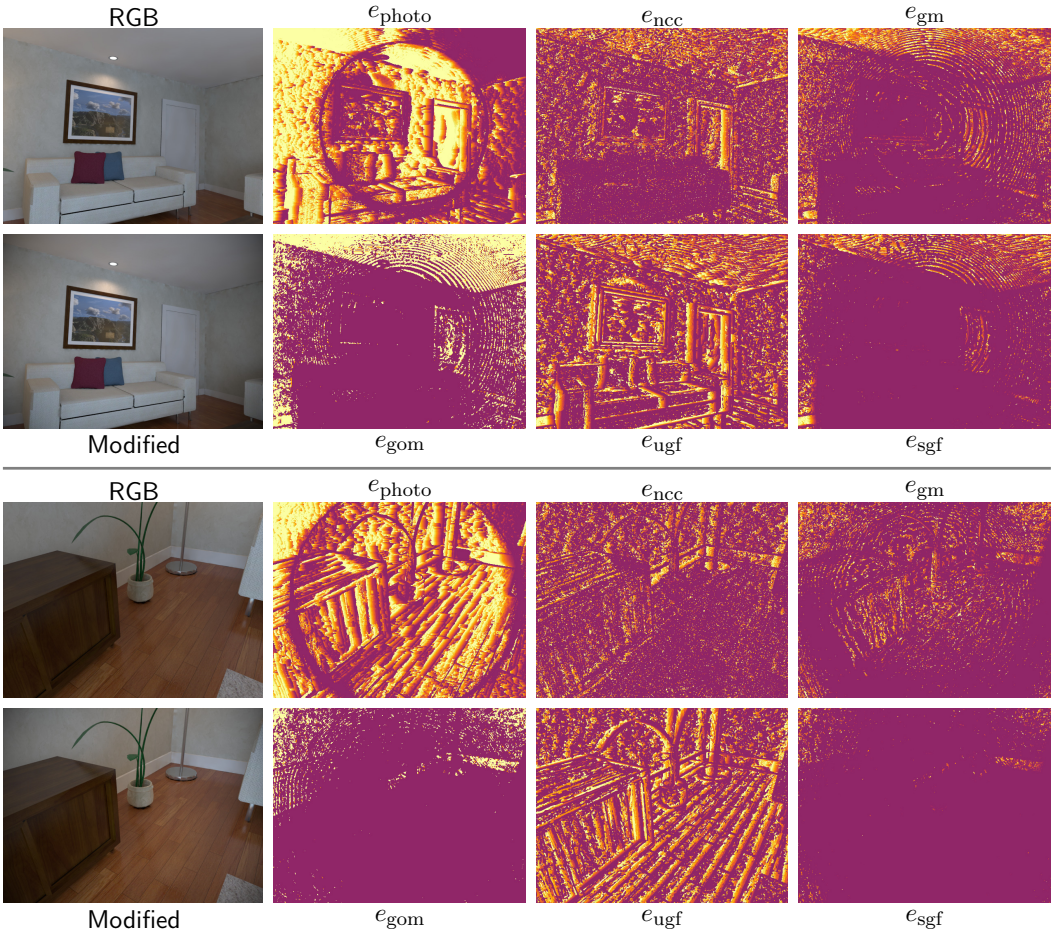

Figure 4.1: Matching cost comparison on ICL-NUIM (Handa et al., 2014): Disparity estimation against the same image with slight vignetting and different exposure time results in large disparity errors (yellow) for most cost functions. The circle in e_{photo} occurs where vignetting and exposure change cancel out.

In summary, we thoroughly evaluate our gradient-based dissimilarity to support our key claims, which are:

- First, our proposed metric is better suited for stereo disparity estimation than existing approaches.
- Second, it is also well-suited for direct image alignment.
- Third, our metric can be integrated into existing VO systems and increase their robustness while running at the frame rate of a typical camera.

Thus, the metric improves camera pose estimation and, in turn, the mapping capabilities of mobile robots. Our experimental evaluation indicates that our metric leads to more robust and more accurate estimates of the scene depth in typical disparity tasks as well as camera trajectories from direct image registration.

We believe that a series of existing VO and Visual-SLAM (VSLAM) systems can benefit from our findings reported here.

Preface

This chapter is adapted from Quenzel et al. (2020), previously published by IEEE and presented at the International Conference on Robotics and Automation (ICRA 2020).

Statement of Personal Contribution

> "The author of this thesis [Jan Quenzel] substantially contributed to all aspects of the previous publication (Quenzel et al., 2020), including the conception, literature survey, design, and implementation of the proposed methods, the preparation and conduct of experiments and evaluation of the proposed approach, conducting the experimental evaluation, the analysis and interpretation of the experimental results, drafting the manuscript, as well as the revision and final approval of the version to be published."

4.1 Related Work

There has been extensive work to improve the robustness of VO and VSLAM methods to ensure photometric consistency under changing illumination. Typically, feature-based methods are more resilient towards such variations as descriptor designs aim for discernability even under severe changes, across seasons, and for invariance of camera type. SIFT (Lowe, 2004) is the method of choice for structure-from-motion (SfM) (Schönberger and Frahm, 2016), but it often comes with a prohibitive computational cost for real-time applications. Nevertheless, PTAM (Klein and Murray, 2007) using features from accelerated segment test (FAST) keypoints (Rosten and Drummond, 2006) and ORB-SLAM (Mur-Artal et al., 2015; Mur-Artal and Tardós, 2017; Campos et al., 2021) exploit binary descriptors (Calonder et al., 2010; Rublee et al., 2011) to perform feature-based VSLAM in real-time.

With a good initial guess, direct methods can obtain more accurate estimates of the camera trajectory (Engel et al., 2018) than feature-based approaches, as they exploit more intensity measurements of the images. Hence, BundleFusion (Dai et al., 2017) initializes and constrains dense alignment using features. A popular approach, e.g., used by Schneider et al. (2012), initially extracts GoodFeaturesToTrack (Shi and Tomasi, 1994) based on the Shi-Tomasi-Score and continues alignment with the optical flow Kanade–Lucas–Tomasi tracker (KLT) tracker (Tomasi and Kanade, 1992) which operates directly on the intensity values of extracted patches. Similarly, Basalt (Usenko et al., 2020) estimates the optical flow for patches at FAST keypoints by minimization of the locally scaled intensity differences.

A further popular method for motion estimation from camera images is LSD-SLAM (Engel et al., 2014). For robustness, the authors use the Huber norm (Huber, 1964) during motion estimation and map creation while minimizing a variance-weighted photometric error. LSD-SLAM creates in parallel the map for tracking by searching along the epipolar lines, minimizing the sum of squared differences (SSD). For the stereo version, Engel et al. (2015) alternate between estimating a global affine function to model changing brightness and optimizing the

relative pose during alignment. As an alternative, Kerl et al. (2013b) propose to weight the photometric residuals with a t-distribution that better matches the RGB and depth (RGB-D) sensor characteristics. Engel et al. (2018) furthermore proposed with direct sparse odometry (DSO) a sparse-direct approach that further incorporates photometric calibration if available or estimates affine brightness changes. They maintain an information filter to jointly estimate all involved variables.

Pascoe et al. (2017) proposed to use the normalized information distance (NID) metric for direct monocular VSLAM. NID works well even for tracking across seasons and under diverse illumination. However, the authors reportedly prefer the photometric depth estimation for a stable initialization and only use NID after revisiting. Furthermore, Park et al. (2017) presented an evaluation of different direct alignment metrics for VSLAM. They favored the gradient magnitude due to its accuracy, robustness, and speed, while the census transform provided more accurate results at a much larger computational cost. Common in stereo matching are the absolute gradient difference combined with the photometric error, e.g., in StereoPatchMatch (Bleyer et al., 2011), and the census transform, e.g., in MeshStereo (Zhang et al., 2015).

In our work, we improve the gradient orientation-based metric of Haber and Modersitzki (2006) by introducing a magnitude-dependent scaling term to match gradient magnitude and orientation simultaneously. We apply this to solve direct image alignment for VO as well as semi-dense disparity, and depth estimation. We integrated our metric in two stereo-matching algorithms as well as two VO systems. Hence, we evaluate and compare the metric against existing approaches on two stereo estimation and VO datasets.

After the original publication (Quenzel et al., 2020), the introduction of vision transformer (ViT) (Dosovitskiy et al., 2021) and large-scale datasets (Li and Snavely, 2018; Wang et al., 2019; Reizenstein et al., 2021; Arnold et al., 2022) led to more accurate learned models for monocular as well as MVS depth estimation. In general, MVS methods (Wang et al., 2024a) replaced classic dissimilarity metrics with learned feature extraction like the feature pyramid network (FPN) (Lin et al., 2017) in CasMVS (Gu et al., 2020), the pretrained DINOv2 (Oquab et al., 2024) in MVS-Former++ (Cao et al., 2024) or the CroCo-v2 architecture (Weinzaepfel et al., 2023) in DUST3R (Wang et al., 2024c). Newer methods like the Large Reconstruction Model (LRM) (Wei et al., 2024; Xie et al., 2024; Zhang et al., 2024a) and DUST3R/MASt3R (Leroy et al., 2024; Wang et al., 2024c) remove the explicit cost volume computation. In contrast, the training of recent monocular methods still uses primarily supervised disparity (Ranftl et al., 2022) or log-depth (Bhat et al., 2023) losses, e.g., with MAE/MAD after scale and shift alignment. However, additional image gradient matching losses in MiDaS (Ranftl et al., 2022) lead to improved depth prediction. The authors of DepthAnything v2 (Yang et al., 2024) and Depth Pro (Bochkovskiy et al., 2025) further emphasize the importance of gradient matching losses to learn fine-grained details from synthetic data.

In radiance field-based reconstruction, photometric rendering losses are the standard in neural radiance field (NeRF) (Mildenhall et al., 2021) and Gaussian splatting (GS) (Kerbl et al., 2023; Huang et al., 2024). NeRF methods usually employ random-sampled rays as the high computational cost of the volumetric rendering is a limiting

factor. The number of rays used per training iteration[1] is, thus, small relative to the total number of rays required for inference of a full image. As a result, patch-based losses remain the exception (Xie et al., 2023) in NeRF methods, even though Wu et al. (2025) recently demonstrated improved depth and view consistency.

In contrast, GS commonly includes the patch-based (D-)SSIM loss as the full image rendering works at high frame rates, even on commodity graphic cards. However, the structured similarity index measure (SSIM) tends to underestimate blur (Liu et al., 2012; Xue et al., 2014), which leads to fainter edges in images. Here, image gradient losses can potentially improve the quality of GS, similar to the aforementioned works on monocular depth estimation.

As discussed in the previous Sec. 3.1, the photometric error remains important in visual-inertial odometry (VIO) for KLT tracking (Geneva et al., 2020; Usenko et al., 2020; Luo et al., 2022; Huai and Huang, 2024; Lin and Zhang, 2024) or direct image alignment (Miao et al., 2022; Dexheimer and Davison, 2024), especially in real-time scenarios and on constrained compute platforms. Moreover, the tracking within modern RGB-D-simultaneous localization and mapping (SLAM) (Zhu et al., 2022b; Liso et al., 2024; Yan et al., 2024) involves combinations of geometric and photometric losses for rendered depth and intensity.

However, our image gradient losses remain applicable for the benefit of the respective vision systems.

4.2 Our Method

Our approach provides a new metric for pixelwise matching and is easy to integrate into existing visual state estimation systems. The metric measures the image gradient's orientation while also taking its magnitude into consideration. A pixel has image coordinates $\boldsymbol{u} = (u_x, u_y)^\intercal$ in the image domain $\Omega \subset \mathbb{R}^2$. We aim to find for a pixel $\boldsymbol{u}_i$ in image i the corresponding pixel $\boldsymbol{u}_j$ in image j that minimizes a dissimilarity measurement $e\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right)$.

4.2.1 *Photometric Dissimilarity Measures*

At first, we detail common metrics (Park et al., 2017). In direct image alignment, image i is the current frame and j a previous (key-) frame, while they correspond to the left (i) and right image (j) in stereo matching.

The simple error function e_{photo} assumes photometric consistency of the image intensity:

$$e_{\text{photo}}\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right) = I_i\left(\boldsymbol{u}_i\right) - I_j\left(\boldsymbol{u}_j\right). \tag{4.1}$$

Instead, more robust versions rely on the intensity gradient ∇I:

$$e_{\text{gm}}\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right) = \left\|\nabla I_i\left(\boldsymbol{u}_i\right)\right\| - \left\|\nabla I_j\left(\boldsymbol{u}_j\right)\right\|, \tag{4.2}$$

$$\boldsymbol{e}_{\text{gn}}\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right) = \nabla I_i\left(\boldsymbol{u}_i\right) - \nabla I_j\left(\boldsymbol{u}_j\right). \tag{4.3}$$

1 Mildenhall et al. (2021) use only 4096 random sampled rays, whereas Instant-NGP (Müller et al., 2022) processes up to 105 k.

While e_{gm} prefers pixels with similar gradient magnitude GM, it disregards the gradient direction. Instead, the gradient difference $\boldsymbol{e}_{\text{gn}}$ incorporates both magnitude and orientation. In practice, many stereo algorithms like the popular PatchMatch (Bleyer et al., 2011) combine the photometric error with the ℓ_1-norm of the gradients difference:

$$e_{\text{pm}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = (1-\alpha)|e_{\text{photo}}(\boldsymbol{u}_i, \boldsymbol{u}_j)| + \alpha \left\|e_{\text{gn}}(\boldsymbol{u}_i, \boldsymbol{u}_j)\right\|_{\ell_1}. \tag{4.4}$$

Another common approach (Scharstein and Szeliski, 2002; Hirschmüller, 2011) is cross-correlation over a fixed-size window $W \in \Omega$ centered at pixel $\boldsymbol{u}$. A notable example is normalized cross-correlation (NCC) (Schönberger et al., 2016; Park et al., 2017):

$$e_{\text{ncc}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = \frac{\sum_{\boldsymbol{a}\in W_i, \boldsymbol{b}\in W_j} (I_i(\boldsymbol{a}) - \overline{I}_{W_i})(I_j(\boldsymbol{b}) - \overline{I}_{W_j})}{\sqrt{\sum_{\boldsymbol{a}\in W_i} (I_i(\boldsymbol{a}) - \overline{I}_{W_i})} \sqrt{\sum_{\boldsymbol{b}\in W_j} (I_j(\boldsymbol{b}) - \overline{I}_{W_j})}}, \tag{4.5}$$

with $\overline{I}_W$ being the mean intensity of window W.

4.2.2 *Normalized Gradient-based Dissimilarity Measure*

Aligning the gradient's orientation is a complementary approach. A naïve method may use the costly atan-operation to compute the orientation angle θ and simply calculate differences. Instead, the dot product provides a more efficient solution due to its relation to the cosine as a measure of orientation (Haber and Modersitzki, 2006; Taylor et al., 2015). The cosine of the angle $\cos\theta$ between two unit vectors $\boldsymbol{a}, \boldsymbol{b}$ equals the dot product $\boldsymbol{a} \cdot \boldsymbol{b}$, which is zero for perpendicular vectors, one for the same and minus one for opposite orientation. Simple normalization using the gradient's magnitude adversely enhances noise in low-gradient regions such that the noise predominates the orientation. Hence, Taylor et al. (2015) employ normalization over a window W:

$$e_{\text{gom}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = 1 - \frac{\sum_{u\in W} |\nabla I_i(\boldsymbol{u}_i) \cdot \nabla I_j(\boldsymbol{u}_j)|}{\sum_{u\in W} \|\nabla I_i(\boldsymbol{u}_i)\| \, \|\nabla I_j(\boldsymbol{u}_j)\|}. \tag{4.6}$$

Instead, we regularize the magnitude by a parameter ε, as in Haber and Modersitzki (2006):

$$\varepsilon = \frac{1}{|\Omega|} \sum_{\boldsymbol{u}\in\Omega} \|\nabla I(\boldsymbol{u})\|^2, \tag{4.7}$$

$$\nabla_\varepsilon I = \frac{\nabla I}{\sqrt{\|\nabla I\|^2 + \varepsilon}}. \tag{4.8}$$

Here, ε downweighs low gradient regions towards a magnitude close to zero. We compute ε on a per-image basis and will use ε and ϑ to make the distinction between different images (e.g., i, j) more visible. Haber and Modersitzki (2006) minimize the per-pixel error $e_{\text{ngf}} \in [0, 2]$:

$$e_{\text{ngf}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = 1 - \left[\nabla_\varepsilon I_i(\boldsymbol{u}_i) \cdot \nabla_\vartheta I_j(\boldsymbol{u}_j)\right]^2. \tag{4.9}$$

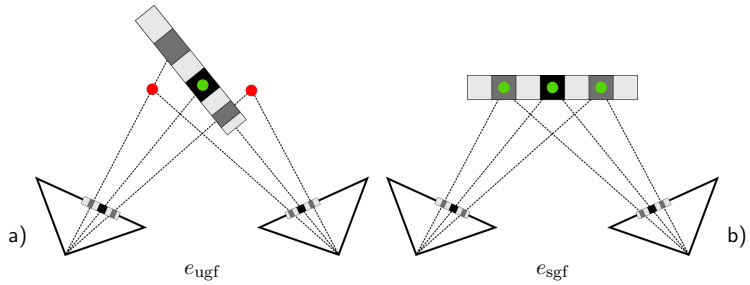

Figure 4.2: Association impact. a) e_{ngf} and e_{ugf} tend to match patches with similar gradient orientation but stronger magnitude. This can cause severe distortions in the 3D reconstruction. b) Associating patches with similar gradient orientation and magnitude using e_{sgf} allows for correct triangulation.

The squared dot product, or even its absolute value, results in gradients that coincide with the same as well as opposite orientation. This is a desirable property in multi-modal image registration, e.g., to register MRT against CT data or vice versa when image gradients have opposite directions.

However, the error e_{ngf} promotes matching low-gradient pixels with higher ones rather than similar gradients. During depth estimation, matching against the largest magnitude edge skews the search region when successively reduced or directly produces inconsistent depth estimates with high reprojection errors, as visualized in Fig. 4.2.

When we use images from the same sensor type, we omit the square which results in the uni-modal residual $e_{\text{ugf}} \in [0, 2]$:

$$e_{\text{ugf}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = 1 - \nabla_\vartheta I_j\left(\boldsymbol{u}_j\right) \cdot \nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right). \tag{4.10}$$

To ensure the correct behavior for smaller gradients, as visualized in Fig. 4.3, we scale the dot product by the maximum of both regularized gradients squared ℓ_2-norm:

$$e_{\text{sgf}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = 1 - \frac{\nabla_\vartheta I_j\left(\boldsymbol{u}_j\right) \cdot \nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right)}{\max\left(\left\|\nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right)\right\|^2, \left\|\nabla_\vartheta I_j\left(\boldsymbol{u}_j\right)\right\|^2, \tau\right)}, \tag{4.11}$$

with a small constant τ to prevent division by zero. For regularized gradients with equal magnitude ($\|\nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right)\| = \|\nabla_\vartheta I_j\left(\boldsymbol{u}_j\right)\|$), this reduces to $1 - cos(\theta)$ with unit vectors. Otherwise, the dot product in the numerator will be smaller than the larger squared magnitude in the denominator, even for codirectional gradients, which results in a higher error e_{sgf} compared to matching with a similar magnitude. Hence, the scaling term of SGF should increase the number of successfully estimated points in semi-dense depth estimation.

We derive two additional combinations of magnitude and orientation with the aim of simplifying the mathematical operations in the above equation and ideally removing the division in Eq. 4.11:

$$s_{ij}(\boldsymbol{u}_i, \boldsymbol{u}_j) = \frac{\|\nabla_\vartheta I_j\left(\boldsymbol{u}_j\right)\|}{\|\nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right)\|} \left\|\nabla I_i\left(\boldsymbol{u}_i\right)\right\|^2, \tag{4.12}$$

$$s_{ji}(\boldsymbol{u}_i, \boldsymbol{u}_j) = \frac{\|\nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right)\|}{\|\nabla_\vartheta I_j\left(\boldsymbol{u}_j\right)\|} \left\|\nabla I_j\left(\boldsymbol{u}_j\right)\right\|^2, \tag{4.13}$$

$$e_{\text{sgf2}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = \max\left(s_{ij}\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right), s_{ji}\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right)\right) - \nabla I_j\left(\boldsymbol{u}_j\right) \cdot \nabla I_i\left(\boldsymbol{u}_i\right), \tag{4.14}$$

$$e_{\text{sgf3}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = \left\|\nabla I_i\left(\boldsymbol{u}_i\right)\right\| \left\|\nabla I_j\left(\boldsymbol{u}_j\right)\right\| - \nabla I_j\left(\boldsymbol{u}_j\right) \cdot \nabla I_i\left(\boldsymbol{u}_i\right). \tag{4.15}$$

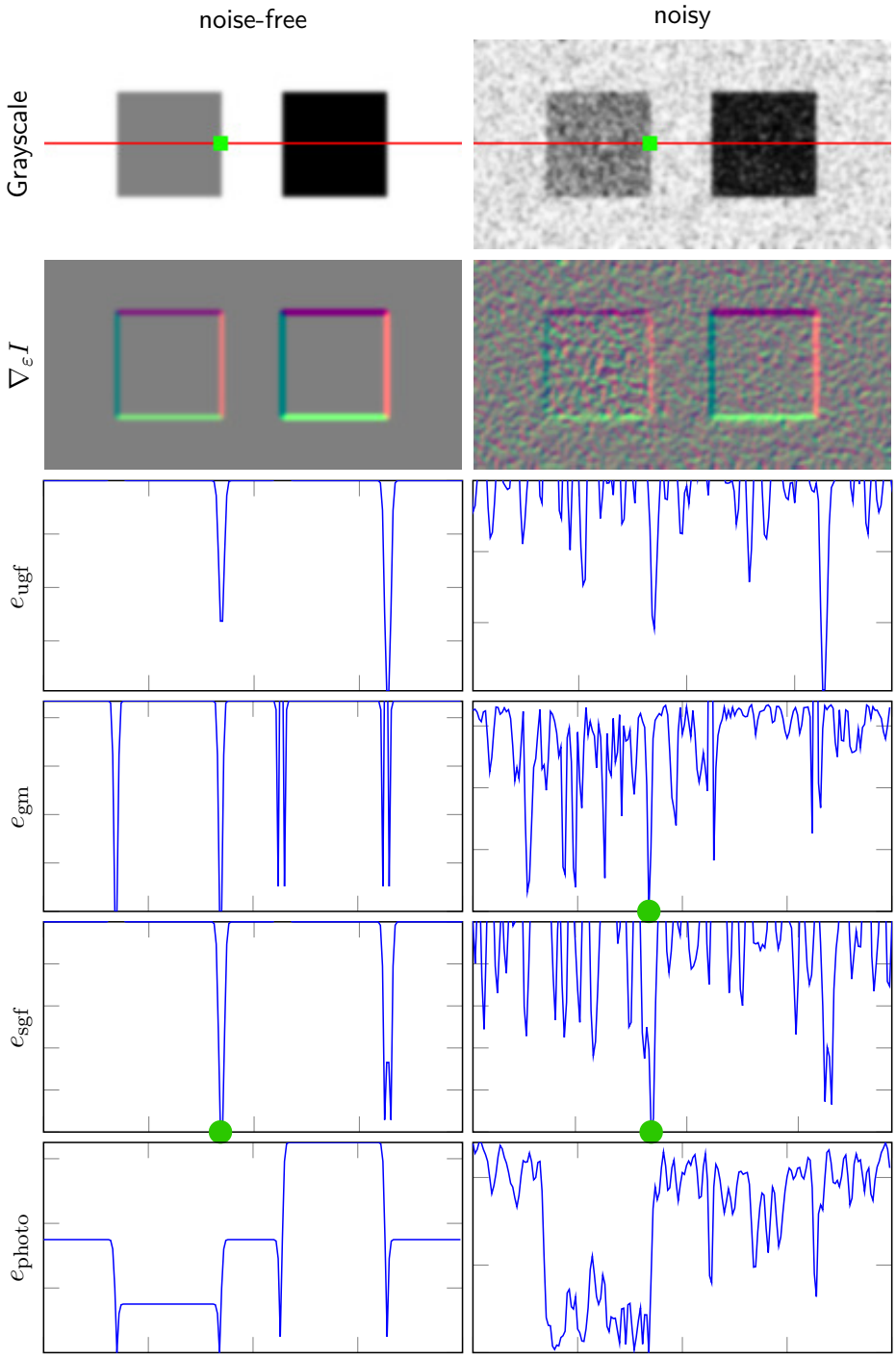

Figure 4.3: Error comparison for various metrics on a toy example. The lower boxes (3rd-6th row) show the respective error between the green reference and a shifted window along the red horizontal line. e_{ugf} prefers strong edges with same orientation, while e_{gm} does not take the orientation into account and thus generates further local minima. Our e_{sgf} provides the correct minima which are marked with a green circle.

4.2.3 *Stereo Matching and Direct Image Alignment*

Given our gradient-based dissimilarity functions, we will now formulate the stereo matching and direct image alignment problems.

Stereo matching aims to find for each pixel $\boldsymbol{u}_i$ in frame i the corresponding pixel $\boldsymbol{u}_j$ in frame j that minimizes a dissimilarity measurement $e\left(\boldsymbol{u}_i, \boldsymbol{u}_j\right)$. Then, the disparity d is the distance along the x-axis of a stereo-rectified left and right image pair (I_l, I_r):

$$\boldsymbol{u}_r\left(\boldsymbol{u}_l, d\right) = \boldsymbol{u}_l - (d, 0)^\intercal, \tag{4.16}$$

$$d^*_{\boldsymbol{u}} = \arg\min_{d \in \mathcal{R}} \sum_{\boldsymbol{u}_l \in W} e\left(\boldsymbol{u}_l, \boldsymbol{u}_r\left(\boldsymbol{u}_l, d\right)\right). \tag{4.17}$$

Instead of a single pixel, stereo matching uses a patch $W_{\boldsymbol{u}}$ centered around the pixel $\boldsymbol{u}$ with window size w to improve robustness.

Direct image alignment seeks the transformation T_{cr} that registers the reference with the current image optimally w.r.t. an error metric e between a reference pixel-patch $\mathcal{N}_{\boldsymbol{p}_r}$ around $\boldsymbol{p}_r$ and its projection onto I_c:

$$T_{cr} = \arg\min \sum_{\boldsymbol{p}_r \in \mathcal{M}} \sum_{\boldsymbol{p}_k \in \mathcal{N}_{\boldsymbol{p}_r}} \rho\left(\left\|e\left(\boldsymbol{p}_i\right)\right\|^2\right). \tag{4.18}$$

A robust cost function ρ, like the Huber norm (Huber, 1964), reduces the impact of outliers on the solution. Standard Gauss-Newton (GN) (see Sec. 2.2) solves Eq. 4.18 iteratively and requires the Jacobians for e w.r.t. the projected pixel $\boldsymbol{u}_c$ in the current image I_c.

With the Hessian $\left(\nabla_2\right) I_i$ of the intensity at pixel $\boldsymbol{u}_i$, the Jacobians for e_{sgf} is:

$$s_1 = \nabla_\vartheta I_j\left(\boldsymbol{u}_j\right) \cdot \nabla_\varepsilon I_i\left(\boldsymbol{u}_i\right) \begin{cases} -1, & \text{if } \left\|\nabla_\vartheta I_j\right\|^2 > \left\|\nabla_\varepsilon I_i\right\|^2 \\ 1 - \frac{2}{\left\|\nabla_\varepsilon I_i\right\|}, & \text{otherwise} \end{cases} \tag{4.19}$$

$$\frac{\partial e_{\text{sgf}}}{\partial \boldsymbol{u}_i} = -\frac{\left(\nabla_\vartheta I_j + s_1 \nabla_\varepsilon I_i\right)^\intercal}{\max\left(\left\|\nabla_\varepsilon I_i\right\|, \left\|\nabla_\vartheta I_j\right\|\right)} \frac{\left(\nabla_2\right) I_i}{\left\|\nabla I_i\right\|_\varepsilon}. \tag{4.20}$$

For e_{sgf2}, e_{sgf3}, the Jacobians are:

$$s_2 = \begin{cases} \frac{\left\|\nabla_\vartheta I_j\right\|}{\left\|\nabla_\varepsilon I_i\right\|}\left(2 - \frac{\left\|\nabla I_i\right\|^2}{\left\|\nabla I_i\right\|^2 + \varepsilon}\right), & \text{if } s_{ij} > s_{ji} \\ \frac{\left\|\nabla_\varepsilon I_i\right\|}{\left\|\nabla_\vartheta I_j\right\|} \frac{\left\|\nabla I_j\right\|^2}{\left(\left\|\nabla I_i\right\|^2 + \varepsilon\right)}, & \text{otherwise} \end{cases}, \tag{4.21}$$

$$\frac{\partial e_{\text{sgf2}}}{\partial \boldsymbol{u}_i} = \left(s_2 \nabla I_i - \nabla I_j\right)\left(\nabla_2\right) I_i, \tag{4.22}$$

$$\frac{\partial e_{\text{sgf3}}}{\partial \boldsymbol{u}_i} = \left(\frac{1}{2}\frac{\left\|\nabla I_j\right\|}{\left\|\nabla I_i\right\|} \nabla I_i - \nabla I_j\right)\left(\nabla_2\right) I_i. \tag{4.23}$$

4.3 Evaluation

The design of our first experiment illustrates the influence of small image variations on our robust metric in Fig. 4.1. We used images from sequence "lr kt2" of the ICL-NUIM (Handa et al., 2014) dataset with adapted exposure time and added vignetting

Table 4.1: Evaluation on Middlebury Stereo 2014 training set (Scharstein et al., 2014). Lower values are better (↓) with second and **best** highlighted.

		Orig.	e_{sad}	e_{agm}	e_{pm}	e_{sgf}
StereoBM	mean	7.20	5.80	6.31	4.56	**3.29**
	bad 1	18.36	20.51	21.33	17.19	**12.60**
	bad 2	16.41	17.01	17.79	14.25	**10.36**
	bad 4	14.88	14.19	14.69	11.94	**8.61**
	invalid	40.44	**34.51**	52.69	44.74	45.49
MeshStereo	mean	5.68	11.22	7.85	6.70	**4.17**
	bad 1	**16.87**	46.55	33.45	28.51	20.61
	bad 2	**13.02**	40.25	27.38	23.32	15.94
	bad 4	**10.71**	33.18	22.02	18.78	12.53
	invalid	**0.01**	1.01	0.09	0.08	0.04

to frames 120 and 808 to underline how minimal changes impact the dissimilarity metrics even for the same image. We evaluate the different metrics between the original and the modified image for disparities $d \in [0, 20)$ with a window size of 3. The purple regions have minimal disparity error, with $d = 0$ being the correct solution. As expected, the photometric error e_{photo} is largest (avg. 8.13 px / 7.76 px), while gradient orientation alone (e_{ugf}) achieves on avg. 4.49 px / 4.78 px. Normalized cross-correlation (e_{ncc}) results in a disparity error of 3.04 px / 2.38 px. The magnitude e_{mag} (2.11 px / 1.40 px) and GOM e_{gom} (2.02 px / 0.49 px) perform reasonably well. The PatchMatch error e_{pm} (1.24 px / 0.18 px) obtains the second-best result after our metric e_{sgf} (1.21 px / 0.18 px), which exhibits the smallest dissimilarity values.

4.3.1 *Stereo Matching*

The second experiment showcases our metric's suitability for (semi-) dense depth estimation, supporting the first claim. For this, we integrated a variety of metrics into the cost volume calculation of OpenCV's StereoBM[2] as well as the more sophisticated MeshStereo algorithm (Zhang et al., 2015). The original implementation of MeshStereo computes the cost volume with the Census-Transform (Park et al., 2017), while OpenCV uses the absolute difference (ℓ_1-norm):

$$e_{\text{sad}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = |e_{\text{photo}}(\boldsymbol{u}_i, \boldsymbol{u}_j)|. \tag{4.24}$$

Additionally, we compare these baselines against the PatchMatch dissimilarity e_{pm}, our metric e_{sgf}, and the absolute difference in gradient magnitude:

$$e_{\text{agm}}(\boldsymbol{u}_i, \boldsymbol{u}_j) = |e_{\text{gm}}(\boldsymbol{u}_i, \boldsymbol{u}_j)|. \tag{4.25}$$

2 https://docs.opencv.org/4.9.0/d9/dba/classcv_1_1StereoBM.html, based on Konolige (2010)

Table 4.2: Evaluation on KITTI Stereo 2015 training set (Menze et al., 2015). Lower values are better (↓) with second and **best** highlighted.

		Orig.	e_{sad}	e_{agm}	e_{pm}	e_{sgf}
StereoBM	mean	6.11	3.21	3.17	1.74	**1.61**
	bad 1	19.80	19.79	22.13	15.93	**13.99**
	bad 2	11.60	10.07	11.04	6.87	**5.91**
	bad 4	9.03	6.34	6.73	3.94	**3.41**
	invalid	46.74	**29.57**	53.02	39.33	45.17
MeshStereo	mean	2.03	2.94	2.92	2.07	**2.02**
	bad 1	**27.95**	42.34	33.84	29.60	29.35
	bad 2	**12.00**	25.45	17.32	13.67	13.48
	bad 4	**5.57**	14.01	8.85	6.77	6.67
	invalid	0.07	0.15	0.10	0.08	**0.06**

Our implementation uses central finite differences to obtain the image gradient ∇I at pixel $\boldsymbol{u}$:

$$\nabla I(\boldsymbol{u}) = \frac{1}{2}\begin{bmatrix} I(u_x+1, u_y) - I(u_x-1, u_y) \\ I(u_x, u_y+1) - I(u_x, u_y-1) \end{bmatrix}. \tag{4.26}$$

All metrics in OpenCV are without prefiltering, except for e_{sad} where a different prefilter provided better results. For comparison, we report the percentage of "bad" pixels with disparity error larger than 1, 2, and 4 px as well as the mean disparity error on the training sets of the Middlebury Stereo Benchmark (Scharstein et al., 2014) (half-size) in Tab. 4.1 and the KITTI Stereo Benchmark (Menze et al., 2015) in Tab. 4.2. Our metric provides in all cases the best mean disparity error, while the results for $\boldsymbol{e}_{\text{gn}}$ are nearly indistinguishable from e_{pm} and thus omitted for brevity.

Figure 4.4 shows exemplary results for the original stereo methods in comparison with the two best-performing dissimilarities, e_{pm} and e_{sgf}, on the Teddy sequence of the Middlebury Stereo Benchmark. Although our metric's disparity is a slightly sparser, the disparity exhibits less incorrect matches in the background. Moreover, the right side of the Teddy's contour is more accurately reconstructed with the modified MeshStereo, whereas the other metrics show incorrect infill. Figure 4.5 shows exemplary results for the original stereo methods in comparison with the two best-performing dissimilarities, e_{pm} and e_{sgf}, on the KITTI Stereo Benchmark. Please note for e_{sgf}, although MeshStereo does not recover the bicyclist well, it is clearly visible with StereoBM. The original StereoBM exhibits many incorrect too-close (red) pixels in more distant areas, while e_{pm} shows thicker outlines in both algorithms.

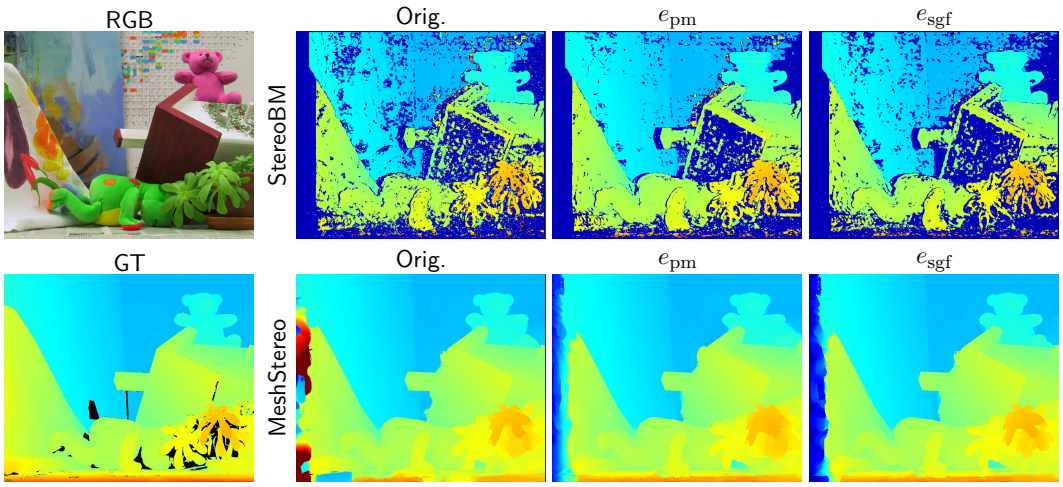

Figure 4.4: Disparity comparison on "Teddy" of the Middlebury Stereo 2014 Benchmark (Scharstein et al., 2014) for the original OpenCV Stereo Block matching (left) and MeshStereo (Zhang et al., 2015; right) and with the second best (e_{pm}) and best metric (e_{sgf}).

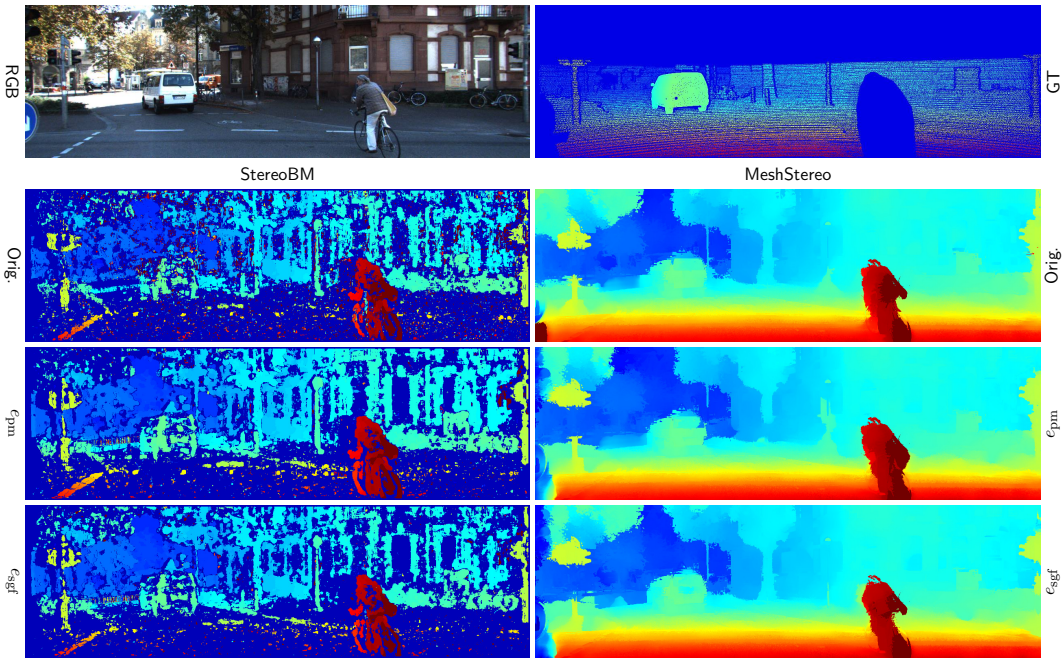

Figure 4.5: Disparity comparison on image pair 2 of the KITTI Stereo 2015 Benchmark (Menze et al., 2015) for the original OpenCV stereo block matching (left) and MeshStereo (Zhang et al., 2015; right) and with the second best (e_{pm}) and best metric (e_{sgf}).

Table 4.3: RMS-ATE [m] on EuRoC dataset (Burri et al., 2016). Lower values are better (↓) with second and **best** highlighted.

		MH1	MH2	MH3	MH4	MH5	V11	V12	V13	V21	V22	Avg.
Original	OKVIS	0.085	0.083	0.135	0.143	0.278	0.041	0.956	0.102	0.054	0.063	0.194
	ORB-SLAM2	0.124	0.094	0.253	0.151	0.132	0.090	0.219	0.270	0.149	0.203	0.168
	SVO2	0.093	0.111	0.355	2.444	0.456	0.074	0.174	0.270	0.109	0.158	0.424
	DSO	**0.051**	0.045	0.165	0.164	0.460	0.194	0.151	1.075	0.080	0.098	0.227
	Basalt	0.076	0.045	0.058	0.096	0.141	0.041	0.052	0.073	0.032	**0.046**	0.066
Ours	DSO w/ e_{sgf}	0.071	0.050	0.264	0.235	0.237	0.142	0.178	0.933	0.072	0.086	0.206
	Basalt w/ e_{gm}	0.090	0.044	0.084	**0.091**	0.135	0.049	0.099	0.161	0.030	0.079	0.086
	Basalt w/ $\boldsymbol{e}_{\text{gn}}$	0.076	0.055	**0.057**	0.112	0.115	**0.039**	**0.042**	0.093	0.037	0.048	0.067
	Basalt w/ e_{sgf}	0.078	0.062	0.080	0.215	0.111	0.043	0.107	0.156	0.037	0.108	0.100
	Basalt w/ e_{sgf2}	0.086	0.065	0.081	0.109	0.148	0.040	0.069	**0.061**	**0.029**	0.058	0.075
	Basalt w/ e_{sgf3}	0.061	**0.042**	0.065	0.094	**0.106**	0.041	0.056	0.082	0.034	0.054	**0.063**

4.3.2 *Direct Image Alignment*

To support our second and third claim, we provide comparisons on the EuRoC dataset (Burri et al., 2016) for a set of state-of-the-art VO and VIO approaches including Basalt (Usenko et al., 2020), DSO (Engel et al., 2018), OKVIS (Leutenegger et al., 2015), ORB-SLAM2 (Mur-Artal and Tardós, 2017) and SVO2 (Forster et al., 2017).

For a fair comparison, the evaluation uses, if provided, the tailored parameters for the EuRoC dataset. Similarly, Basalt runs purely in VIO mode and without ORB-SLAM2 without global bundle adjustment.

We integrated the different metrics in the optical flow frontend of Basalt. We carried out a two-fold cross-validation with hyperopt (Bergstra et al., 2013) to obtain suitable parameters for each metric.

Instead of finite differences as before, we use the "3x3-int" Scharr-Operator (see Tab. B.11 in Scharr (2004)) on the rotated patches for the intensity gradient:

$$\boldsymbol{s}_f = [1, 0, -1], \tag{4.27}$$

$$\boldsymbol{s}_g = [47, 162, 47], \tag{4.28}$$

$$\nabla I_x = \frac{1}{512} \boldsymbol{s}_g^\intercal * \boldsymbol{s}_f * I, \tag{4.29}$$

$$\nabla I_y = \frac{1}{512} \boldsymbol{s}_f^\intercal * \boldsymbol{s}_g * I, \tag{4.30}$$

for its improved rotation symmetry as we observed degraded precision using finite differences. Additionally, we modified the depth estimation of DSO and replaced the original patch similarity metric based on Brightness-Constancy-Assumption e_{photo} with our e_{sgf} term.

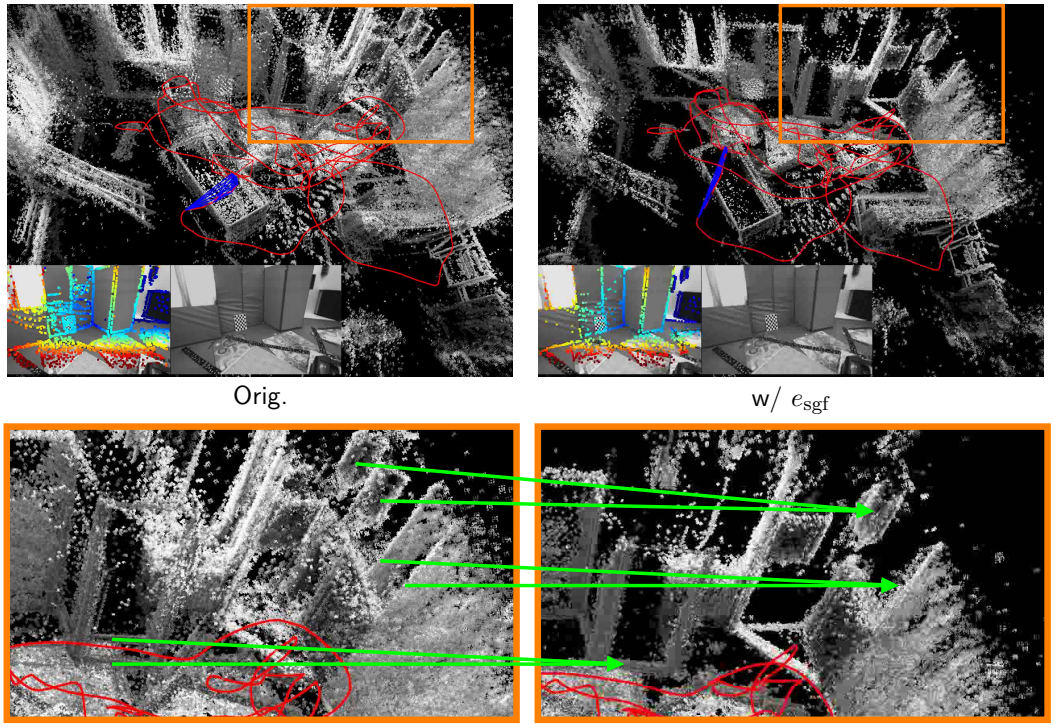

Figure 4.6: Resulting map and trajectory (red line) of DSO (Engel et al., 2018) using photometric consistency (left) and with e_{sgf} (right) for depth estimation on V1_01 of the EuRoC dataset (Burri et al., 2016). The reduced drift is clearly visible in the sharper edges and a reduction of double walls as highlighted in the cutout region (orange).

For all methods, we report the average root-mean-squared (RMS) ATE (Eq. 2.90) over all aligned trajectories (Zhang and Scaramuzza, 2018) in Fig. 4.3. The alignment uses a rigid SE(3) transform for stereo systems and a similarity transform for DSO. In order to account for randomness and runtime effects, we average the results per sequence over multiple runs.

The comparison in Fig. 4.6 shows a sparser, more consistent map with reduced drift and no double walls for our e_{sgf} on sequence V1_01 of the EuRoC dataset. This is also evident in the ATE since our modified DSO achieves a lower average ATE than the original DSO. Furthermore, we observed reduced variance in ATE and an increase in successful tracking attempts by 10 % on V1_02 and V1_03, which exhibit strong lighting changes.

Overall Basalt performs very well for all tested metrics. Presumably, due to the more complex and more difficult to-optimize Jacobian, e_{sgf} performs worse than our other derived metrics. Here, the simplifications of e_{sgf2} and e_{sgf3} payoff and e_{sgf3} achieves the best result.

4.4 Summary

In this chapter, we proposed a new gradient-based dissimilarity metric for direct image alignment, called SGF, and verified our key claims.

Our measure introduces a scaling term that promotes matching of similar gradient magnitude into the gradient orientation metric of Haber and Modersitzki (2006). As a result, SGF improves the robustness of image alignment and is beneficial for stereo matching and VO computation alike. Additionally, we propose some simplified variations and provide the respective Jacobians required for optimization with our new metrics.

We applied and evaluated our approach in a multitude of settings as our metric is easy to integrate into existing visual systems. In disparity estimation, the proposed approach outperforms the existing metrics integrated in OpenCV's StereoBM and MeshStereo on the Middlebury and KITTI Stereo benchmarks. Moreover, DSO achieves a significant reduction in drift after integrating SGF into DSO's depth estimation. Even in image alignment, our approach performs well, as demonstrated with Basalt on the EuRoC dataset. All the while, the modified methods maintain their real-time performance running at typical camera frame rates and thus can make a positive impact on various VO, SLAM, or similar state estimation approaches running on systems with limited computational power.

5

LiDAR Odometry

Light Detection and Ranging (LiDAR) plays a major role in environment perception and mapping for unmanned aerial vehicles (UAVs) (Beul et al., 2018), unmanned ground vehicles (UGVs) (Shan and Englot, 2018), and autonomous driving (Geiger et al., 2012). Many autonomy or assistance functionalities rely on simultaneous localization and mapping (SLAM) and odometry systems to operate out-of-sight in global navigation satellite system (GNSS)-denied environments or to improve the situational awareness while reducing cognitive strain on operators. Despite much progress, robustness and reliability remain difficult in crowded, dynamic scenes and close to structures. These factors become crucial when a risk-minimizing state, like stopping, is difficult to maintain, e.g., for UAVs. Furthermore, new LiDAR sensors immensely increased the density and amount of measurements within recent years, posing new challenges for processing large point clouds in real-time.

Most odometry and SLAM systems do not fully exploit dependencies between consecutive LiDAR scans when registering against a previous scan or local map but merely initialize with prior motion estimates. This may result in unrealistic jumps in the trajectory, while the actual sensor motion imposes a dependency between consecutive scans. We address this limitation using a continuous-time Lie-Group B-spline trajectory representation (Sommer et al., 2020). The main contribution is our novel real-time LiDAR odometry (LO) (Quenzel and Behnke, 2021) which directly optimizes the spline knots by jointly aligning multiple local multi-resolution surfel maps using a Gaussian mixture model (GMM) formulation (Droeschel et al., 2017), as visualized in Fig. 5.1. Sparse permutohedral lattices and voxel grids ensure fast storage and access to map surfels while an adaptive selection scheme chooses the most efficient surfel resolution to speed up registration. We improve numerical stability by modifying the GMM formulation of Droeschel et al. (2017) and introduce a normal-distance-based weighting.
In summary, our key claims for the proposed LO system, called MARS, are:

- First, MARS provides reliable pose estimates with state-of-the-art quality on a variety of datasets.
- Second, our GMM is numerically more stable and more suitable for typical LiDAR sensor geometry.
- Third, adaptive selection of the appropriate surfel resolution improves registration runtime without compromising accuracy.
- Fourth, our system runs in real-time onboard a UAV, enabling safe operation in GNSS-denied environments.

The experimental evaluation highlights the performance of our approach on multiple datasets and during real-robot experiments. MARS is open-source and available at: `https://github.com/AIS-Bonn/lidar_mars_registration`.

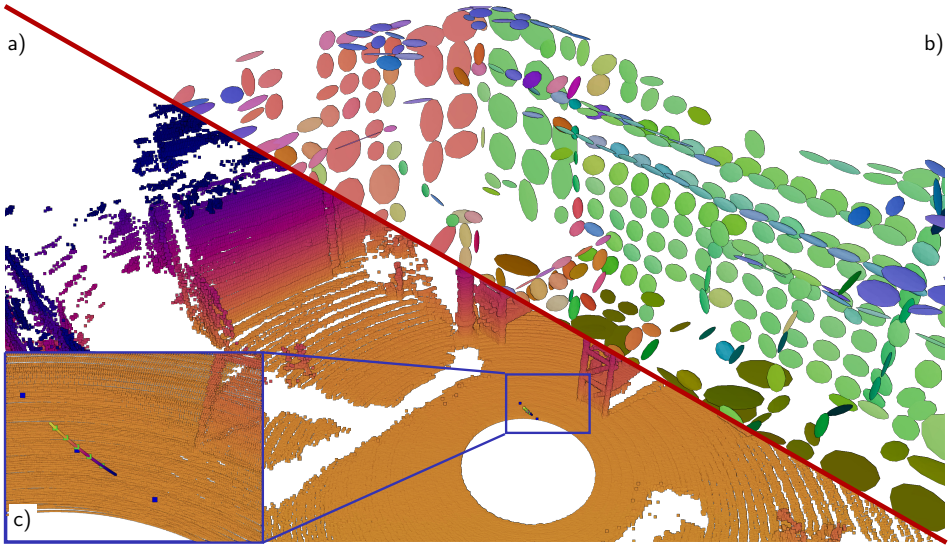

Figure 5.1: Multi-resolution surfel maps with adapted resolution represent the point clouds [a)] during registration with a continuous-time trajectory B-spline. b) The surfel color depends on their normal, while point color depends on height. c) The control points (blue dots) interpolate the scan poses (green dots) and form the spline (blue to yellow).

Preface

This chapter is adapted from Quenzel and Behnke (2021), previously published by IEEE and presented at the International Conference on Intelligent Robots and Systems (IROS 2021).

Statement of Personal Contribution

> "The author of this thesis [Jan Quenzel] substantially contributed to all aspects of the previous publication (Quenzel and Behnke, 2021), including the conception, literature survey, design, and implementation of the proposed methods, the preparation and conduct of experiments and evaluation of the proposed approach, conducting the experimental evaluation, the analysis and interpretation of the experimental results, drafting the manuscript, as well as the revision and final approval of the version to be published."

5.1 Related Work

Point cloud registration is a well-researched topic and has wide applicability (Holz et al., 2015). A basic registration method (Besl and McKay, 1992) is the iterative closest point (ICP) algorithm. ICP aligns scan and model point clouds in an iterative two-step process. In the first step, the algorithm establishes correspondences between the two point clouds. The second step calculates a transformation to reduce the distance between all corresponding points and repeats both steps with the transformed scan until ICP reaches some termination criteria. The original formulation assumes

perfect correspondences, thus suffering in reality when the sampling locations differ due to a moving sensor. Segal et al. (2009) rephrased ICP within a probabilistic framework [generalized ICP (GICP)] that allows incorporating information from the correspondence's covariance. Hence, points on a planar surface will be pulled together in normal direction but have more leeway along the surface.

Another popular scan registration approach is Normal distribution transform (NDT) (Magnusson et al., 2007; Kung et al., 2021; Renaut et al., 2023). Here, normal distributions within a regular grid represent the model point cloud. This reduces memory consumption and computation time for nearest-neighbor searches. NDT aims to maximize the likelihood of all scan points to observe the underlying surface element (surfel) described by the normal distributions.

Once there is a method to register (feature-)points, continuously estimating the sensor pose w.r.t. an updating local map becomes possible. The localization and mapping (LOAM) paradigm by Zhang and Singh (2014) extracts feature points on planar surfaces and edges from the current scan based on the local curvature. Matching features against the previous scan allows estimation of the relative motion at scan frequency. Previous pose estimates help to undistort a newly incoming scan. Every n-th scan is then further processed in the mapping thread where it is aligned against the map and integrated into it. LeGO-LOAM (Shan and Englot, 2018) adapted the general approach for horizontally placed LiDARs on UGVs under the assumption of always being able to measure the ground plane. Some filter-based approaches (Ye et al., 2019; Qin et al., 2020) use the same feature extraction to fuse LiDAR with IMU directly.

SuMa (Behley and Stachniss, 2018) performs a projection-based data association to avoid the need for costly nearest-neighbor associations. For this, it projects the current point cloud from spherical coordinates to an image and renders a model view of the surfel map using OpenGL and the currently estimated pose. The projection allows easy association between projected points and rendered surfels and enables frame-to-model alignment via ICP with the point-to-plane metric. Afterwards, the map integrates the new scan surfels with an exponentially moving average if they are more accurate than the existing surfels. A binary Bayesian filter estimates the stability and reliability per surfel such that only stable surfels are kept. A pose graph reduces overall drift upon loop closures and directly deforms the surfel map via the sensor poses.

Instead of a uniform resolution, MRSLaserMaps (Droeschel et al., 2017) represents the environment close to the sensor with greater detail. The registration performs expectation maximization (EM) of the joint log-likelihood that a scan surfel is an observation of the GMM of the local map. Circular buffers over grid cells and the fixed number of points stored within each cell enable map shifting to preserve the map's egocentric property. The shifted local map is added to a pose graph to reduce drift over time using constraints between neighboring local maps. Droeschel and Behnke (2018) extended the approach to a hierarchical pose graph where a refinement step realigns scans within a previous local map. These scans were further undistorted by a least-squares fit of a cubic Lie group B-spline to interpolate between scan poses. In practice, this method provides offline-generated maps for robot localization (Beul et al., 2018).

Elastic LiDAR Fusion (Park et al., 2018) uses only a linear continuous-time trajectory. Here, a single transformation linearly interpolates the trajectory within a time segment under an inherent constant-velocity assumption. For the rotation, the Log(·)-map (Sec. 2.4) of the Lie group $SO(3)$ lifts the relative rotation between the start and end pose to its vector-spaced Lie algebra $\mathfrak{so}(3)$, where the interpolation itself takes place. The Exp(·)-map maps back to the interpolated rotation. This simple strategy is quite efficient and fast but is limited in practice by the constant-velocity assumption. Their trajectory optimization facilitates geometric constraints penalizing deviations along the normal direction between individual surfels at different time steps within the same scan as well as towards the global map and inertial constraints for rotational velocity and acceleration from inertial measurement unit (IMU). Furthermore, the authors improve map consistency on loop closures through a deformation graph.

Although we base our work upon MRSLaserMaps (Droeschel et al., 2017), our real-time odometry system (Fig. 5.2) is a full redesign with robustness and efficiency in mind to cope with the large number of scan points generated by modern LiDAR sensors. While most odometry systems align each LiDAR scan individually against the map, we jointly register multiple scans at once in a sliding window using the continuous-time trajectory B-spline representation by Sommer et al. (2020). In contrast to MRSLaserMaps, we do not use dense but sparse voxel grids or lattices for each level within our multi-resolution surfel map. Additionally, we scale the surfels' GMM weight to balance the influence of differently sampled areas due to sensor geometry. We adaptively select the appropriate resolution for registration instead of the finest available. Furthermore, we fuse and shift maps via their surfels instead of pointwise and apply a keyframe-based sliding window for the local map such that we integrate only scans with differing view poses.

After the original publication (Quenzel and Behnke, 2021), many other works (Talbot et al., 2025) also adapted continuous-time trajectories. In CT-ICP, Dellenbach et al. (2022) represent the continuous motion within a scan using a linear B-spline. Despite this, the authors explicitly decouple the end and start pose of consecutive scans to prevent over-smoothing of the trajectory. On the contrary, Traj-LO (Zheng and Zhu, 2024b) adds smoothness constraints during sliding window optimization of the linear B-spline trajectory.

When sudden rotational changes deform a scan, a linear B-spline may be too inaccurate to represent the whole scan. Hence, the adaptive temporal subdivision of Zhou et al. (2024) decreases the time interval between knots for higher angular velocities or increases for underconstrained scan segments. Instead, Shen et al. (2025) combine a Gaussian process (GP) as their continuous-time representation with a Kalman filter (KF).

Further approaches explore uncertainty modeling, different map representations, robustness, or multiple LiDARs. VoxelMap (Yuan et al., 2022) explicitly models the (planar) surfel uncertainty w.r.t. a point's range and angle. This comes at the cost of storing all points to update the estimates within an IEKF (Xu et al., 2022). Instead, DLO (Chen et al., 2022) stores keyframes and selects which ones to fuse into the submap for GICP registration based on the convex hull over the keyframe positions.

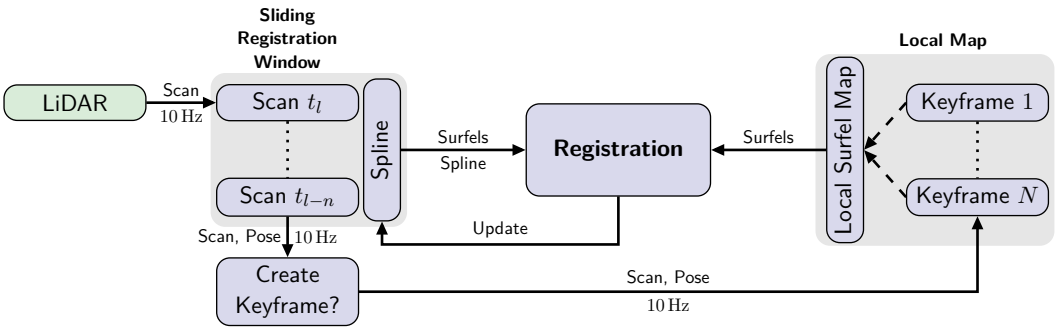

Figure 5.2: System overview: A continuous-time trajectory spline describes each scan's pose within the current sliding registration window. The registration aligns the scan surfels with a local surfel map and updates the spline. Keyframes are added if necessary to the sliding window of the local map and combined in the local surfel map.

KISS-ICP (Vizzo et al., 2023) robustifies point-to-point ICP through a motion-dependent adaptive distance threshold for correspondences and optimizes with the outlier robust Geman-McClure kernel (Barron, 2019). Building upon this, GenZ-ICP (Lee et al., 2025) adaptively balances the respective influence of point-to-point constraints and additional point-to-plane factors to improve robustness. As an alternative, X-ICP (Tuna et al., 2024) inspects the contribution of individual point-to-plane correspondences on the ICP's Hessian to filter uninformative pairs. Constrained optimization further restricts slippage due to insufficient information, e.g., in tunnels or long corridors.

More recently, learning-based LO methods utilize learned descriptor matching (Ali et al., 2023), neural implicit maps to refine keyframe-based maps (Isaacson et al., 2023) or register scans using a predicted signed distance function (SDF) from point-based neural maps (Pan et al., 2024).

5.2 Our Method

We take as input a scan P_s consisting of points $\boldsymbol{p}_i \in \mathbb{R}^3$, as well as angular velocity $\boldsymbol{\omega}_\mathrm{m}$ from an IMU or relative orientation estimates ΔR from robot odometry. In the following, we assume the scan timestamp t_s corresponds to the last acquired points within the scan. Given a spatial subdivision of $\mathbb{R}^3$, e.g., due to voxelization, each cell contains a surfel with mean $\boldsymbol{\mu}$ and covariance Σ for the embedded points. We apply a numerically stable one-pass scheme (Schubert and Gertz, 2018) for mean and covariance estimation and store the number of points M, their sum $\boldsymbol{s}_s$, and their sum of squared deviations S_s. The incremental update fuses a surfel $(M, \boldsymbol{s}, S)_a$ with another surfel $(M, \boldsymbol{s}, S)_b$ or a point $(1, \boldsymbol{p}_i, 0)_b$ according to:

$$\boldsymbol{\delta}_s = \boldsymbol{s}_a M_b - M_a \boldsymbol{s}_b, \tag{5.1}$$

$$\boldsymbol{s}_{a+b} = \boldsymbol{s}_a + \boldsymbol{s}_b, \tag{5.2}$$

$$S_{a+b} = S_a + S_b + \left(\frac{\boldsymbol{\delta}_s}{M_a M_b}\right) \cdot \left(\frac{\boldsymbol{\delta}_s}{M_a + M_b}\right)^\mathsf{T}. \tag{5.3}$$

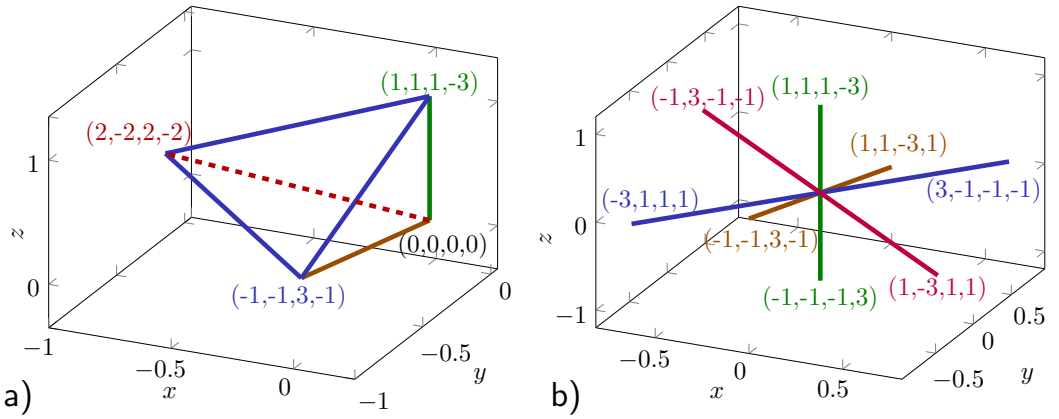

Figure 5.3: A tetrahedral [a)] covers the projection of the canonical simplex V_3 into 3D. Coordinates of neighboring lattice vertices differ by $\pm[-1\dots d\dots -1] \in \mathbb{Z}^{d+1}$ and their projected axes [b)] are not perpendicular in 3D.

Mean and covariance are directly obtained as:

$$\boldsymbol{\mu}_s = \frac{1}{M_s}\boldsymbol{s}_s, \tag{5.4}$$

$$\Sigma_s = \frac{1}{M_s - 1}S_s. \tag{5.5}$$

We regard a surfel as *valid* if it represents at least 10 points and it is a disc or ellipsoid:

$$M_s \geq 10, \tag{5.6}$$

$$\lambda_1 > 1 \times 10^{-6}, \tag{5.7}$$

with normalized Eigenvalues $\lambda_0 \leq \lambda_1 \leq \lambda_2 \in \mathbb{R}\, s.t. \sum_i \lambda_i = 1$. The surfel normal $\boldsymbol{n}$ corresponds to the Eigenvector of the smallest Eigenvalue λ_0 of Σ. The registration (Sec. 5.2.2) aligns the current sliding registration window against the local map fused from multiple keyframes' surfels, as shown in Fig. 5.2, optimizing a continuous-time B-spline trajectory.

5.2.1 *Multi-resolution Surfel Map*

Our map covers a cubic volume with side length b, e.g., twice the sensor range, with its origin in the center of the volume. We regularly subdivide the volume with fixed-size distance between adjacent corners/vertices into cells with either a 3-dimensional permutohedral lattice (Rosu et al., 2020) or a uniform 3D volume element (voxel) grid.

The lattice stems from the projection of a regular grid $(d+1)\mathbb{Z}^{d+1}$ along the one vector $\mathbf{1}$ onto the hyperplane $H_d : \boldsymbol{c} \cdot \mathbf{1} = 0$. Thus, the vertices $\boldsymbol{c}$ with integer coordinates sum up to zero, and d coordinates define the last one implicitly. Although vertices have fixed integer coordinates, scaling a point $\boldsymbol{p}$ prior to projection allows the representation of arbitrary resolutions. Per vertex, there are $2(d+1)$ neighbors where the difference between neighboring vertices is always of the form $[-1\dots d\dots -1]$.

Similarly, $d+1$ vertices span a simplex. For $d=3$, the simplex V_3 forms a tetrahedral, as shown in Fig. 5.3, with the following vertices $\boldsymbol{c}_i$:

$$V_3 = [\boldsymbol{c}_0, \boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3] = \begin{bmatrix} 0 & 1 & 2 & -1 \\ 0 & 1 & -2 & -1 \\ 0 & 1 & 2 & 3 \\ 0 & -3 & -2 & -1 \end{bmatrix}. \tag{5.8}$$

A simple rounding algorithm (Adams et al., 2010) provides the enclosing simplex for a point. Embedding points into this lattice is generally referred to as splatting. Rosu et al. (2020) let all points contribute to all vertices of the enclosing simplex using learned barycentric weights. Instead, we splat each embedded point only onto the nearest vertex for runtime efficiency.

For higher dimensions, the lattice scales advantageously compared to the regular voxel grid. More importantly for the efficiency of the GMM soft-assignments, a lattice vertex has only $2\,(d+1)$ instead of 3^d direct neighbor vertices, as shown in Fig. 5.3. LiDAR data is sparse in 3D due to free space. Hence, we use a sparse lattice or a (block-) sparse voxel grid based on small preallocated hash maps and do not allocate the entire dense volume. The sparseness and sensor geometry also imply that a single resolution of surfels may not represent the underlying scene well. Coarse surfels can blur details, while fine surfels will ignore distant measurements due to the lack of points per surfel. Hence, we introduce several map levels, e.g., 3, centered on the origin. Starting with the coarsest map at level 0, each finer level has half the cell size of the previous one to increase detail closer to the sensor. Our registration adaptively selects or uses the finest available resolution (Sec. 5.2.3). Each cell stores a double-ended queue (deque) with individual surfels from different scans and a combined surfel. In comparison to MRSLaserMaps (Droeschel et al., 2017), the deque enables faster and easier removal of old scans from the map. A new scan creates a surfel at the back of the deque and fuses all points belonging to this cell in the new surfel. Prior to fusion, subtraction of the cell center position improves numerical stability for distant points in the covariance estimation. After processing all points, the changed cells update their combined surfel since our registration requires only the combined surfels. We apply Eq. 5.1 to Eq. 5.3 to fuse one surfel $(M, \boldsymbol{s}, S)_a$ with another surfel $(M, \boldsymbol{s}, S)_b$. Figure 5.4 shows an example of a single scan as well as the local map with either a voxel grid or the lattice.

Each map stores its sensor origin $\boldsymbol{o} \in \mathbb{R}^3$, since these may change during map fusion. Additionally, the local map stores its position relative to the world frame which allows to shift the map to maintain its egocentric property. We employ a sliding registration window $\mathcal{W}_l$ in which each scan has its own surfel map $\mathcal{S}$ at the respective sensor origin.

Adding a scan to the window $\mathcal{W}_l$ at an orientation other than identity, e.g., using the current estimated orientation R, requires partitioning of the orientation R into an offset $R_{\text{off}} \neq I$ and optimized rotation $R^\star$. While that is unproblematic for a single pose or a fixed window, a changing offset within a sliding window with a continuous-time trajectory behaves similar to a jump in the trajectory. It would require a more complex or higher-order spline which takes longer to optimize. Without such an offset, the grid discretization differs from the local map but greatly simplifies and speeds up

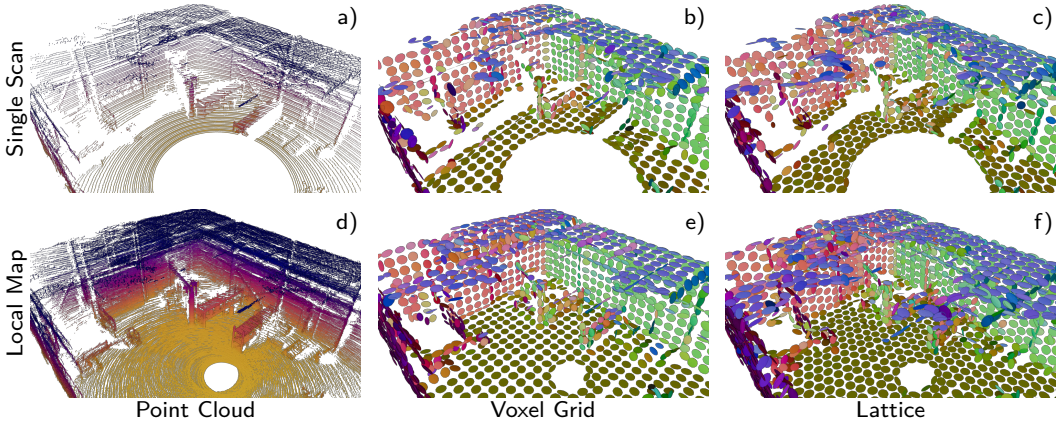

Figure 5.4: Resulting surfel maps for a single scan [a)] and a local map [d)] with voxel grid [b),e)] and permutohedral lattice [c),f)].

registration. As a grid realignment through reembedding is very costly, Stückler and Behnke (2014) trilinearly interpolate local map surfels only for the best matching model surfel in the voxel map. Interpolation on the lattice, called slicing (Adams et al., 2010), uses the barycentric coordinates at the embedded position to weigh the information stored at the simplex vertices. In our case, the soft-assignments between a scene surfel and multiple map surfels already alleviate the differences in discretization (Sec. 5.2.2). We found that slicing can improve the accuracy when using only a single matching model surfel, but not with the soft-assignments where slicing incurs a too high computational burden.

The required local map $\mathcal{M}$ consists of a fixed-size sliding keyframe window. Surfelwise fusion enables efficient addition and removal of individual keyframes to the map. After reaching a predefined distance to the previous keyframe, the oldest scan P within the sliding registration window becomes a new keyframe and pushes out the oldest keyframe if necessary. Prior to integration, we transform P into the local map frame to ensure alignment of all keyframe grids and check if the sensor position is within the same voxel/simplex as the origin on the coarsest level. If these voxels/simplices differ, shifting the whole local map maintains its egocentric property during registration. Here, we use a whole-numbered multiple of the coarsest cell size to enable efficient cell swapping instead of interpolation or cumbersome pointwise recomputation. On finer levels, this will always be a multiple of the coarse size due to our subdivision scheme, e.g., a coarse shift in the x-direction by one cell requires a shift by two finer cells on the finer level. Recomputation of the combined surfels takes place after keyframe integration.

5.2.2 *Sliding Window Continuous-time Trajectory Registration*

Our continuous-time trajectory with poses $T(t) \in SE(3)$ at time step t uses the uniform cumulative B-spline formulation of Sommer et al. (2020) on the composite manifold $SO(3) \times \mathbb{R}^3$. The manifold $SO(3) \times \mathbb{R}^3$ (Solà et al., 2018) decouples translation ($\mathbb{R}^3$) and rotation ($SO(3)$) while the resulting transform remains in the Lie group $SE(3)$. N control points $\mathcal{X}_t \in (SO(3) \times \mathbb{R}^3)^N$, also called knots, define the

uniform B-spline of order N at time t. Here, uniform refers to the timewise spacing of subsequent knots with a fixed time increment Δt and the splines' first knot at the start time t_0. For a given time t, we obtain the index i of the first active control point with non-zero weight:

$$i\,(t) = \lfloor (t - t_0)/\Delta t \rfloor, \tag{5.9}$$

in the active time segment $[t_i, t_{i+1})$ and the normalized time u:

$$u\,(t) = ((t - t_0) \bmod \Delta t)/\Delta t, \tag{5.10}$$

since the start of that segment. The normalized time u defines the cumulative knot weights $\boldsymbol{\lambda}(u)$ together with the cumulative basis matrix $\tilde{M}_N$ (Qin, 1998) of the Nth-order B-spline:

$$\boldsymbol{\lambda}(u) = \tilde{M}_N \left[u^0, \ldots, u^{N-1}\right]^\mathsf{T}. \tag{5.11}$$

With these weights, Sommer et al. define the cumulative B-spline in a Lie group $\mathcal{L}$ with knots $X_i \in \mathcal{L}$ (Def. 5.1 in Sommer et al. (2020)) using the Lie Group's exponential ($\mathrm{Exp}(\cdot)$) and logarithmic ($\mathrm{Log}(\cdot)$) maps as:

$$X\,(i, u) = X_i \cdot \prod_{j=1}^{N-1} \mathrm{Exp}\left(\lambda_j\,(u) \cdot \mathrm{Log}\left(X_{i+j}^{-1} X_{i+j-1}\right)\right). \tag{5.12}$$

The $\mathrm{Log}(\cdot)$-map (Solà et al., 2018) transfers elements of a Lie Group $\mathcal{L}$ to its tangent m-dimensional vector space $\mathfrak{l} \subset \mathbb{R}^m$, called Lie algebra. Conversely, $\mathrm{Exp}(\cdot)$ maps elements of the Lie algebra back to the Lie Group. In comparison to $SE(3)$, the composite $SO(3) \times \mathbb{R}^3$ decouples the rotation $R \in SO(3)$ and translation $\boldsymbol{p} \in \mathbb{R}^3$ such that both maps for the translation trivially become the identity function ($f(\boldsymbol{x}) = \boldsymbol{x}$) while the $\mathrm{Log}(\cdot)$- and $\mathrm{Exp}(\cdot)$-maps of $SO(3)$ (see Sec. 2.4) apply to the rotational part. This allows partitioning of Eq. 5.12 for $\mathcal{L} = SO(3) \times \mathbb{R}^3$ into:

$$\boldsymbol{p}\,(i, u) = \boldsymbol{p}_i + \sum_{j=1}^{N-1} \lambda_j\,(u) \cdot (\boldsymbol{p}_{i+j} - \boldsymbol{p}_{i+j-1}), \tag{5.13}$$

$$\boldsymbol{d}_{ij} = \mathrm{Log}\left(R_{i+j-1}^{-1} R_{i+j}\right) \in \mathbb{R}^3, \tag{5.14}$$

$$R\,(i, u) = R_i \cdot \prod_{j=1}^{N-1} \mathrm{Exp}\left(\lambda_j\,(u) \cdot \boldsymbol{d}_{ij}\right). \tag{5.15}$$

Although each control point $X_i \in \mathcal{X}$ is a tuple of $(R_i, \boldsymbol{p}i) \in SO(3) \times \mathbb{R}^3$ defined on two separate splines, the result $T_\mathcal{X}\,(t)$ is a rigid transform in $SE(3)$:

$$T_\mathcal{X}\,(t) = \begin{bmatrix} R\,(i\,(t), u\,(t)) & \boldsymbol{p}\,(i\,(t), u\,(t)) \\ \mathbf{0} & 1 \end{bmatrix}. \tag{5.16}$$

In practice, we set the order to $N = 3$ and optimize all N control points during registration — independent of the number of scans L. The sliding registration window adapts the start time t_0 and time interval Δt such that $t_0 = t_{l-n}$ is the time of the last shifted out scan and the newest scan l is within the current time interval $[t_{l-n}, t_l]$.

After shifting, we want the new spline to interpolate the previous estimates of the unshifted spline. For the newest scan, we evaluate the translational and rotational velocity and utilize the constant velocity assumption to initialize both rotation R_l and position $\boldsymbol{p}_l$. If, instead, measurements from an IMU gyroscope are available, R_l integrates the angular rate $\boldsymbol{\omega}$ of measurements between both scans with the gyroscope frequency f_ω:

$$R_l = R_{l-1} \prod \operatorname{Exp}\left(\frac{1}{f_\omega}\boldsymbol{\omega}\right). \tag{5.17}$$

Thus, we initialize the control points $\mathcal{X}$ by minimizing the following quadratic errors using the Levenberg-Marquardt (LM) algorithm (Sec. 2.2):

$$j\left(i\right) = l - n + i, \tag{5.18}$$

$$\boldsymbol{e}_{\boldsymbol{p}}\left(i\right) = \boldsymbol{p}\left(u\left(t_{j(i)}\right)\right) - \boldsymbol{p}_{j(i)-1}, \tag{5.19}$$

$$\boldsymbol{e}_R\left(i\right) = \operatorname{Log}\left(R_{j(i)-1}^{-1} R\left(u\left(t_{j(i)}\right)\right)\right), \tag{5.20}$$

$$\mathcal{X}_{init} = \arg\min_{\mathcal{X}} \sum_{i=1}^{n+1} \boldsymbol{e}_{\boldsymbol{p}}\left(i\right)^\intercal \boldsymbol{e}_{\boldsymbol{p}}\left(i\right) + \boldsymbol{e}_R\left(i\right)^\intercal \boldsymbol{e}_R\left(i\right). \tag{5.21}$$

Given the initialized spline and surfel maps, we now detail our registration approach that adapts the method of Droeschel et al. (2017) to a sliding registration window. We model the likelihood of a scene surfel s observing a map surfel m as the following normal distribution with scene pose $T(t)$ defined by Eq. 5.16:

$$\Sigma_{sm}\left(T(t)\right) = \Sigma_m + R(t)\Sigma_s R(t)^\intercal, \tag{5.22}$$

$$\boldsymbol{d}_{sm}\left(T(t)\right) = T(t)\boldsymbol{\mu}_s - \boldsymbol{\mu}_m, \tag{5.23}$$

$$\boldsymbol{e}_{sm}\left(T(t)\right) \sim \mathcal{N}\left(\boldsymbol{d}_{sm}, \Sigma_{sm} + \sigma_l^2 I\right), \tag{5.24}$$

with a resolution-depending scaling term σ_l^2. For better readability, we will drop the time argument in the above equations whenever possible. Equation 5.24 is similar to the GICP model (Segal et al., 2009) and requires a hard decision on whether surfel s and surfel m correspond. Instead, a GMM allows a soft assignment by representing the mixture of surfel s observing multiple associated map surfels $\mathcal{A}_s$ according to Eq. 5.24 with a prior association likelihood $p(a_{sm})$, similarity $p(\delta_{sm})$ and additional uniform outlier component $p(o_s)$ (Droeschel et al., 2017) with weight ζ_0:

$$p_s\left(T\right) = p\left(o_s\right) + \sum_{m \in \mathcal{A}_s} p\left(a_{sm}\right) p\left(\delta_{sm}\right) p\left(\boldsymbol{e}_{sm}\right), \tag{5.25}$$

$$p\left(o_s\right) = p\left(o\right) p\left(\mathcal{N}\left(\boldsymbol{0}, R(t)\Sigma_s R(t)^\intercal + \sigma^2 I\right)\right), \tag{5.26}$$

$$= \zeta_0 \cdot \frac{M_s}{\sum_{s \in S_l} M_s} \frac{1}{\sqrt{(2\pi)^3 \det\left(R(t)\Sigma_s R(t)^\intercal + \sigma_l^2 I\right)}}, \tag{5.27}$$

$$p\left(a_{sm}\right) = (1 - \zeta_0)\frac{M_m}{\sum_{m \in \mathcal{M}} M_m}. \tag{5.28}$$

Although Eq. 5.27 contains $R(t)$, we precompute $p(o_s)$ using $\det(R) = \det(R^\intercal) = 1$ and $\det(AB) = \det(A)\det(B)$ for symmetric positive semidefinite (s.p.d.) matrices:

$$\sigma_o = \det\left(R(t)\Sigma_s R(t)^\intercal + \sigma_l^2 I\right) = \det\left(\Sigma_s + \sigma_l^2 I\right), \tag{5.29}$$

$$p\left(o_s\right) = \frac{M_s}{\sum_{s\in S_l} M_s} \cdot \frac{\zeta_0}{\sqrt{(2\pi)^3 \sigma_o}}. \tag{5.30}$$

Measuring the similarity of associated surfels in normal $\boldsymbol{n}$ and viewing direction $\boldsymbol{f}$ follows the simple normal distribution:

$$p_v\left(\boldsymbol{v}\right) \sim \mathcal{N}\left(\arccos\left(\boldsymbol{v}_m^\intercal R(t)\boldsymbol{v}_s\right), \left(\pi/8\right)^2\right). \tag{5.31}$$

For more details on the GMM, we refer the reader to Droeschel et al. (2017). Additionally, we consider the distance in the normal direction:

$$d_n \sim \mathcal{N}\left(\boldsymbol{n}_m^\intercal \Sigma_{sm}^{-1} \boldsymbol{d}_{sm}, \sigma^2\right). \tag{5.32}$$

Assuming independence, results in the following approximation for $p(\delta_{sm})$:

$$p(\delta_{sm}) = p_v(\boldsymbol{n}) p_v(\boldsymbol{f}) p(d_n). \tag{5.33}$$

Given our GMM formulation, we seek the spline control points $\mathcal{X}_l^\star$ that maximize the joint observation log-likelihood over the mixture for the current window $\mathcal{W}_l$:

$$\mathcal{X}_l^\star = \arg\max_{\mathcal{X}} \sum_{i=1}^{n} \sum_{s\in\mathcal{S}_i} \log\left(p_s\left(T_{\mathcal{X}}\left(t_{l-n+i}\right)\right)\right). \tag{5.34}$$

The two-step EM algorithm allows us to solve (5.34). In the E-Step, we establish the associations $\mathcal{A}_s$ to the local map $\mathcal{M}$ for all surfels within a surfel map $\mathcal{S}_i$ of the sliding registration window $\mathcal{W}_l$. For this, we transform the surfel mean $\boldsymbol{\mu}_s$ given the current pose $T_c = T_{\mathcal{X}}(t_i)$ from the sensor into the map frame and perform a lookup for valid surfels in the 1-hop-neighborhood of the corresponding surfel in the local map $\mathcal{M}$. A 3D voxel has up to 27 scan-map associations. At the same time, there is a maximum of 9 per lattice simplex. Given the current estimate T_c, we calculate the conditional likelihood w_{sm} for each association:

$$w_{sm}\left(T_c\right) = \frac{p\left(a_{sm}\right) p\left(\delta_{sm}\right) p\left(\boldsymbol{e}_{sm}\right)}{p_s(T_c)}. \tag{5.35}$$

Fixing the associations and weights w_{sm} during the M-Step allows us to optimize the control points $\mathcal{X}$ with the LM algorithm (Sec. 2.2) that minimizes the Mahalanobis distance r_{sm} between associated surfels for the current window $\mathcal{W}_l$ at time t_l:

$$r_{sm}(T_c) = \boldsymbol{d}_{sm}^\intercal \Sigma_{sm}^{-1} \boldsymbol{d}_{sm}, \tag{5.36}$$

$$\mathcal{X}_l^\star = \arg\min_{\mathcal{X}} \sum_{i=1}^{n} \sum_{s\in\mathcal{S}_i} \sum_{m\in\mathcal{A}_s} w_{sm} r_{sm}(T_{\mathcal{X}}(t_{l-n+i})). \tag{5.37}$$

We found empirically that the GMM assigns a higher weight w_{sm} in the vicinity of the 3D-LiDAR. This biases the estimate towards staying in place in situations with motion close to the sensor or underconstrained translation, e.g., in open park areas

with distant trees. Analysis of the surfel normals' covariance C_n and its condition number $\kappa(C_n)$ (see Sec. 2.3):

$$C_{\boldsymbol{n}} = w \cdot \boldsymbol{n}_s \boldsymbol{n}_s^\intercal, \tag{5.38}$$

$$\kappa(C_{\boldsymbol{n}}) = \|C_{\boldsymbol{n}}\| \, \|C_{\boldsymbol{n}}^{-1}\| = \frac{\lambda_{\max}(C_{\boldsymbol{n}})}{\lambda_{\min}(C_{\boldsymbol{n}})}, \tag{5.39}$$

showed that weighting with the conditional likelihood $w = w_{sm}$ increases the condition number $\kappa(C_{\boldsymbol{n}})$ — in some cases by a factor of up to 50 compared to $w = 1$. Such an increase is remarkable since κ reflects the difficulty of accurately solving a linear system of equations (Higham, 2002), like the normal equations in LM. The increase relates to the higher measurement density in the LiDAR vicinity due to the sensor geometry's non-uniform sampling in 3D because the prior association likelihood $p(a_{sm})$ and outlier component $p(o_s)$ incorporate the number of measurements per surfel. Thus, inverse weighting of $p(o_s)$ (Eq. 5.27) and $p(a_{sm})$ (Eq. 5.28) in w_{sm} with the number of measurements M_s and M_m levels the influence between close and far surfels. Moreover, the prior association likelihood $p\left(a_{sm}\right)$ is now a constant factor for the surfel s. We use the equality[1]:

$$\frac{a}{b+a} = \frac{1}{\frac{b}{a}+1} \tag{5.40}$$

to rephrase Eq. 5.35 as:

$$w_{sm}\left(T_c\right) = \frac{p\left(\delta_{sm}\right) p\left(\boldsymbol{e}_{sm}\right)}{\frac{p(o_s)}{p(a_{sm})} + \sum_{m \in \mathcal{A}_s} p\left(\delta_{sm}\right) p\left(\boldsymbol{e}_{sm}\right)}, \tag{5.41}$$

$$= \frac{\overline{w}_{sm}}{w_{\text{oa}} + \sum_{m \in \mathcal{A}_s} \overline{w}_{sm}}. \tag{5.42}$$

Here, the prior term w_{oa} combines Eq. 5.28 and Eq. 5.30 to:

$$w_{\text{oa}} = \frac{\zeta_0}{1-\zeta_0} \frac{\sum_{m \in \mathcal{M}} M_m}{\sum_{s \in \mathcal{S}_l} M_s} \frac{1}{\sqrt{(2\pi)^3 \sigma_o}}. \tag{5.43}$$

5.2.3 *Adaptive Resolution Selection*

Using the finest map resolution is computationally inefficient for planar surfaces like roads, floors, walls, or ceilings. Instead, we adaptively select a more appropriate resolution from finest to coarsest. We collect valid surfels starting on the finest map scale. If the normalized Eigenvalues $\lambda_0 \leq \lambda_1 \leq \lambda_2 \in \mathbb{R} \; s.t. \sum_i \lambda_i = 1$ of the surfel covariance matrix $\boldsymbol{\Sigma}$ satisfy one of the following conditions:

$$\lambda_0 < \alpha, \tag{5.44}$$

$$\lambda_0 < \beta \lambda_1, \tag{5.45}$$

$$\lambda_1 < \gamma, \tag{5.46}$$

1 The equality is the reciprocal of the equality: $\left(\frac{b}{a}+1\right) = \left(\frac{b+a}{a}\right)$.

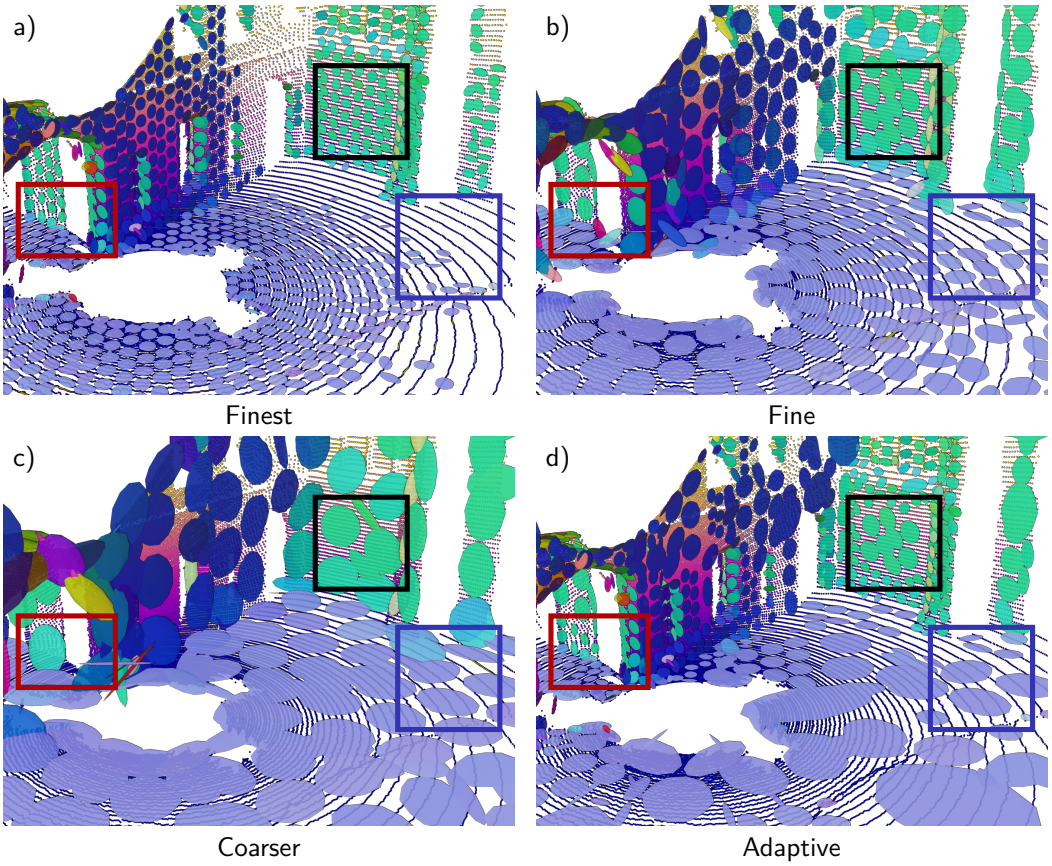

Figure 5.5: Adaptive resolution selection chooses coarser surfels in areas with planar surfels of finer resolution. We retain finer [red in a)] and reject coarser ones [black in c)] when approaching edges between surfaces to preserve details. Merging of multiple degenerate surfels [blue in b)] allows to harness more distant measurements in sparse areas.

the surfel s becomes a candidate for coarsening. The first two cases (Eq. 5.44 and Eq. 5.45) directly relate to planar and elongated surfaces, while the last case often occurs for degenerate surfels created from a single scan line. Figure 5.5 illustrates these.

We check the candidate's valid neighbors that fall within the same coarser surfel c as well as the coarser surfel and compute the finer ones' mean normal vector $\bar{\boldsymbol{n}}$. The coarse surfel c replaces the finer ones if all finer surfels become candidates and the coarse normal is similar to $\bar{\boldsymbol{n}}$:

$$|\boldsymbol{n}_c \cdot \bar{\boldsymbol{n}}| > \delta_n. \tag{5.47}$$

Additionally, we reject a coarse surfel if the orientations of its coarse neighbors vary too strongly (Eq. 5.47). This primarily occurs on the transition between multiple surfaces where we retain the finer resolution for higher detail.

The process repeats on the next coarser level until it reaches the coarsest level. Processing one coarser surfel instead of many smaller ones reduces the total number of surfels and speeds up the registration. We found empirically the following thresholds $\alpha = 0.01$, $\beta = 0.01$, $\gamma = 0.1$, $\delta_n = 0.8$.

5.3 Evaluation

We evaluate our method, called MARS, on the Newer College (Ramezani et al., 2020) dataset, the Urban Loco (Wen et al., 2020) dataset for autonomous driving, and nine self-recorded UAV flights through the DRZ Living Lab (Kruijff-Korbayova et al., 2021). Additionally, we show qualitative results from multiple UAV flights. All experiments were conducted on a laptop with an Intel Core i7-6700HQ CPU with 32 GB of RAM and an NVIDIA GeForce GTX 960M GPU.

The two LOAM variants A-LOAM[2] and F-LOAM[3], as well as SuMa[4], provide the baseline for our method. We enforce real-time processing in A-LOAM[5] and F-LOAM[6] using ROS message_filters[7] with a limited queue size of 10 to prevent the processing of too old information. Furthermore, we deactivated the loop-closing of SuMa for fair comparison. We omitted MRSLaserMaps as it lost tracking on all tested sequences.

Our method runs on the central processing unit (CPU) only and is implemented in C++. The LM algorithm performs up to three iterations to optimize (5.37) in all experiments. The default storage type is a sparse voxel grid unless otherwise noted.

For all methods, we compare the at-runtime estimated poses against the provided ground-truth using the root-mean-squared (RMS) absolute trajectory error (ATE). The best and second-best results will be in bold, respectively, underlined. For both LOAM-derivatives, we use the more accurate optimized poses instead of the initial odometry poses for ATE computation and ignore unoptimized scans.

5.3.1 *Newer College Dataset*

Ramezani et al. (2020) replicated the original trajectories from the New College Dataset (Smith et al., 2009) with a handheld 64-beam Ouster OS-1 LiDAR running at 10 Hz. The sequences "01_short" (NC01) and "02_long" (NC02) consist of multiple loops between buildings and through a park for over 1530 s and 2656 s. The shorter "05_quad" sequence (NC05) contains partially swinging and fast motion throughout four loops in a quad. Ground-truth poses stem from registering the LiDAR scans against a map obtained with a terrestrial laser scanner. Table 5.1 presents the resulting ATE. A-LOAM and F-LOAM only managed to fully optimize every third scan in real-time, although each scan was processed using its odometry.

SuMa lost track in the park area of the long sequence, whereas both LOAM variants drifted significantly during orientation changes. A-LOAM benefits in sequence "NC05" from the limited size of the quad since the local map optimization allows it to effectively relocalize and keeps the ATE low. In contrast, our sliding keyframe window accumulates drift over time. However, MARS maintains high accuracy such that extracting loop-closure candidates based on some vicinity criterion is possible.

2 https://github.com/HKUST-Aerial-Robotics/A-LOAM
3 https://github.com/wh200720041/floam
4 https://github.com/jbehley/SuMa
5 https://github.com/JanQuenzel/A-LOAM
6 https://github.com/JanQuenzel/floam
7 http://wiki.ros.org/message_filters

Table 5.1: RMS-ATE [m] evaluation on scenes from the Newer College (NC, Ramezani et al. (2020)) and Urban Loco (UL, Wen et al. (2020)) datasets. Lower values are better (↓) with second and **best** highlighted.

Sequence	Ours		A-LOAM	F-LOAM	SuMa
	Grid	Lattice			
NC01	2.1723	**1.9784**	3.3077	101.899	2.0481
NC02	**4.9355**	5.1124	62.6424	87.0877	X
NC05	0.4683	0.4151	**0.1482**	2.8184	1.8784
ULCT	4.9962	**4.9661**	10.3723	6.0562	7.7715
ULLS	7.8239	7.5732	8.4000	**7.5602**	10.6955
ULHH	2.3556	2.4079	2.4328	2.4242	**2.3256**
ULLL	2.7318	2.2330	**2.0709**	2.3138	8.4411
ULSL	3.1927	3.2661	3.1657	3.2705	**2.8777**
ULWH	2.9506	2.9943	2.4235	**2.2592**	3.5004
ULT2	2.5442	**1.4411**	17.3492	2.4282	2.2169
ULT3	1.5969	**1.5553**	9.6700	2.0297	2.9448
ULH5	1.5518	**1.5052**	17.3521	2.4282	2.2169

5.3.2 *Urban Loco Dataset*

Wen et al. (2020) equipped two cars with LiDAR and other sensors to capture highly urbanized areas throughout Hong Kong and San Francisco. A navigation system with IMU and RTK-GPS provides the reference poses for all sequences. A Velodyne HDL-32E LiDAR provides scans throughout the seven Hong Kong sequences, whereas the San Francisco datasets use an RS-LiDAR-32. The Hong Kong sequences are up to 2 km long through dense urban environments with durations between 150 s and 365 s. The Coli Tower sequence (ULCT) is a 1.8 km drive uphill within a dynamic environment that took 248 s. Similarly, the Lombard Street sequence (ULLS) is a 1 km drive over 253 s.

In comparison to the Newer College sequences, these sensor trajectories are smoother while exhibiting stronger variation in height. The street is measured primarily under steep angles and at greater distances whereas lateral obstacles, e.g., walls, are measured more uniformly w.r.t. range and surface normal. This may explain why the methods primarily deviate in their estimated height. These differences are clearly visible in the "ULLS" sequence in Fig. 5.6. The car starts to drive up the hill (from left to right) and makes a left turn at a junction halfway up (lower middle of the image). After three blocks, the car turns right and drives back downhill to the junction.

SuMa and A-LOAM overestimate the steepness of the streets and strongly underestimate the height.

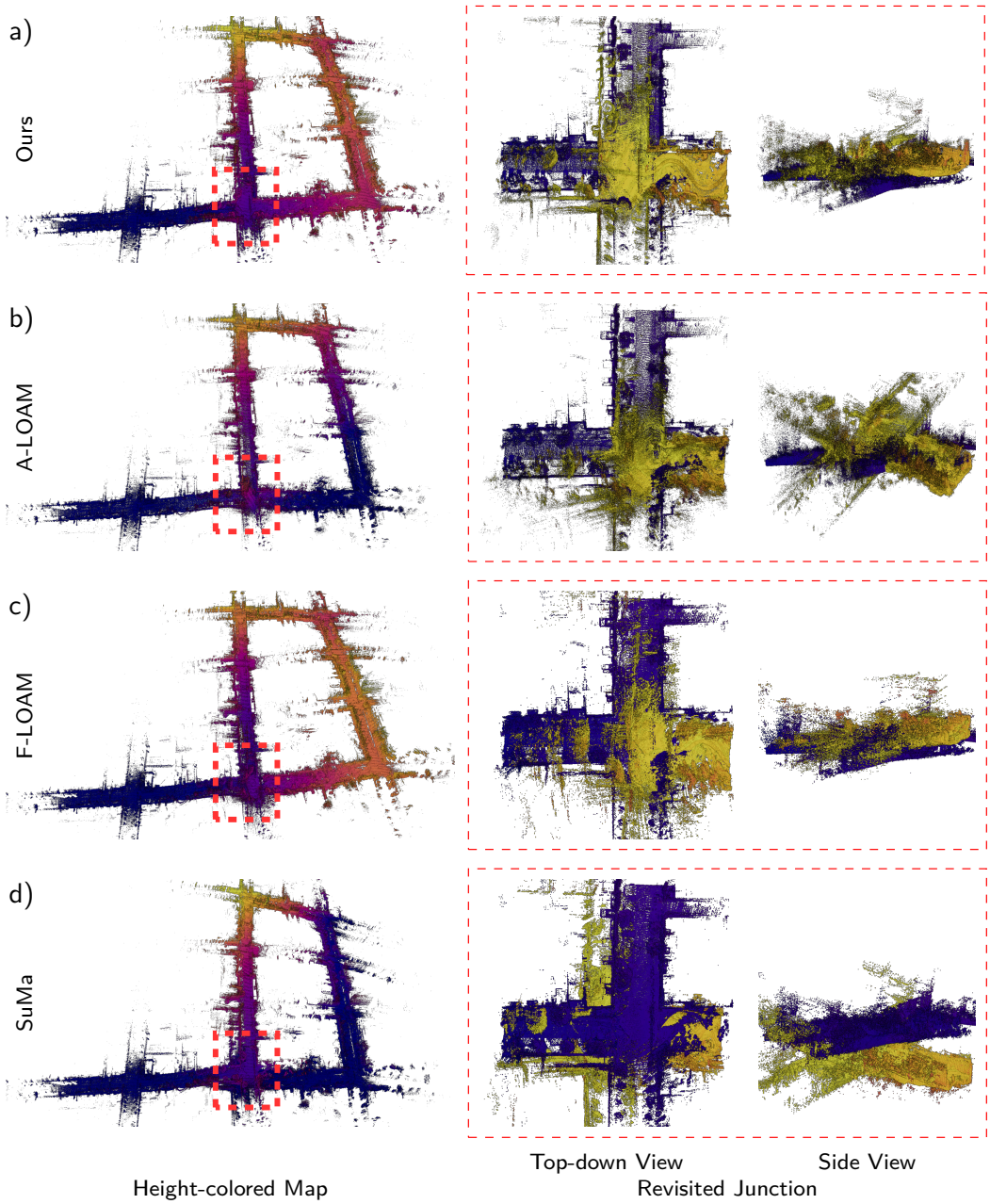

Figure 5.6: Maps (left) computed by our MARS [a)], A-LOAM [b)], F-LOAM [c)] and SuMa [d)] for the "Lombard Street" sequence (ULLS; Wen et al., 2020). Color encodes height from low (blue) to high (yellow). The car visits the highlighted junction [a-d), rectangle] twice. The close-ups in top-down (middle) and side view (right) reveal aggregated drift between the maps from the first traversal (blue) to the second traversal (yellow).

All methods estimate the sequence's end to be close to the previously visited junction with noticeable offsets for the position. Furthermore, A-LOAM and SuMa show apparent deviations in roll and pitch, whereas F-LOAM deviates in yaw.

Our method, MARS, achieves consistent state-of-the-art results, as shown in Tab. 5.1. MARS places first in three and second in another two out of nine sequences.

Table 5.2: Statistics and RMS-ATE [m] evaluation for the DRZ Living Lab dataset. Lower ATE values are better (↓) with second and **best** highlighted.

Seq.	Scans	med$\|\boldsymbol{a}\|$ [m/s^2]	med$\|\boldsymbol{\omega}\|$ [rad/s]	max$\|\boldsymbol{\omega}\|$ [rad/s]	Ours		A-LOAM	F-LOAM	SuMa
					Grid	Lattice			
Fast	1479	1.1312	0.2288	1.8936	0.0949	**0.0557**	2.2646	2.1102	0.0637
Hall	2421	0.3989	0.0976	1.1468	0.2054	0.0921	**0.0303**	0.0385	0.0726
3P-S	1085	0.2744	0.0790	0.8619	0.0218	**0.0147**	0.0224	0.0167	0.0383
3P-M	650	0.5892	0.1135	1.0811	0.0287	**0.0239**	0.0252	0.0308	0.0403
3P-F	604	1.9902	0.5254	3.5473	0.0607	**0.0500**	0.7180	0.8317	0.0947
S1	826	0.4888	0.1044	1.5156	0.0440	**0.0397**	0.0456	0.0464	0.0420
M1	1458	0.9001	0.2192	1.9784	0.0822	**0.0639**	3.7353	3.6215	0.1757
F2	795	2.3075	0.4866	3.9811	0.1046	0.1002	2.6255	2.8037	**0.0884**
F3	957	1.0491	0.2494	1.6319	0.0852	**0.0737**	0.1441	2.4597	0.1799

5.3.3 *DRZ Living Lab*

We collected LiDAR scans onboard a DJI M210 v2 with a 128-beam Ouster OS-0 running at 10 Hz while flying through the DRZ Living Lab[8]. A Motion Capture (MoCap) system in the lab's starting area provides ground-truth poses for multiple runs with up to 2.5 min of flight time. All sequences, except for the "Hall", remain within the MoCap volume with varying linear acceleration and angular velocity, as shown in Tab. 5.2.

In the "Hall" sequence, the UAV traverses back and forth through the Living Lab. Figure 5.7 shows the aggregated point cloud for this sequence using poses from our MARS. The visible tube-like artifacts stem from people moving through the scene. At the time of recording, the MoCap coverage was limited to the left half of the building. Hence, the UAV started and landed within the confines of the MoCap volume.

We recorded multiple runs with a slow, medium, and fast flying UAV in a static environment as well as with three people moving through the scene. For fair evaluation, we compensate for scan distortion due to the rotation with the UAVs' IMU orientation and supply all methods with these compensated scans. Our MARS using the permutohedral lattice outperforms the other approaches, as shown in Tab. 5.2. The exception is the "Hall" sequence, which we attribute to the accumulated drift from the sliding keyframe window similar to the results on "05_quad" the Newer College Dataset. A-/F-LOAM struggle on the faster sequences even with undistorted scans, while SuMa provides consistent results.

We evaluate multiple spline parameter sets on the challenging sequence "F2" to analyze the parameter's impact on accuracy and runtime during dynamic flight. The storage type was set to a sparse voxel grid since the timing for sparse and block-sparse grids were similar due to the low surfel count. Table 5.3 reports the resulting ATE with the time spent during registration for a varying number of control points N and the number of jointly optimized scans L. Optimizing more than three scans or

8 https://rettungsrobotik.de/living-lab/

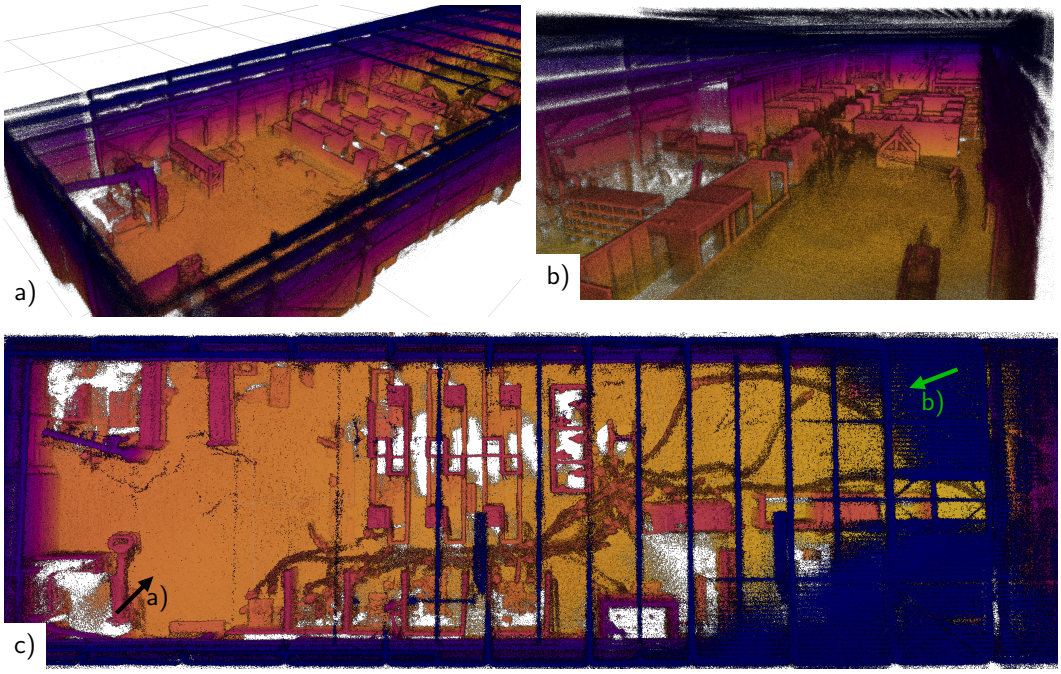

Figure 5.7: Aggregated point cloud from a traverse through the DRZ Living Lab. The left half contained the MoCap volume. The left and right arrows in c) show the view directions of a) and b). Tube-like artifacts stem from people moving throughout the lab during recording. The roof is partially removed for better visualization.

Table 5.3: Statistics for varying spline parameters using sparse voxel grids with adaptive resolution selection on the "F2" sequence in the DRZ Living Lab. Entries with * required five instead of three iterations. Second and **best** are highlighted.

Spline		ATE	Avg. Time
N	L	[m] (↓)	[ms] (↓)
2	2	0.1063	**41.28**
	3	0.1097*	65.45
	4	0.2247*	71.03
3	3	**0.1046**	50.57
	4	0.1076*	82.53
	5	0.1123*	91.85
4	4	0.1113	117.03
	5	0.1125	121.88
	6	0.1233	121.88

increasing the spline order did not provide any benefit in terms of accuracy while requiring more LM iterations and thus slows computation down.

For our standard parameters $N = 3$ and $L = 3$, we further analyze the influence of the map representation and the selected resolution. Switching from a (block-)sparse voxel grid to the permutohedral lattice reduces the average runtime from 50.57 ms

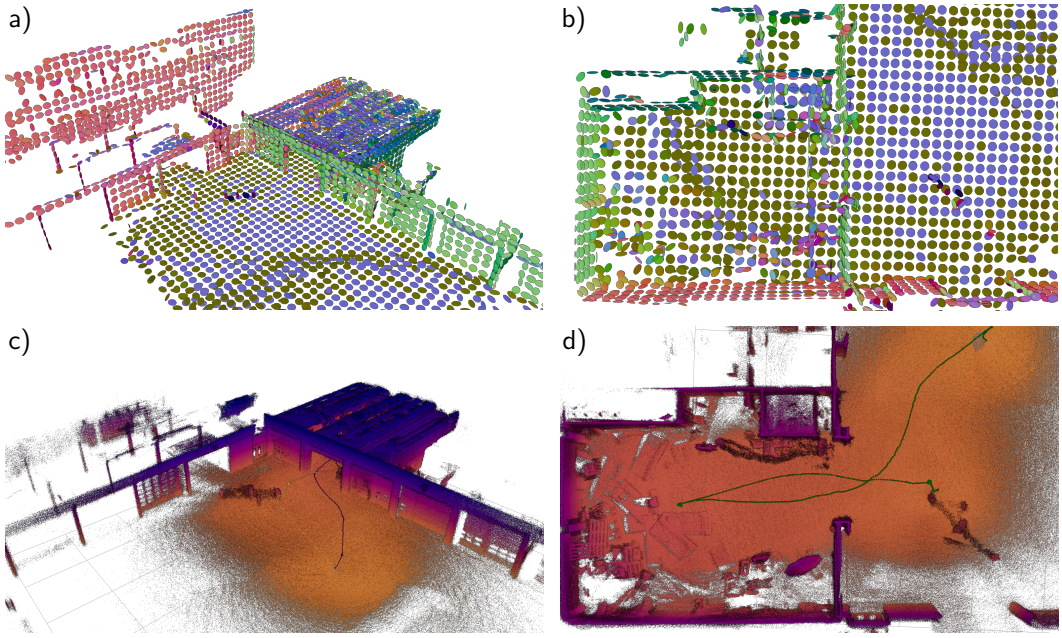

Figure 5.8: Local surfel map [a), b)] and aggregated point cloud [c), d)] from an autonomous flight into a garage at the LBH in Bonn, Germany. The green line in c) and d) shows the flown trajectory.

down to 22.95 ms. Furthermore, Tab. 5.1 and Tab. 5.3 show that the lattice obtains lower ATE on average.

Although disabling the adaptive resolution selection for the lattice reduces the ATE on "F2" from 0.1002 m to 0.0949 m, the registration required 29.82 ms. In contrast, we obtained more consistent and accurate results in open environments with enabled adaptive resolution selection. Similarly, the voxel grid without adaptive selection could not correctly estimate all rotations, resulting in a 90° drift during a high angular velocity maneuver.

5.3.4 *Qualitative UAV Experiment*

An early development version of MARS provided the onboard LO for multiple autonomous flights (Schleich et al., 2021) of the DRZ technology demonstrator D1 in GNSS-denied areas. For this, MARS processed the OS-0 LiDAR scans in real-time on the Intel Core i7-8559U CPU of an Intel NUC attached to the DJI M210v2 UAV. During these flights, the state estimation fused our estimated LiDAR poses with the aircraft's IMU in an EKF. We refer the reader to Schleich et al. (2021) for more details on the autonomous UAV flights. Figure 5.8 shows an example of the finest resolution of the local surfel map and the corresponding aggregated point cloud during an autonomous mission at the LBH in Bonn, Germany.

After publication, the open-sourced version of MARS continued to be a key component of the D1 UAV with many more successful autonomous flights.

5.4 Summary

In this chapter, we presented a novel continuous-time LiDAR odometry called MARS and verified our key claims. MARS embeds individual LiDAR scans at multiple resolutions into a sparse permutohedral lattice or voxel grid to store the individual surfels. Our approach jointly aligns multiple surfel maps from the sliding registration window with the local map using a GMM formulation. For this, we directly optimize the control points of the continuous-time B-spline trajectory.

The local map maintains a sliding window over the last keyframes using an efficient surfelwise fusion without costly pointwise reintegration. This fusion facilitates spatial shifting of the map to ensure locality. Furthermore, we adaptively select the appropriate surfel resolution, e.g., on planar surfaces, to improve the run-time. The sparse permutohedral lattice further reduces the number of neighboring surfels to consider during the GMM-based soft-assignment. Moreover, our adapted GMM improved numerical stability, making it more suitable for typical LiDAR sensor geometry.

We compared MARS against multiple state-of-the-art LO methods on a variety of datasets, from automotive driving to handheld sensors. Additionally, we recorded multiple challenging sequences onboard a rapidly flying UAV with reference trajectories from a MoCap system at the DRZ Living Lab. Overall, MARS shows state-of-the-art performance on all evaluated datasets.

Our system runs in real-time onboard a UAV, enabling safe operation in GNSS-denied environments. MARS was a core component, responsible for map creation and state estimation, during numerous autonomous missions. The code of our method is publicly available to foster future research in this direction and for the benefit of the research community. Moreover, MARS laid the foundation for registering multi-resolution surfel maps against georeferenced models (Quenzel et al., 2024).

In the following chapter (Ch. 6), we extend MARS to the more accurate and faster LIO-MARS.

6

LiDAR Inertial Odometry

Reliable real-time perception is essential for robotic autonomy. In particular, accurate mapping and ego-motion estimation are key components for safe interaction in complex and unstructured environments. Due to their precision and measurement density, modern LiDARs are often used in these scenarios, e.g., in the DARPA Subterranean Challenge (Khattak et al., 2020; Zhao et al., 2021; Reinke et al., 2022; Zhao et al., 2024).

Sensor motion during scanning distorts the point cloud and degrades the quality of the map. This intra-scan motion is either compensated by de-skewing prior to registration (Li et al., 2021; Quenzel and Behnke, 2021; Xu et al., 2022; Chen et al., 2023a; Vizzo et al., 2023) or by modeling it with a continuous-time trajectory (Droeschel and Behnke, 2018; Lv et al., 2021; Dellenbach et al., 2022). The former uses the previous state estimate and, optionally, an inertial measurement unit (IMU) to predict the motion and transform points to a common reference time. The latter approach optimizes the trajectory directly at intermediate time steps. However, this comes at the cost of reduced real-time capability and requires either costly reintegration of surfels (Droeschel and Behnke, 2018) or a limited number of selected pointwise features [e.g., CT-ICP (Dellenbach et al., 2022), CLINS (Lv et al., 2021)].

To overcome these limitations of continuous-time methods, our novel real-time LiDAR-inertial odometry (LIO) jointly optimizes temporally partitioned scan segments (Fig. 6.1) by registering multi-resolution surfel maps (see Sec. 5.2.1) while an unscented transform (UT) compensates the intra-surfel motion. Rephrasing the computation of the Gaussian mixture model (GMM) and surfels using vectorized Kronecker sums and products (Lancaster and Tismenetsky, 1985; Horn and Johnson, 1991) reduces redundancy and improves processing speed. We introduce *relative* inertial and motion constraints from complementary modalities, such as IMU and robot odometry, and derive their analytic Jacobians w.r.t. the spline knots to increase robustness and accuracy. Furthermore, we use a *non-uniform* continuous-time B-spline trajectory as an elegant solution to address variations in scan time without increased delay [e.g., as in Coco-LIC (Lang et al., 2023)] at greater numerical stability compared to its uniform counterpart.
In summary, we thoroughly evaluate the proposed LIO-MARS to support our key claims, which are:

- First, the non-uniform spline has improved numerical stability in real robotics applications.
- Second, an unscented transform (UT) enables motion compensation for individual surfels.
- Third, a timewise separation into intra-scan segments facilitates motion compensation at optimization time.

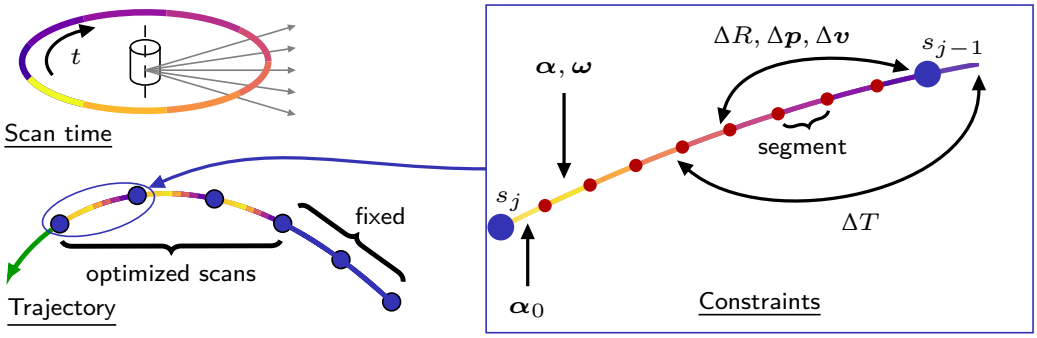

Figure 6.1: Joint registration of LiDAR scans embedded in multi-resolution surfel maps (colored by relative scan time) optimizes a *non-uniform* continuous-time B-spline trajectory. An UT compensates motion within a surfel, while a temporal partitioning into segments enables optimization of motion distortion between surfels intrinsically. Inclusion of relative $(\Delta R, \Delta \boldsymbol{v}, \Delta \boldsymbol{p}, \Delta T)$ and absolute $(\boldsymbol{\alpha}_0, \boldsymbol{\alpha}, \boldsymbol{\omega})$ soft-constraints further improves robustness in challenging situations.

- Fourth, leveraging relative inertial and motion constraints improves accuracy.
- Fifth, rephrasing the Gaussian mixture model (GMM) and surfel covariances with Kronecker sum and products improves parallelization.

We will open-source LIO-MARS at: `https://github.com/AIS-Bonn/lio_mars`.

Preface

This chapter is an extension of Quenzel and Behnke (2021), previously published by IEEE and presented at the International Conference on Intelligent Robots and Systems (IROS 2021).

Statement of Personal Contribution

> "The author of this thesis [Jan Quenzel] substantially contributed to all aspects of the previous publication (Quenzel and Behnke, 2021), including the conception, literature survey, design, and implementation of the proposed methods, the preparation and conduct of experiments and evaluation of the proposed approach, conducting the experimental evaluation, the analysis and interpretation of the experimental results, drafting the manuscript, as well as the revision and final approval of the version to be published."

6.1 Related Work

In Ch. 5, we reviewed the state-of-the-art for LiDAR-only odometry. As a consequence, we focus on LIO systems in this chapter.

The alignment of new scans commonly involves an iterative closest point (ICP)-variant (Besl and McKay, 1992; Segal et al., 2009) with local convergence. As such, it is crucial to obtain a good initialization, e.g., from preintegration of IMU

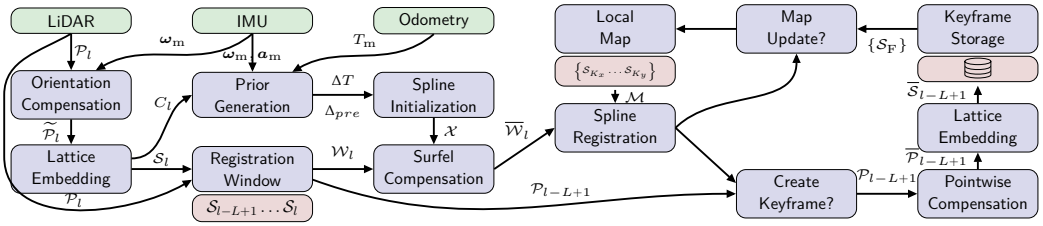

Figure 6.2: System overview: A non-uniform continuous-time spline trajectory defined by knots $\mathcal{X}$ describes the sensor motion. A new raw scan $\mathcal{P}_l$ is preoriented with IMU before lattice embedding into a multi-resolution surfel map $\mathcal{S}_l$. Motion priors $(\Delta T, \Delta_{pre})$ aid the spline initialization of the sliding registration window $\mathcal{W}_l$. Sensor motion within surfels is compensated prior to alignment against a keyframe-based local surfel map $\mathcal{M}$ under motion constraints. After spline registration, a new pointwise undistorted keyframe is added to the storage if necessary, or the local map is updated with the closest keyframes.

measurements (Shan et al., 2020; Zhao et al., 2021; Xu et al., 2022) or wheel/robot odometry (Reinke et al., 2022; Guadagnino et al., 2025). Despite the inclusion of priors, the registration may slip or diverge (Zhang et al., 2016), e.g., in featureless corridors (Nashed et al., 2021) or in open areas (Tuna et al., 2024, 2025).

Hence, LIO systems (Shan et al., 2020; Li et al., 2021; Xu and Zhang, 2021; He et al., 2023; Jung et al., 2023; Chen et al., 2024b; Zhang et al., 2024b) optimize Light Detection and Ranging (LiDAR) jointly with IMU at discrete timesteps (e.g., end of scan) which requires temporal interpolation. The widely adapted iterated error-state Kalman filter (IEKF) of Fast-LIO (Xu and Zhang, 2021; Xu et al., 2022) propagates the state forward with IMU measurements before back-propagating it to the respective point times to undistort the scan. As an alternative to processing the accumulated scan, Zhang et al. (2024b) adapt the sliding window length depending on the spatial overlap between the map and the voxelized scan. Point-LIO (He et al., 2023) only processes points with the same timestamp at once, whereas Liu et al. (2024) employ Kalman smoothing over the last scan.

Complementary to previous approaches, RI-LIO (Zhang et al., 2023a) and COIN-LIO (Pfreundschuh et al., 2024) exploit per-point reflectivity estimates from the LiDAR.

A continuous-time trajectory representation facilitates the inclusion of multi-modal data. Talbot et al. (2025) extensively review continuous-time state estimation, dividing methods into three groups.

The first group applies linear interpolation (Lovegrove et al., 2013; Park et al., 2018; Daun et al., 2021) between consecutive states. Per scan, the state typically includes the position and orientation with optional velocity and IMU biases. The *offline* simultaneous localization and mapping (SLAM) method SLICT (Nguyen et al., 2023) uses this trajectory representation to align a scan window using the point-to-plane error against a hierarchical surfel map. To prevent over-smoothing, Dellenbach et al. (2022) assume continuity during each scan, with residuals influencing the scan's start and end pose, but decouple consecutive scans from one another. SE-LIO (Yuan et al., 2025) weakens this assumption by enforcing similarity between the end and subsequent start pose while restraining inter-scan motion via preintegrated IMU.

The second group describes the continuous-time trajectory using splines with temporal basis functions. Our LO MARS (Ch. 5) belongs to this group and uses a flexible uniform Nth order B-spline (Sommer et al., 2020) for a sliding registration window. The registration jointly optimizes multi-resolution surfel maps for multiple scans without enforcing continuity outside the window.

In contrast, CLINS (Lv et al., 2021) targets *offline* SLAM and retains points on planar surfaces and edges (Zhang and Singh, 2014). The full trajectory is represented with the same uniform B-spline as MARS but minimizes errors on raw IMU measurements and point-to-plane(-line) distance for undistorted planar (resp. edge) points with automatic differentiation using Ceres Solver (Agarwal et al., 2022). The follow-up work, CLIC (Lv et al., 2023), integrates analytical Jacobians and a camera-based frontend. Concurrently, SLICT2 (Nguyen et al., 2024b) adopts the same trajectory as SLICT and follows our iterated expectation maximization (EM)-strategy (Sec. 5.2.2) of alternating between correspondence search and spline optimization without multiple inner iterations. Coco-LIC (Lang et al., 2023) further introduces a non-uniform cubic B-spline to adaptively select the number of knots per scan depending on the measured angular velocity and linear acceleration.

The third group focuses on temporal Gaussian processes (GPs) where the process model $\mathcal{GP}(\boldsymbol{\mu}(t), \Sigma(t_i, t_{i-1}))$ describes the transition between temporally adjacent states with prior mean $\boldsymbol{\mu}(t)$ and prior covariance $\Sigma(t_i, t_{i-1})$ functions. Although GP-based approaches may be seen as a weighted combination of infinite temporal basis functions (Tong et al., 2013), evaluating the trajectory at time $\tau \in [t_{i-1}, t_i)$ involves only the states at t_i and t_{i-1} for priors based on linear, time-varying stochastic differential equation (SDE) (Anderson et al., 2015). Common priors assume white noise on acceleration (Talbot et al., 2025), velocity (Zheng and Zhu, 2024a), or jerk (Nguyen et al., 2024a). In contrast to spline-based methods, the state space typically contains the velocity in addition to the pose (Wu et al., 2023; Burnett et al., 2024). Traj-LIO (Zheng and Zhu, 2024a) replaces the linear interpolation within Traj-LO (Zheng and Zhu, 2024b) with GP interpolation using various GP motion priors.

The aforementioned continuous-time methods expect regular scan input and degrade with missing or irregular intermediate scans. In these situations, a non-uniform continuous-time B-spline adapts better while being equivalent to its uniform counterpart for regular scan input. Hence, we perform real-time LIO with non-uniform continuous-time B-splines with analytic Jacobians and motion compensation during optimization. For this, we extend MARS (see Ch. 5) by introducing a non-uniform B-spline and tightly coupling LiDAR with IMU. The analytical Jacobians for full relative motion constraints applied to the spline are derived and included to improve consistency and robustness while maintaining real-time processing.

6.2 Our Method

We take a raw LiDAR scan $\mathcal{P}_l$ in the sensor frame captured at time t_l as input. $\mathcal{P}_l$ consists of a set of measurements with range $r \in \mathbb{R}$ and direction $\overrightarrow{\boldsymbol{v}} \in \mathbb{R}^3$. This yields a point $\boldsymbol{p} = r\overrightarrow{\boldsymbol{v}}$ at time $t(\boldsymbol{p}) \in (t_l - \Delta t_e, t_l]$ with scan duration Δt_e. During one revolution, the LiDAR measures h ranges (e.g., 64 or 128) simultaneously under

w varying directions (e.g., 1024) with small fixed directional offsets o_r. The scan timestamp t_l corresponds to the last acquired point(s) within the LiDAR's revolution. In the absence of a measured point time $t(\boldsymbol{p})$, we estimate $t(\boldsymbol{p})$ from the organized image-like structure $(h \times w)$ using the column index $u_{\boldsymbol{p}} \in [0, w)$ of point $\boldsymbol{p}$, which directly relates to the time within a scan revolution as:

$$t(\boldsymbol{p}) = t_l - \Delta t_e + \frac{\Delta t_e}{w} u_{\boldsymbol{p}}. \tag{6.1}$$

If the column index is unavailable, we compute $u_{\boldsymbol{p}}$ using the azimuthal angle in spherical sensor coordinates with sensor offset o_s:

$$u(\boldsymbol{p}, o_r) = \frac{w}{2\pi} \arctan(p_y / p_x) + o_s + o_r. \tag{6.2}$$

During a revolution, the sensor motion continuously influences the scan origin and measurement direction. We obtain an initial estimate of the sensor's change in orientation ΔR_{ref} from integration of raw IMU measurements of angular rate $\boldsymbol{\omega}_{\text{m}}$ at t_{m_j} with the exp-map (Eq. 2.67):

$$\Delta R_j = \Delta R_{j-1} \exp_R \left(\boldsymbol{\omega}_{\text{m}_j} \right), \text{ with } \Delta R_0 = I, \tag{6.3}$$

$$\Delta \mathcal{R}_{\text{IMU}} = \left\{ \Delta R_j | \forall j \text{ s.t. } t_{\text{m}_j} \in (t_l - \Delta t_e, t_l] \right\}. \tag{6.4}$$

The rotations $\Delta \mathcal{R}_{\text{IMU}}$ enable us to use spherical linear interpolation (slerp) (Shoemake, 1985) on $\Delta \mathcal{R}_{\text{IMU}}$ to pre-orient $\mathcal{P}_l$ to a common reference time t_{ref}:

$$\Delta R_{\text{ref}} = \text{slerp}(\Delta \mathcal{R}_{\text{IMU}}, t_{\text{ref}}), \tag{6.5}$$

$$\widetilde{\boldsymbol{p}} = \Delta R_{\text{ref}}^{-1} \, \text{slerp}(\Delta \mathcal{R}_{\text{IMU}}, t_{\boldsymbol{p}}) \, \boldsymbol{p}. \tag{6.6}$$

Then, the oriented scan $\widetilde{\mathcal{P}}_l$ is embedded into a local multi-resolution sparse lattice $\mathcal{S}_l$ with tetrahedral cells (Sec. 5.2.1) and adaptive side length. Each cell stores a surfel s with mean $\boldsymbol{\mu}_s$ and covariance Σ_s for the embedded points. The surfel normal $\boldsymbol{n}_s$ corresponds to the Eigenvector $\boldsymbol{v}_0$ of the smallest Eigenvalue λ_0 of Σ_s.

We insert the surfels in $\mathcal{S}_l$ into the sliding registration window $\mathcal{W}_l$, as shown in Fig. 6.2. The window contains the last L scans (Sec. 6.2.2) with their trajectory represented by a non-uniform continuous-time Lie-Group B-spline $T_{\mathcal{X}}(t)$ (Sec. 6.2.1). From $\mathcal{S}_l$, the scan covariance C_l (Sec. 6.2.8) is computed to weight a motion prior ΔT from poses of robot odometry T_{m}. Then, the prior ΔT and the preintegrated IMU measurement Δ_{pre} (Shan et al. (2020); Sec. 6.2.7), from angular rate $\boldsymbol{\omega}_{\text{m}}$ and linear acceleration $\boldsymbol{a}_{\text{m}}$, aid the initialization of spline knots $\mathcal{X}$. The spline allows motion compensation for surfels within the window $\mathcal{W}_l$ (Sec. 6.2.9).

The registration (see Sec. 6.2.3) aligns the compensated window $\overline{\mathcal{W}}_l$ with the local surfel map $\mathcal{M}$ (see Sec. 5.2.1) by optimizing the knots $\mathcal{X}$ of the trajectory spline $T_{\mathcal{X}}(t)$. The local surfel map $\mathcal{M}$ contains surfels of selected spatially separated scans. If necessary, the oldest raw scan $\mathcal{P}_{l-L+1}$ is pointwise motion compensated towards t_{l-L+1} and re-embedded before its surfels $\overline{\mathcal{S}}_{l-L+1}$ are added as a keyframe to the keyframe storage (see Sec. 6.2.10).

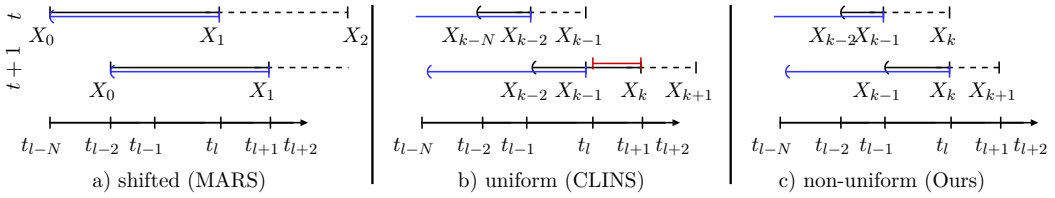

Figure 6.3: Influence of knot placement for $N = 3$: a) Knots X_k influence the time (black/blue) between t_{X_0} and t_{X_1}. MARS (Ch. 5) necessitates reinitialization of its knots at every timestep since the uniform spline requires a constant Δt (black) between knots while the knot times t_X move forward with variable Δt_l. b) Uniform knot placement with fixed Δt as in CLINS (Lv et al., 2021) may enforce continuity by appending new and fixing previous knots. The difference between scan time t_l and furthest knot X_{k+1} [red in b)] impairs constraints on X_{l-1}. c) Our non-uniform window has minimal difference and thus constrains the furthest knot better.

6.2.1 *Non-Uniform Continuous-time Trajectory*

Sommer et al. (2020) define an Nth order continuous-time cumulative B-spline trajectory $T_{\mathcal{X}}(t) \in SE(3)$ with rotation $R(t) \in SO(3)$, translation $\boldsymbol{p}(t) \in \mathbb{R}^3$ and knots $\mathcal{X}$. Each knot $X_k \in \mathcal{X}$ is a tuple $(R_k, \boldsymbol{p}_k)$ of the composite manifold $SO(3) \times \mathbb{R}^3$ and temporally placed at time t_k. Then, N temporally uniform-spaced knots $\{X_k, X_{k+1}, ..., X_{k+N-1}\}$ describe the trajectory interval $[t_k, t_{k+1})$ with duration $\Delta t > 0$. In the non-uniform case, $\Delta t_k = t_{k+1} - t_k > 0$ may vary, thus, leading to interval-specific B-spline basis[1] matrices (Qin, 1998) and requiring $2(N-1)$ timestamps:

$$\{t_{k-N+2}, \dots, t_k, \dots t_{k+N-1}\}\,. \tag{6.7}$$

Qin (1998) provides a recursive algorithm to compute non-uniform basis matrices for order N and closed forms up to $N = 4$.

We use this non-uniform continuous-time trajectory $T_{\mathcal{X}}(t)$ to represent the sensor pose $T_{\mathrm{m,s}}$ relative to the local surfel map $\mathcal{M}$. Hence, a point $\boldsymbol{p}_{\mathrm{s}}$ in the sensor frame corresponds to the point $\boldsymbol{p}_{\mathrm{m}} = T_{\mathrm{m,s}} \cdot \boldsymbol{p}_{\mathrm{s}}$ in the local map frame. Initially, we set the world frame to coincide with the local map frame $T_{\mathrm{w,m}} = I$. To maintain the locality of the local map, shifting gradually changes $T_{\mathrm{w,m}}$ by an integer multiple of the coarsest cell size while the orientation remains unchanged.

As is common in state estimation with inertial sensors (Qin et al., 2020; Sommer et al., 2020; Xu et al., 2022), we select the IMU as the reference sensor and transform scans prior to their lattice embedding into the reference frame with the IMU-LiDAR extrinsic $T_{\mathrm{s,l}}$. If no IMU is available, we set $T_{\mathrm{s,l}} = I$.

6.2.2 *Sliding Registration Window*

Our previous approach MARS (Ch. 5) optimizes the trajectory $T_{\mathrm{w,l}}(t)$ of L scans within the interval $[t_{l-L}, t_l]$ and requires reinitialization as t_k (see Fig. 6.4) and Δt change (see Fig. 6.3). Instead, CLINS (Lv et al., 2021) and SLICT2 (Nguyen

1 Sommer et al. (2020) refer to basis matrices as blending matrices.

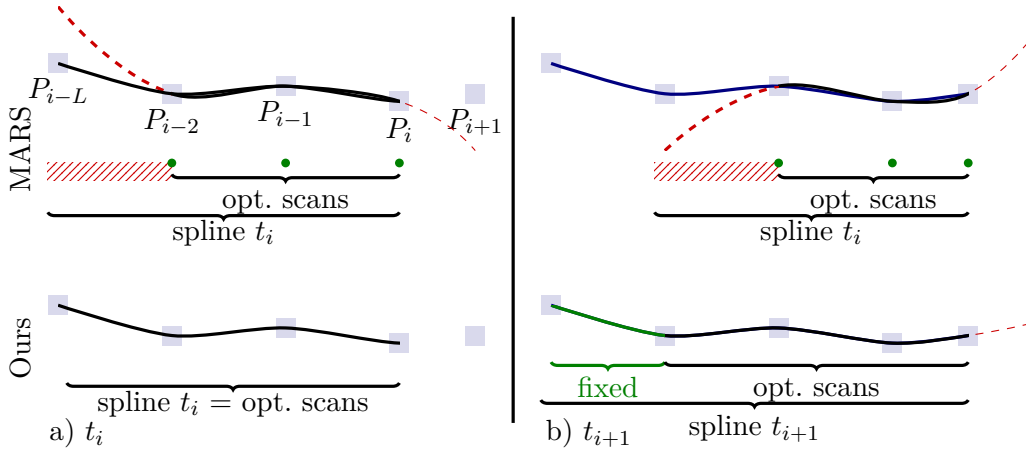

Figure 6.4: Spline window: a) MARS (Ch. 5) only optimizes a Nth order spline for L scans (e.g., $N = L = 3$) for a single interval Δt from scan $j - L$ until j allowing discontinuity between $j - L$ and $j - L + 1$ (left, red dashed). b) Our method enforces continuity and keeps previous intervals fixed while optimizing one interval per scan.

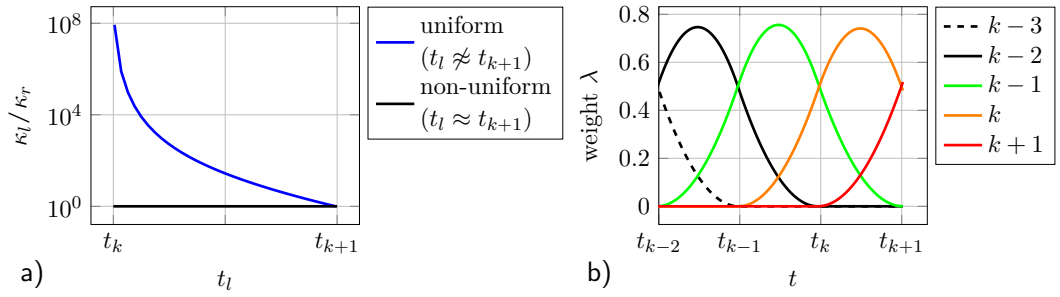

Figure 6.5: Ill-conditioning of uniform B-spline w.r.t. the last constraint. The condition number κ_l improves for t_l approaching t_{k+1} [a)] since the weight λ [b)] increases for the newest knot $(k + 1)$.

et al., 2023) fixate older knots not contributing to the current scan resp. window, and add new knots uniformly such that $t_l \in [t_k, t_{k+1})$. Although Coco-LIC (Lang et al., 2023) uses a non-uniform B-spline, the approach retains the notion of a fixed time interval Δt, e.g., 0.1 s, and subdivides each interval $[t_\kappa, t_{\kappa+1})$ into a variable number of uniformly placed knots depending on IMU excitation. Below a certain IMU threshold, the subdivision leads to a uniform B-spline, as with CLINS. Moreover, their IMU-based selection strategy delays the optimization by up to $(N - 1)\Delta t$ time intervals, e.g., 0.2 s for low IMU excitation. This is a direct consequence of the basis matrix computation[2] since IMU measurements within $[t_{\kappa+N-1}, t_{\kappa+N})$ influence the knot times (Eq. 6.7) required for $[t_\kappa, t_{\kappa+1})$.

The normal equations easily become ill-conditioned if the current scan[3] ends very close to the previous knot $(t_l - t_k \ll t_{k+1} - t_l)$. We show this in Fig. 6.5 by analysis

2 Although not explicitly stated in Lang et al. (2023), the basis matrix for $[t_\kappa, t_{\kappa+1})$ requires the knot time t_{k+N-1} where $t_{k+N-1} > t_{\kappa+1}$.

3 On the Newer College "Cloister" sequence (Zhang et al., 2021), around half of all scans have a Δt slightly larger than the expected $\Delta t_e = 0.1$ s.

of the condition number of the normal equations' $H = JWJ^\intercal$ matrix. For a spline with $N = 3$ and a window of $L = 3$ scans, we evaluate over 100 uniform time steps in $[t_{l-L}, t_l]$ under the assumption of uniform data constraints (e.g., $W = I_6$). The ill-conditioning becomes apparent by analysis of the condition number κ_l for t_l relative to the reference κ_r with $t_l \approx t_{k+1}$. The reason behind this is the small B-spline knot weight at the end (resp. beginning) of the knot's local support. We found that the optimization may approach local minima where the end knots are far away from reasonable values.

Without further constraints, this ill-conditioning leads to unrealistic motion, e.g., with high acceleration. We investigate the relevance of this situation by computing the difference between knot and scan time $(t_{k+1} - t_l)$ for CLINS[4] on the Newer College dataset (Zhang et al., 2021). Ideally, the difference is close to zero. Our test showed that 11.4 % of those time differences are above 40 ms (or 80 % of Δt) on the "Quad-Hard" sequence. The average is around 10.8 ms or 21.7 % of Δt. The situation is worse for the "Cloister" sequence, where the average is around 21.6 ms or 43.18 % of Δt. Here, 42 % of all time differences are above 40 ms (or 80 % of Δt). Accordingly, it is a common occurence and not the exception.

We combine the approaches of MARS and CLINS more robustly. Our non-uniform spline sets the knot time such that $t_{k+1} = t_l + \epsilon$ with $\epsilon = 1\,\text{ns}$ and thus $\kappa_l \approxeq \kappa_r$ prevents the above ill-conditioning. Furthermore, we optimize the L knots influencing the newest L intervals such that our window is $\mathcal{W}_l = \{S_{l-L+1}, \dots S_l\}$. Each interval spans the time between two consecutive scans. Knots $X_{k+2}, \dots, X_{k+N-2}$ are set with the expected constant Δt_e and are temporally corrected once a new scan arrives. Hence, we do not introduce a delay as in Coco-LIC (Lang et al., 2023).

6.2.3 *Registration*

We use the registration of MARS (Ch. 5) including its adaptive surfel resolution selection. A normal distribution $\boldsymbol{e}_{sm} \sim \mathcal{N}\left(\boldsymbol{d}_{sm}, \Sigma_{sm} + \sigma_l^2 I\right)$ models the observation likelihood of scene surfel $s \in \mathcal{S}_l$ for map surfel $m \in \mathcal{M}$ with $\boldsymbol{d}_{sm}$:

$$\boldsymbol{d}_{sm} = T(t)\boldsymbol{\mu}_s - \boldsymbol{\mu}_m, \tag{6.8}$$

$$\Sigma_{sm} = \Sigma_m + R(t)\Sigma_s R(t)^\intercal. \tag{6.9}$$

We use the covariance as is for *non-planar* surfels. For *planar* surfels, we scale the covariance Σ during registration based on their Eigendecomposition:

$$\Sigma = VDV^\intercal, \tag{6.10}$$

$$V = [\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2], \tag{6.11}$$

$$D = \operatorname{diag}\left([\lambda_0, \lambda_1, \lambda_2]^\intercal\right), \tag{6.12}$$

with Eigenvalues $\lambda_0 \leq \lambda_1 \leq \lambda_2 \in \mathbb{R}$ and corresponding Eigenvectors $\boldsymbol{v}_i \in \mathbb{R}^3$.

4 CLINS uses a $\Delta t = 50\,\text{ms}$ for more dynamic datasets, such as hand-held ones, to create two knots per scan.

Replacing D with a scaled matrix $\widetilde{D}$ based on the cell size c_l on the surfels' level:

$$\widetilde{\boldsymbol{\lambda}} = \left[\min(\lambda_0, 0.001), \frac{c_l}{2}, \frac{c_l}{2}\right]^\intercal, \tag{6.13}$$

$$\widetilde{D} = \operatorname{diag}\left(\widetilde{\boldsymbol{\lambda}}\right), \tag{6.14}$$

reinforces the constraint in the normal direction while giving more leeway in the other directions.

In Ch. 5, MARS uses a GMM to represent a scene surfel $s \in \mathcal{S}_l$ observing multiple map surfels $A_s \in \mathcal{M}$ with $\boldsymbol{e}_{sm}$ while taking their similarity into account with weight w_{sm} (Eq. 5.35) and jointly minimizes the data term $\mathcal{L}_{\text{MARS}}$ for L scans:

$$\mathcal{L}_{\text{MARS}}(l) = \sum_{s \in S_l} \sum_{m \in A_s} w_{sm} \boldsymbol{d}_{sm}^\intercal \Sigma_{sm}^{-1} \boldsymbol{d}_{sm}, \tag{6.15}$$

with Levenberg-Marquardt (LM) to update the knots $\mathcal{X}$.

We introduce another prior probability $p(\theta)$ into the GMM's similarity $p(\delta_{sm})$ (Eq. 5.33) based on the dot product between surfel normal $\boldsymbol{n}$ and mean view direction $\boldsymbol{f}$ from sensor origin $\boldsymbol{o}$:

$$p(\theta) \sim \mathcal{N}\left(\arccos\left(\boldsymbol{n}^\intercal \boldsymbol{f}\right), (\pi/8)^2\right). \tag{6.16}$$

This prior reduces the influence of surfaces measured under a steep angle whose normals are less reliable, e.g., when measuring the floor or road far ahead.

To distribute constraints temporally more evenly and to follow the actual sensor trajectory more closely, we linearly subdivide each scan into O time segments with $o \in [0, O-1]$:

$$t_{\text{seg}}(o) = t_{l-1} + \frac{o}{O-1}\left(t_l - t_{l-1}\right) \in (t_{l-1}, t_l]. \tag{6.17}$$

It is a reasonable assumption for a spinning LiDAR that points fused within the same surfel are also close in time. Hence, we group surfels to segments w.r.t. their mean time. Then, we register each segment, evaluated at its respective time $t_{\text{seg}}(o)$, using the data term $\mathcal{L}_{\text{MARS}}$. This acts as a segment-wise undistortion during optimization.

6.2.4 *Optimization*

Complementary to Eq. 6.15, we include constraints $\mathcal{L}_c$ and a marginalization prior $\mathcal{L}_{\text{marg}}$ in the optimization. The constraints $\mathcal{L}_c$ comprise weighted terms for IMU $\mathcal{L}_{\text{IMU}}$ (see Eq. 6.92), zero-acceleration $\mathcal{L}_{\text{z}}$ (see Eq. 6.94), and relative poses $\mathcal{L}_{\boldsymbol{d}_{\Delta T}}$ (see Eq. 6.97):

$$\mathcal{L}_c = w_{\text{IMU}} \mathcal{L}_{\text{IMU}} + w_{\text{z}} \mathcal{L}_{\text{z}} + w_{\text{d}_{\Delta \text{T}}} \mathcal{L}_{\boldsymbol{d}_{\Delta T}}. \tag{6.18}$$

We discuss these terms in greater detail further below in Sec. 6.2.7 and Sec. 6.2.8.

An IMU provides angular velocity $\boldsymbol{\omega}_{\text{m}}$ and linear acceleration $\boldsymbol{a}_{\text{m}}$ readings of the gyroscope and accelerometer, respectively. Fortunately, the spline allows direct evaluation of these quantities with $N \geq 2$ for $\boldsymbol{\omega}(t)$ and $N \geq 3$ for $\boldsymbol{a}(t)$ with zero-mean additive biases $\boldsymbol{b}_{\text{gyr}}(t)$, $\boldsymbol{b}_{\text{acc}}(t)$, and gravity $\boldsymbol{g}_{\text{w}}$:

$$\boldsymbol{\omega}_{\text{m}} = \boldsymbol{\omega}(t) + \boldsymbol{b}_{\text{gyr}}(t), \tag{6.19}$$

$$\boldsymbol{a}_{\text{m}} = R(t)^\intercal \left(\boldsymbol{a}(t) + \boldsymbol{g}_{\text{w}}\right) + \boldsymbol{b}_{\text{acc}}(t). \tag{6.20}$$

We extend our state $\mathbf{x}$ with time-varying accelerometer and gyroscope biases $\boldsymbol{b}_{\text{acc}}(t), \boldsymbol{b}_{\text{gyr}}(t) \in \mathbb{R}^3$. Hence, two non-uniform B-splines (as in Eq. 5.13) represent the biases with a reduced order $N_B \in \{1, 2\}$ and bias knots $B_k = (\mathbb{R}^3 \times \mathbb{R}^3) \in \mathcal{B}$.

Accelerometer biases are quite small ($< |0.1| \mathrm{m\,s^{-2}}$) compared to the earth's gravity ($\approx 9.81\,\mathrm{m\,s^{-2}}$). The earth's gravity $\boldsymbol{g}_{\text{w}}$ commonly dominates $\boldsymbol{a}(t)$ in Eq. 6.20 in our robotic applications ($\leq |2.5| \mathrm{m\,s^{-2}}$ in Tab. 6.4). Yet, gravity $\boldsymbol{g}_{\text{w}}$ acts in a specific direction, which makes its compensation straightforward based on the orientation $R(t)$. Conversely, imperfect initialization of the orientation impairs the trajectory and map accuracy. Thus, we model the gravity direction $\boldsymbol{g}$ on the unit 2-sphere $S^2 = \{\boldsymbol{x} \in \mathbb{R}^3 : \|\boldsymbol{x}\| = 1\}$ (Xu et al., 2022). Then the gravity vector $\boldsymbol{g}_{\text{w}}$ becomes:

$$\boldsymbol{g}_{\text{w}} = 9.81 \cdot \boldsymbol{g} \text{ with } \boldsymbol{g} \in S^2. \tag{6.21}$$

Initially, we set $\boldsymbol{g} = [0, 0, -1]^\intercal$ and $R(t)$ according to a 3-axis magnetometer. In the absence of the magnetometer, acceleration measurements of a stationary sensor allow the observation of roll and pitch of $R(t)$ while the yaw remains unobservable (Geneva et al., 2020) and can be chosen arbitrarily. In practice, averaging over a short period in the beginning is sufficient (Chen et al., 2022; Xu et al., 2022; Chen et al., 2023a).

Overall, we optimize the following cost functions using LM (Sec. 2.2):

$$\underset{\mathcal{X},\mathcal{B},\boldsymbol{g}_{\text{w}}}{\arg\min} \sum_{l=0}^{L-1} \left(\sum_{o=0}^{O-1} \mathcal{L}_{\text{MARS}}(l_o) \right) + \mathcal{L}_c(l) + \mathcal{L}_{\text{marg}}. \tag{6.22}$$

6.2.5 *Marginalization*

Following the standard procedure in LIO (Shan et al., 2020; Xu et al., 2022) and visual-inertial odometry (VIO) (Engel et al., 2018; Geneva et al., 2020; Usenko et al., 2020), we marginalize old state variables leaving the sliding optimization window. These are the oldest knots for pose X_{l-N} and bias B_{l-N_B} within the window $\mathcal{W}_l$.

Reordering the corresponding entries of the normal equations $H, \boldsymbol{b}$ for Eq. 6.22 into marginalizing $\boldsymbol{x}_\beta$ and kept $\boldsymbol{x}_\gamma$ variables:

$$\begin{bmatrix} H_{\beta\beta} H_{\beta\gamma} \\ H_{\gamma\beta} H_{\gamma\gamma} \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_\beta \\ \boldsymbol{x}_\gamma \end{bmatrix} = \begin{bmatrix} \boldsymbol{b}_\beta \\ \boldsymbol{b}_\gamma \end{bmatrix}, \tag{6.23}$$

allows the use of the Schur complement (Leutenegger et al., 2015). Thus, the new marginalization prior (Usenko et al., 2020; Demmel et al., 2021) $H_{\gamma'\gamma'}, \boldsymbol{b}_{\gamma'}$ becomes:

$$H_{\gamma'\gamma'} = H_{\gamma\gamma} - H_{\gamma\beta} H_{\beta\beta}^{-1} H_{\beta\gamma}, \tag{6.24}$$

$$\boldsymbol{b}_{\gamma'} = \boldsymbol{b}_\gamma - H_{\gamma\beta} H_{\beta\beta}^{-1} \boldsymbol{b}_\beta. \tag{6.25}$$

Entries for new variables are initialized to zero in the next iteration.

For our initialization, we perform a separate marginalization of all bias knots $\mathcal{B}$ and the gravity $\boldsymbol{g}_{\text{w}}$ except for the remaining poses.

6.2.6 *Initialization*

In contrast to the previous Ch. 5, we use associated surfels directly for the older scans. For a new scan, we incorporate its constraints $\mathcal{L}_c$ and none of its surfels. The

linearization point is fixed for each knot—except for the new ones. Thus, we employ First-Estimate Jacobians (Huang et al., 2008) and fix surfel associations beforehand. When all previous knots fit well from earlier registrations, any modification likely increases their respective errors. With all knots but one quasi-fixed, the spline evaluations in Eq. 5.15 and Eq. 5.13 become linear interpolations for timestep t. As a result, only the newest knot may change to minimize the constraints' errors. In some cases, we found oscillations occurring at twice the frequency of placed knots. Hence, we increase the weight of the IMU term by a factor of 10 for the newest scan to prevent early local minima and optimize:

$$X^{\text{init}} = \arg\min_{\mathcal{X}} \sum_{l=0}^{L-1} \mathcal{L}_c(l) + \sum_{l=0}^{L-2} \left(\sum_{o=0}^{O-1} \mathcal{L}_{\text{MARS}}(l_o) \right) + \mathcal{L}_{\text{marg}}. \tag{6.26}$$

For biases, new knots are set to the previous estimate.

6.2.7 *Inertial Spline Constraints*

Constraints on the angular velocity $\boldsymbol{\omega}(t)$ and linear acceleration $\boldsymbol{a}(t)$ arise naturally from the gyroscope (Eq. 6.19) and accelerometer measurements (Eq. 6.20):

$$\boldsymbol{d}_{\text{gyr}} = \boldsymbol{\omega}(t) - (\boldsymbol{\omega}_{\text{m}} - \boldsymbol{b}_{\text{gyr}}(t)), \tag{6.27}$$

$$\boldsymbol{d}_{\text{acc}} = R(t)^{\intercal} (\boldsymbol{a}(t) + \boldsymbol{g}_{\text{w}}) - (\boldsymbol{a}_{\text{m}} - \boldsymbol{b}_{\text{acc}}(t)). \tag{6.28}$$

However, IMUs' high sampling frequencies lead to many additional terms in Eq. 6.22 which increase the computational load for optimization. Preintegration (Leutenegger et al., 2015; Geneva et al., 2020; Shan et al., 2020; Usenko et al., 2020) is a convenient method to combine multiple IMU measurements. Here, a single preintegrated pseudo-measurement has mean Δ_{pre} and covariance $\Sigma^{\Delta_{\text{pre}}}$. The integration starts at time t_{m_i} with linearized biases $\boldsymbol{b}_{\text{acc},i}, \boldsymbol{b}_{\text{gyr},i}$. We use the subscript m_j to emphasize the last integrated measurement with time t_{m_j}:

$$\Delta_{\text{pre}} = \left(\Delta\boldsymbol{p}_{\text{m}_j}, \Delta R_{\text{m}_j}, \Delta\boldsymbol{v}_{\text{m}_j} \right) \in \mathbb{R}^3 \times SO(3) \times \mathbb{R}^3, \tag{6.29}$$

$$\Sigma^{\Delta_{\text{pre}}} \in \mathbb{R}^{9\times 9}. \tag{6.30}$$

After bias correction of a new IMU measurement $(\boldsymbol{\omega}, \boldsymbol{a})_{\text{m}_{j+1}}$ at $t_{\text{m}_{j+1}}$:

$$\overline{\boldsymbol{\omega}}_{\text{m}_{j+1}} = \boldsymbol{\omega}_{\text{m}_{j+1}} - \boldsymbol{b}_{\text{gyr},i}, \tag{6.31}$$

$$\overline{\boldsymbol{a}}_{\text{m}_{j+1}} = \boldsymbol{a}_{\text{m}_{j+1}} - \boldsymbol{b}_{\text{acc},i}, \tag{6.32}$$

the integration updates the mean Δ_{pre} using:

$$\Delta t_{\text{m}_j} = (t_{\text{m}_{j+1}} - t_{\text{m}_j}), \tag{6.33}$$

$$R_{\text{w}} = \Delta R_{\text{m}_j} \cdot \text{Exp}\left(\frac{\Delta t_{\text{m}_j}}{2} \cdot \overline{\boldsymbol{\omega}}_{\text{m}_{j+1}} \right), \tag{6.34}$$

as follows (Usenko et al., 2020):

$$\Delta\boldsymbol{p}_{\text{m}_{j+1}} = \Delta\boldsymbol{p}_{\text{m}_{j+1}} + \Delta t_{\text{m}_j} \cdot \Delta\boldsymbol{v}_{\text{m}_j} + \frac{\Delta t_{\text{m}_j}^2}{2} \cdot R_{\text{w}} \overline{\boldsymbol{a}}_{\text{m}_{j+1}}, \tag{6.35}$$

$$\Delta\boldsymbol{v}_{\text{m}_{j+1}} = \Delta\boldsymbol{v}_{\text{m}_j} + \Delta t_{\text{m}_j} \cdot R_{\text{w}} \overline{\boldsymbol{a}}_{\text{m}_{j+1}}, \tag{6.36}$$

$$\Delta R_{\text{m}_{j+1}} = \Delta R_{\text{m}_j} \cdot \text{Exp}\left(\Delta t_{\text{m}_j} \cdot \overline{\boldsymbol{\omega}}_{\text{m}_{j+1}} \right). \tag{6.37}$$

The covariance $\Sigma^{\Delta_{\text{pre}}}$ is propagated accordingly with fixed measurement noises $\Sigma_{\text{acc}}, \Sigma_{\text{gyr}}$:

$$\begin{aligned}\Sigma^{\Delta_{\text{pre}_{j+1}}} &= \left(\boldsymbol{J}^{\Delta_{\text{pre}_{j+1}}}_{\Delta_{\text{pre}_j}}\right) \Sigma^{\Delta_{\text{pre}_j}} \left(\boldsymbol{J}^{\Delta_{\text{pre}_{j+1}}}_{\Delta_{\text{pre}_j}}\right)^{\mathsf{T}} \\ &+ \left(\boldsymbol{J}^{\Delta_{\text{pre}_{j+1}}}_{\overline{\boldsymbol{a}}_{j+1}}\right) \Sigma_{\text{acc}} \left(\boldsymbol{J}^{\Delta_{\text{pre}_{j+1}}}_{\overline{\boldsymbol{a}}_{j+1}}\right)^{\mathsf{T}} \\ &+ \left(\boldsymbol{J}^{\Delta_{\text{pre}_{j+1}}}_{\overline{\boldsymbol{\omega}}_{j+1}}\right) \Sigma_{\text{gyr}} \left(\boldsymbol{J}^{\Delta_{\text{pre}_{j+1}}}_{\overline{\boldsymbol{\omega}}_{j+1}}\right)^{\mathsf{T}}.\end{aligned} \tag{6.38}$$

The Jacobians required for propagation are given in App. B.1.

For simplicity, we use the shorthand f_i (resp. f_j) for a B-spline function $f(t)$ evaluated at t_{m_i} (resp. t_{m_j}) and drop the subscript for preintegrated variables from t_{m_i} to t_{m_j}. For now, we ignore the biases and set $\Delta_{\overline{\text{pre}}} = \Delta_{\text{pre}}$ to define the errors for $\Delta_{\overline{\text{pre}}} = \left(\Delta\overline{\boldsymbol{p}}, \Delta\overline{R}, \Delta\overline{\boldsymbol{v}}\right)$:

$$\Delta t_{\text{m}} = t_{\text{m}_j} - t_{\text{m}_i}, \tag{6.39}$$

$$\boldsymbol{d}_{\Delta\widetilde{\boldsymbol{p}}} = R_i^{\mathsf{T}} \overbrace{\left(\boldsymbol{p}_j - \boldsymbol{p}_i - \boldsymbol{v}_i \Delta t_{\text{m}} - \boldsymbol{g}_{\text{w}} \frac{\Delta t_{\text{m}}^2}{2}\right)}^{\Delta\widetilde{\boldsymbol{p}}} - \Delta\overline{\boldsymbol{p}}, \tag{6.40}$$

$$\boldsymbol{d}_{\Delta\widetilde{R}} = \text{Log}\left(\Delta\overline{R} R_j^{\mathsf{T}} R_i\right), \tag{6.41}$$

$$\boldsymbol{d}_{\Delta\widetilde{\boldsymbol{v}}} = R_i^{\mathsf{T}} \left(\boldsymbol{v}_j - \boldsymbol{v}_i - \boldsymbol{g}_{\text{w}} \Delta t_{\text{m}}\right) - \Delta\overline{\boldsymbol{v}}. \tag{6.42}$$

While the Jacobians depending on a single time t are given[5] by Sommer et al. (2020), we need to derive the right Jacobians $\boldsymbol{J}_X^{f(X)}$ w.r.t. knots X_{k_i} and X_{k_j} for the above *relative* residuals $\boldsymbol{d}_{\Delta_{\text{pre}}} = [\boldsymbol{d}^{\mathsf{T}}_{\Delta\widetilde{\boldsymbol{p}}}, \boldsymbol{d}^{\mathsf{T}}_{\Delta\widetilde{R}}, \boldsymbol{d}^{\mathsf{T}}_{\Delta\widetilde{\boldsymbol{v}}}]^{\mathsf{T}}$.

Sommer et al. (2020) defined the composite manifold[6] for a spline knot $X \in SO(3) \times \mathbb{R}^3$ with composition ($\circ$), a left-increment update ($\oplus$) and a right-decrement downdate ($\ominus$):

$$\text{Exp}(\boldsymbol{\tau}) = \exp\left(\boldsymbol{\tau}^{\wedge}\right) = X \in SO(3) \times \mathbb{R}^3, \tag{6.43}$$

$$\text{Log}(X) = \log\left(X\right)^{\vee} = \boldsymbol{\tau} \in \mathbb{R}^6, \tag{6.44}$$

$$Y = X \oplus \boldsymbol{\tau} = \text{Exp}(\boldsymbol{\tau}) \circ X, \text{ with } \boldsymbol{\tau} \in \mathbb{R}^6 \tag{6.45}$$

$$\boldsymbol{\tau} = Y \ominus X = \text{Log}(X^{-1} \circ Y), \tag{6.46}$$

$$\tag{6.47}$$

as well as a right Jacobian $\boldsymbol{J}_X^{f(X)} \in \mathbb{R}^{6\times 6}$:

$$\boldsymbol{J}_X^{f(X)}(\boldsymbol{\tau}) = \lim_{\boldsymbol{\tau}\to 0} \frac{f(X \oplus \boldsymbol{\tau}) \ominus f(X)}{\boldsymbol{\tau}}, \tag{6.48}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\text{Log}(f(X)^{-1} \circ f(X \oplus \boldsymbol{\tau}))}{\boldsymbol{\tau}}, \tag{6.49}$$

$$= \frac{\partial \text{Log}\left(f(X)^{-1} \circ f(X \oplus \boldsymbol{\tau})\right)}{\partial \boldsymbol{\tau}}\bigg|_{\boldsymbol{\tau}=\mathbf{0}}. \tag{6.50}$$

5 Sommer et al. (2020) provide the right Jacobians for orientation $R(t)$, angular rate $\boldsymbol{\omega}(t)$ and angular acceleration $\boldsymbol{\alpha}(t)$, as well as position $\boldsymbol{p}(t)$, velocity $\boldsymbol{v}(t)$ and linear acceleration $\boldsymbol{a}(t)$.

6 The definition differs from Solà et al. (2018) who use right-$\oplus$ and right-$\ominus$, thus, leading to different Jacobians derived for $SE(3)$ and $SO(3) \times \mathbb{R}^3$.

The adjoint Adj_X is a linear map defined as:

$$\mathrm{Adj}_X = (X\boldsymbol{\tau}^\wedge X^{-1})^\vee. \tag{6.51}$$

The following identities[7] are helpful for further derivations:

$$\exp(X\boldsymbol{\tau}^\wedge X^{-1}) = X\exp(\boldsymbol{\tau}^\wedge)X^{-1}, \tag{6.52}$$

$$\exp(\boldsymbol{\tau}^\wedge)^{-1} = \exp(-\boldsymbol{\tau}^\wedge) \underset{\boldsymbol{\tau}\to\mathbf{0}}{=} I - \boldsymbol{\tau}^\wedge. \tag{6.53}$$

A relative B-spline function $\mathrm{g} = f_X(t_i, t_j)$ depends on two timestamps t_i, t_j. Due to the local support of a B-spline, a knot X_k has non-zero weight for none, one or both timestamps. Hence, we denote the set of N knots with non-zero weight at time t as $\mathcal{X}(t)$. Obviously, the Jacobian $\boldsymbol{J}^{\mathrm{g}}_{X_k}$ of g w.r.t. X_k is zero for $X_k \notin \{\mathcal{X}(t_i) \cup \mathcal{X}(t_j)\}$ since the weight of X_k is zero.

The decoupled logarithm Log_D of $SO(3) \times \mathbb{R}^3$ allows us to treat the Jacobians for the knot's rotation R_k and position $\boldsymbol{p}_k$ independently: $\boldsymbol{J}^{\mathrm{g}}_{X_k} = \left[\boldsymbol{J}^{\mathrm{g}}_{\boldsymbol{p}_k}, \boldsymbol{J}^{\mathrm{g}}_{R_k}\right]$. Thus, we begin with the derivation for the rotation:

$$\Delta\widetilde{R} = \Delta\overline{R}R_j^\intercal R_i, \tag{6.54}$$

in Eq. 6.41[8]. First, we assume no overlap between $\mathcal{X}(t_i)$ and $\mathcal{X}(t_j)$ and consider both cases separately, starting with $R_k \in SO(3)$ evaluated at t_i:

$$\mathbf{J}^{\Delta\widetilde{R}}_{R_k,t_i} = \lim_{\boldsymbol{\tau}\to 0} \frac{\left(\Delta\widetilde{R}(X_k \oplus \boldsymbol{\tau})\right) \ominus \left(\Delta\widetilde{R}(X_k)\right)}{\boldsymbol{\tau}}, \tag{6.55}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\left(\Delta\overline{R}R_j^{-1}R_i(X_k \oplus \boldsymbol{\tau})\right) \ominus \left(\Delta\overline{R}R_j^{-1}R_i(X_k)\right)}{\boldsymbol{\tau}}, \tag{6.56}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(\left(\Delta\overline{R}R_j^{-1}R_i\right)^{-1}\Delta\overline{R}R_j^{-1}\mathrm{Exp}(\boldsymbol{\tau})R_i\right)}{\boldsymbol{\tau}}, \tag{6.57}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(R_i^\intercal R_j \Delta\overline{R}^\intercal \Delta\overline{R}R_j^\intercal \mathrm{Exp}(\boldsymbol{\tau})R_i\right)}{\boldsymbol{\tau}}, \tag{6.58}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(R_i^\intercal \mathrm{Exp}(\boldsymbol{\tau})R_i\right)}{\boldsymbol{\tau}} = \lim_{\boldsymbol{\tau}\to 0} \frac{(R_i^\intercal \boldsymbol{\tau}^\wedge R_i)^\vee}{\boldsymbol{\tau}}, \tag{6.59}$$

$$= \mathrm{Adj}_{R_i^\intercal}\mathbf{J}^{R_i}_{R_k} = R_i^\intercal \boldsymbol{J}^{R_i}_{R_k}. \tag{6.60}$$

Here, $R^{-1} = R^\intercal$ allows simplification in Eq. 6.58 and cancels out many terms. Equation 6.59 uses the identity in Eq. 6.52 and the definition of $\mathrm{Log}(X)$ in Eq. 6.44. The equalities in Eq. 6.60 stem from the definition of the adjoint (Eq. 6.51) and $\mathrm{Adj}_R = R$ (see Eq. (139) in Solà et al. (2018)), whereas Sommer et al. (2020) provide $\boldsymbol{J}^{R_i}_{R_k}$.

7 Equation 6.52 is given as (20) in Solà et al. (2018).

8 The preintegration of Basalt uses a right-increment (Eq. 6.37) and thus $R_j \approx R_i\Delta\overline{R}$ follows, which leads to Eq. 6.41.

The derivation follows similarly for t_j:

$$\boldsymbol{J}^{\Delta\widetilde{R}}_{R_k,t_j} = \lim_{\boldsymbol{\tau}\to 0} \frac{\left(\Delta\widetilde{R}(X_k \oplus \boldsymbol{\tau})\right) \ominus \left(\Delta\widetilde{R}(X_k)\right)}{\boldsymbol{\tau}}, \tag{6.61}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\left(\Delta\overline{R}\left(R_j(X_k \oplus \boldsymbol{\tau})\right)^{-1} R_i\right) \ominus \left(\Delta\overline{R}\left(R_j(X_k)\right)^{-1} R_i\right)}{\boldsymbol{\tau}}, \tag{6.62}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(\Delta\widetilde{R}(X_k)^{-1}\Delta\overline{R}\left(\mathrm{Exp}(\boldsymbol{\tau})R_j(X_k)\right)^{-1} R_i\right)}{\boldsymbol{\tau}}, \tag{6.63}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(\left(\Delta\overline{R}R_j^\intercal R_i\right)^{-1} \Delta\overline{R}R_j^{-1}\mathrm{Exp}(-\boldsymbol{\tau})R_i\right)}{\boldsymbol{\tau}}, \tag{6.64}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(R_i^\intercal R_j \Delta\overline{R}^\intercal \Delta\overline{R}R_j^\intercal \mathrm{Exp}(-\boldsymbol{\tau})R_i\right)}{\boldsymbol{\tau}}, \tag{6.65}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{\mathrm{Log}\left(R_i^\intercal \mathrm{Exp}(-\boldsymbol{\tau})R_i\right)}{\boldsymbol{\tau}} = \lim_{\boldsymbol{\tau}\to 0} \frac{\left(R_i^\intercal(-\boldsymbol{\tau}^\wedge)R_i\right)^\vee}{\boldsymbol{\tau}}, \tag{6.66}$$

$$= -\mathrm{Adj}_{R_i^\intercal}\mathbf{J}^{R_j}_{R_k} = -R_i^\intercal \mathbf{J}^{R_j}_{R_k}. \tag{6.67}$$

Finally, we have for $X_k \in \{\mathcal{X}(t_i) \cap \mathcal{X}(t_j)\}$ with Eq. 6.60 and Eq. 6.67 according to the product rule:

$$\boldsymbol{J}^{\Delta\widetilde{R}}_{R_k} = R_i^\intercal \boldsymbol{J}^{R_i}_{R_k} - R_i^\intercal \boldsymbol{J}^{R_j}_{R_k}. \tag{6.68}$$

This is coincidently the general form for $\boldsymbol{J}^{\Delta\widetilde{R}}_{R_k}$ whereas the special cases (Eq. 6.60 and Eq. 6.67) arise if $\mathcal{X}(t) = \emptyset$ and thus $\boldsymbol{J}^{R(t)}_{R_k} = 0_{3\times 3}$.

Evidently, the Jacobian of $\Delta\widetilde{R}$ w.r.t. the knot's position $\boldsymbol{J}^{\Delta\widetilde{R}}_{\boldsymbol{p}_k}$ is zero and independent of t, which leads with the inverse right Jacobian $\boldsymbol{J}^{\mathrm{Log}(R)}_{R} = J_r^{-1}(\mathrm{Log}(R))$ for $SO(3)$ (Sommer et al., 2020) to:

$$\boldsymbol{J}^{\boldsymbol{d}_{\Delta\widetilde{R}}}_{X_k} = \left[0_{3\times 3}, J_r^{-1}\left(\boldsymbol{d}_{\Delta\widetilde{R}}\right)\boldsymbol{J}^{\Delta\widetilde{R}}_{R_k}\right]. \tag{6.69}$$

The errors for preintegrated position (Eq. 6.40) and velocity (Eq. 6.42) share a common form $\Delta\mathrm{h}$ with $f_{\boldsymbol{p}_k}(t)$ depending on the knot position $\boldsymbol{p}_k$ and some vector $\boldsymbol{e} \in \mathbb{R}^3$:

$$\Delta\mathrm{h} = R_i^\intercal \overbrace{\left(f_{\boldsymbol{p}_k}(t_j) - f_{\boldsymbol{p}_k}(t_i) + \boldsymbol{e}\right)}^{\mathbf{d}}. \tag{6.70}$$

For $\boldsymbol{J}^{\Delta\mathrm{h}}_{R_k}$, we use the following identities[9]:

$$[\boldsymbol{w}]_\times \boldsymbol{v} = -[\boldsymbol{v}]_\times \boldsymbol{w}, \tag{6.71}$$

$$(R\boldsymbol{w})^\wedge R = R\boldsymbol{w}^\wedge. \tag{6.72}$$

9 Equation 6.72 for $SO(3)$ is a direct result from Eq. 6.51 and Eq. 6.71.

Since the target domain of $\Delta\mathrm{h}$ is $\mathbb{R}^3$ the $\ominus$ becomes a subtraction:

$$\boldsymbol{J}^{\Delta\mathrm{h}}_{R_k,t_i} = \lim_{\boldsymbol{\tau}\to 0} \frac{(\Delta\mathrm{h}(X_k \oplus \boldsymbol{\tau})) \ominus (\Delta\mathrm{h}(X_k))}{\boldsymbol{\tau}}, \tag{6.73}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{(\mathrm{Exp}(\boldsymbol{\tau})R_i)^{\mathsf{T}} (f_{\boldsymbol{p}_k}(t_j) - f_{\boldsymbol{p}_k}(t_i) + \boldsymbol{c}) - \Delta\mathrm{h}}{\boldsymbol{\tau}}, \tag{6.74}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{R_i^{\mathsf{T}}(I - \boldsymbol{\tau}^\wedge)\mathbf{d} - \Delta\mathrm{h}}{\boldsymbol{\tau}} = \lim_{\boldsymbol{\tau}\to 0} \frac{-R_i^{\mathsf{T}}\boldsymbol{\tau}^\wedge\mathbf{d}}{\boldsymbol{\tau}}, \tag{6.75}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{-(R_i^{\mathsf{T}}\boldsymbol{\tau})^\wedge R_i^{\mathsf{T}}\mathbf{d}}{\boldsymbol{\tau}} = \lim_{\boldsymbol{\tau}\to 0} \frac{-[R_i^{\mathsf{T}}\boldsymbol{\tau}]_\times \Delta\mathrm{h}}{\boldsymbol{\tau}}, \tag{6.76}$$

$$= \lim_{\boldsymbol{\tau}\to 0} \frac{[\Delta\mathrm{h}]_\times R_i^{\mathsf{T}}\boldsymbol{\tau}}{\boldsymbol{\tau}} = [\Delta\mathrm{h}]_\times R_i^{\mathsf{T}} \boldsymbol{J}^{R_i}_{R_k}, \tag{6.77}$$

$$\boldsymbol{J}^{\Delta\mathrm{h}}_{R_k,t_j} = 0_{3\times 3}. \tag{6.78}$$

Here, Eq. 6.75 uses the third equality of Eq. 6.53 while Equation 6.72 expands Eq. 6.76 to simplify it with Eq. 6.71 subsequently.

The derivation for $\Delta\mathrm{h}$ w.r.t. $f_{\boldsymbol{p}_k}(t)$ is straightforward due to $f_{\boldsymbol{p}_k}(t) \in \mathbb{R}^3$ and the target domain $\mathbb{R}^3$ and thus omitted for brevity:

$$\boldsymbol{J}^{\Delta\mathrm{h}}_{f_{\boldsymbol{p}_k}} = -R_i^{\mathsf{T}} \boldsymbol{J}^{f_{\boldsymbol{p}_k}(t_i)}_{\boldsymbol{p}_k} + R_i^{\mathsf{T}} \boldsymbol{J}^{f_{\boldsymbol{p}_k}(t_j)}_{\boldsymbol{p}_k}. \tag{6.79}$$

The Jacobian of $\Delta\mathrm{h}$ w.r.t. X_k is:

$$\boldsymbol{J}^{\Delta\mathrm{h}}_{X_k} = \left[R_i^{\mathsf{T}} \left(\boldsymbol{J}^{f_{\boldsymbol{p}_k}(t_j)}_{\boldsymbol{p}_k} - \boldsymbol{J}^{f_{\boldsymbol{p}_k}(t_i)}_{\boldsymbol{p}_k} \right), \quad [\Delta\mathrm{h}]_\times R_i^{\mathsf{T}} \boldsymbol{J}^{R_i}_{R_k} \right]. \tag{6.80}$$

For $\boldsymbol{d}_{\Delta\widetilde{\boldsymbol{v}}}$ (Eq. 6.42), its Jacobian $\boldsymbol{J}^{\boldsymbol{d}_{\Delta\widetilde{\boldsymbol{v}}}}_{X_k}$ directly follows from Eq. 6.80 with $f_{\boldsymbol{p}_k}(t) = \boldsymbol{v}(t)$. Similarly, we obtain $\boldsymbol{J}^{\boldsymbol{d}_{\Delta\widetilde{\boldsymbol{p}}}}_{X_k}$ for Eq. 6.40:

$$\boldsymbol{J}^{\boldsymbol{d}_{\Delta\widetilde{\boldsymbol{p}}}}_{X_k} = \left[R_i^{\mathsf{T}} \left(\boldsymbol{J}^{\boldsymbol{p}_j}_{\boldsymbol{p}_k} - \boldsymbol{J}^{\boldsymbol{p}_i}_{\boldsymbol{p}_k} - \Delta t_\mathrm{m} \boldsymbol{J}^{\boldsymbol{v}_i}_{\boldsymbol{p}_k} \right), \quad [\Delta\widetilde{\boldsymbol{p}}]_\times R_i^{\mathsf{T}} \boldsymbol{J}^{R_i}_{R_k} \right]. \tag{6.81}$$

For gravity $\boldsymbol{g}_\mathrm{w}$, the Jacobians are straightforward with $\boldsymbol{J}^{\boldsymbol{g}_\mathrm{w}}_{\boldsymbol{g}}$ due to $\boldsymbol{g} \in S^2$:

$$\boldsymbol{J}^{\boldsymbol{d}_{\Delta\mathrm{pre}}}_{\boldsymbol{g}_\mathrm{w}} = \left[\left(-R_i^{\mathsf{T}} \frac{\Delta t_\mathrm{m}^2}{2} \right)^{\mathsf{T}}, 0_{3\times 3}, (-R_i^{\mathsf{T}} \Delta t_\mathrm{m})^{\mathsf{T}} \right]^{\mathsf{T}} \boldsymbol{J}^{\boldsymbol{g}_\mathrm{w}}_{\boldsymbol{g}}. \tag{6.82}$$

To update our biases, we must consider the linearization of $\boldsymbol{b}_{\mathrm{acc},i}$ and $\boldsymbol{b}_{\mathrm{gyr},i}$. Hence, this modifies $\Delta_{\overline{\mathrm{pre}}}$ (Usenko et al., 2020):

$$\boldsymbol{d}_{b_{acc}} = \boldsymbol{b}_\mathrm{acc}(t_{\mathrm{m}_i}) - \boldsymbol{b}_{\mathrm{acc},i}, \tag{6.83}$$

$$\boldsymbol{d}_{b_{gyr}} = \boldsymbol{b}_\mathrm{gyr}(t_{\mathrm{m}_i}) - \boldsymbol{b}_{\mathrm{gyr},i}, \tag{6.84}$$

$$\Delta\overline{\boldsymbol{p}} = \Delta\boldsymbol{p} + \boldsymbol{J}^{\Delta\boldsymbol{p}}_{\boldsymbol{b}_{\mathrm{acc},i}} \boldsymbol{d}_{b_{acc}} + \boldsymbol{J}^{\Delta\boldsymbol{p}}_{\boldsymbol{b}_{\mathrm{gyr},i}} \boldsymbol{d}_{b_{gyr}}, \tag{6.85}$$

$$\Delta\overline{\boldsymbol{v}} = \Delta\boldsymbol{v} + \boldsymbol{J}^{\Delta\boldsymbol{v}}_{\boldsymbol{b}_{\mathrm{acc},i}} \boldsymbol{d}_{b_{acc}} + \boldsymbol{J}^{\Delta\boldsymbol{v}}_{\boldsymbol{b}_{\mathrm{gyr},i}} \boldsymbol{d}_{b_{gyr}}, \tag{6.86}$$

$$\Delta\overline{R} = \mathrm{Exp}\left(\boldsymbol{J}^{\Delta R}_{\boldsymbol{b}_{\mathrm{gyr},i}} \boldsymbol{d}_{b_{gyr}} \right) \Delta R. \tag{6.87}$$

Fortunately, this does not change the general form of the above Jacobians w.r.t. the knot X_k. With these, we obtain the Jacobians w.r.t. bias knots $\boldsymbol{b}_{\text{acc},k}, \boldsymbol{b}_{\text{gyr},k}$:

$$\boldsymbol{J}^{\boldsymbol{d}_{\Delta\text{pre}}}_{\boldsymbol{b}_{\text{acc},k}} = -\boldsymbol{J}^{\Delta\text{pre}}_{\boldsymbol{b}_{\text{acc},i}} \boldsymbol{J}^{\boldsymbol{b}_{\text{acc}}(t_{\text{m}_i})}_{\boldsymbol{b}_{\text{acc},k}}, \tag{6.88}$$

$$\boldsymbol{J}^{\boldsymbol{d}_{\Delta\text{pre}}}_{\boldsymbol{b}_{\text{gyr},k}} = -\begin{bmatrix} I_{3\times3} & 0_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & J_r^{-1}(\Delta\widetilde{R})\Delta\widetilde{R}^\intercal & 0_{3\times3} \\ 0_{3\times3} & 0_{3\times3} & I_{3\times3} \end{bmatrix} \boldsymbol{J}^{\Delta\text{pre}}_{\boldsymbol{b}_{\text{gyr},i}} \boldsymbol{J}^{\boldsymbol{b}_{\text{gyr}}(t_{\text{m}_i})}_{\boldsymbol{b}_{\text{gyr},k}}. \tag{6.89}$$

Empirically, we found that solely relying on preintegration may lead to slippage in the estimation (esp. in rotation) due to the change from absolute IMU measurements $(\boldsymbol{a}_\text{m}, \boldsymbol{\omega}_\text{m})$ to purely relative ones. In case of high rotational velocities, e.g., $\geq 120\,^\circ\,\text{s}^{-1}$, we switch to raw measurements. Otherwise, we fuse all IMU measurements between two scans except for the last one, which allows us to define the IMU error $\mathcal{L}_\text{IMU}$:

$$\mathcal{L}_\text{raw} = \boldsymbol{d}_\text{acc}^\intercal \Sigma_\text{acc}^{-1} \boldsymbol{d}_\text{acc} + \boldsymbol{d}_\text{gyr}^\intercal \Sigma_\text{gyr}^{-1} \boldsymbol{d}_\text{gyr}, \tag{6.90}$$

$$\mathcal{L}_\text{pre} = \boldsymbol{d}_{\Delta\text{pre}}^\intercal \left(\Sigma^{\Delta\text{pre}}\right)^{-1} \boldsymbol{d}_{\Delta\text{pre}}, \tag{6.91}$$

$$\mathcal{L}_\text{IMU} = w_\text{IMU} \left(\mathcal{L}_\text{pre}(t_{\text{m}_i}, \ldots, t_{\text{m}_{j-1}}) + \mathcal{L}_\text{raw}(t_{\text{m}_j})\right). \tag{6.92}$$

In the absence of further sensor input, we regularize the spline for $N \geq 3$ with zero-acceleration soft-constraints on linear $\boldsymbol{a}_\text{z}(t)$ and angular acceleration $\boldsymbol{\alpha}_\text{z}(t)$ in IMU frame with covariance Σ_z and weight w_z:

$$\boldsymbol{d}_\text{z}(t) = [\boldsymbol{a}_\text{z}(t)^\intercal, \boldsymbol{\alpha}_\text{z}(t)^\intercal]^\intercal, \tag{6.93}$$

$$\mathcal{L}_\text{z}(i) = w_\text{z} \sum_{o=0}^{O-1} \boldsymbol{d}_\text{z}^\intercal(t_\text{seg}(o)) \Sigma_\text{z}^{-1} \boldsymbol{d}_\text{z}(t_\text{seg}(o)). \tag{6.94}$$

This promotes uniform motion towards a constant velocity model without strictly enforcing it.

6.2.8 *Relative Motion Constraints*

In some situations, a robot supplies further motion estimates, e.g., from the wheel or joint encoders or visual odometry (VO). Hence, we promote similarity between robot odometry poses $\Delta T_\text{m} = T_{\text{m}_j}^{-1} T_{\text{m}_i}$ and the spline $\Delta T = T_j^{-1} T_i$ using a relative pose error $\boldsymbol{d}_{\Delta T}$ with weight matrix $W_{\boldsymbol{d}_{\Delta T}}$:

$$\Delta\boldsymbol{p} = R_i^\intercal \left(\boldsymbol{p}_j - \boldsymbol{p}_i\right), \tag{6.95}$$

$$\boldsymbol{d}_{\Delta T} = \begin{bmatrix} \Delta R_\text{m}^\intercal \left(\Delta\boldsymbol{p} - \Delta\boldsymbol{p}_\text{m}\right) \\ \text{Log}\left(\Delta R_\text{m}^\intercal R_j^\intercal R_i\right) \end{bmatrix}, \tag{6.96}$$

$$\mathcal{L}_{\boldsymbol{d}_{\Delta T}} = w_{\Delta T} \boldsymbol{d}_{\Delta T}^\intercal W_{\boldsymbol{d}_{\Delta T}} \boldsymbol{d}_{\Delta T}. \tag{6.97}$$

From Eq. 6.80, we get $\boldsymbol{J}^{\Delta\boldsymbol{p}}_{X_k}$ and obtain $\boldsymbol{J}^{\boldsymbol{d}_{\Delta T}}_{X_k}$ using Eq. 6.69 with $\Delta\overline{R} = \Delta R_\text{m}^\intercal$:

$$\boldsymbol{J}^{\boldsymbol{d}_{\Delta T}}_{X_k} = \begin{bmatrix} \Delta R_\text{m}^\intercal \boldsymbol{J}^{\Delta\boldsymbol{p}}_{X_k} \\ \boldsymbol{J}^{\boldsymbol{d}_{\Delta\widetilde{R}}}_{X_k} \end{bmatrix}. \tag{6.98}$$

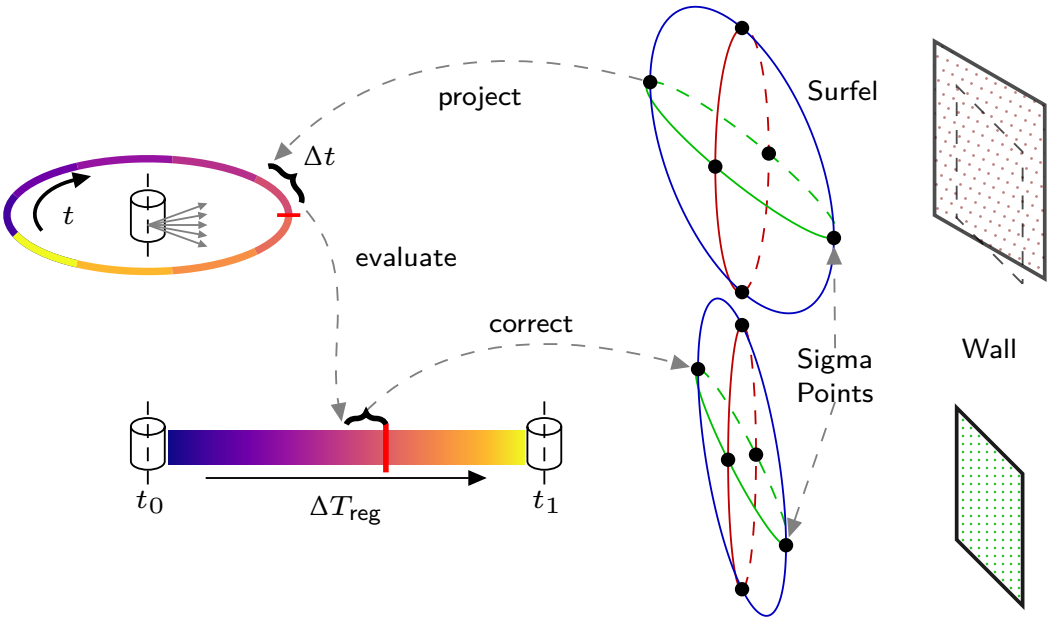

Figure 6.6: Surface element (surfel) motion compensation via unscented transform (UT): sensor motion ΔT_{reg} from t_0 to t_1, e.g., towards a wall, distorts and skews scans and surfels. Sigma points $\mathcal{Y}_\Sigma$ of the surfel covariance Σ_s are projected into the LiDAR to obtain the sigma points' time $t_{\boldsymbol{y}}$. Spline evaluation allows motion correction towards a segments' reference time t_{seg} (red) before surfels are re-fused.

Without a covariance estimate for ΔT, $W_{\boldsymbol{d}_{\Delta T}}$ provides an opportunity to improve the method's resilience. The limited field-of-view (FoV), range of LiDARs, and the environment's geometry can lead to an uneven constraint distribution for all degrees of freedom (DoFs). An example are tunnel-like structures when there are mostly measurements of the walls, floor and ceiling. Here, constraints for the translational DoF along the tunnel are likely underrepresented or dominated by noise (Zhang et al., 2016; Tuna et al., 2024).

Nashed et al. (2021) selectively constrain translation along the Eigenvectors $\boldsymbol{v}_i$ of the normals' covariance $C_{\boldsymbol{n}}$ if the condition number $\kappa_i = \lambda_{max}/\lambda_i$ (Sec. 2.3) is above $\tau_\kappa = 10$. We empirically found an adaptive scaling w.r.t. the data term $\mathcal{L}_{\text{MARS}}$ and the inclusion of the orientation direction to be worthwhile:

$$C_l = \frac{1}{|S_l|} \sum_{s \in S_l} \boldsymbol{h}_s^\intercal \boldsymbol{h}_s \text{ with } \boldsymbol{h}_s = [\boldsymbol{n}_s, \boldsymbol{\mu}_s \times \boldsymbol{n}_s]^\intercal . \tag{6.99}$$

Hence, we scale the Eigenvectors $\boldsymbol{v}_i$ of C_l with κ_i and use the weight matrix[10]:

$$W_l = \sum_i \boldsymbol{v}_i \kappa_i \boldsymbol{v}_i^\intercal, \tag{6.100}$$

for the relative pose error $\mathcal{L}_{\boldsymbol{d}_{\Delta T}}$ within $(t_{l-L}, t_l]$. A similar W_l is obtained for a position-only prior $\mathcal{L}_{\Delta \boldsymbol{p}}$ (resp. orientation-only $\mathcal{L}_{\Delta R}$) from the 3×3 top-left (resp. bottom-right) block matrix of C_l.

10 Since C_l is real and symmetric positive semidefinite (s.p.d.), inverting its Eigenvalues λ_i leads to the inverse $C_l^{-1} = V \Lambda^{-1} V^\intercal$. Thus, the scaling directly provides the information matrix W.

6.2.9 *Unscented Transform for Motion Compensation*

The spline allows us to compensate ego-motion towards a reference time t_{ref}, given the points' capture time $t(\boldsymbol{p})$:

$$\bar{\boldsymbol{p}} = T_{\mathcal{X}}\left(t_{\text{ref}}\right)^{-1} T_{\mathcal{X}}\left(t(\boldsymbol{p})\right) \boldsymbol{p}. \tag{6.101}$$

Full compensation for all points with Eq. 6.101 is computationally prohibitively expensive during registration in real-time applications with modern LiDARs[11]. In our case, this also requires re-embedding of the point cloud.

Instead, we adapt the unscented transform (UT) (Julier and Uhlmann, 1997; Wan and Van Der Merwe, 2000) for an efficient surfelwise undistortion, as shown in Fig. 6.6. The UT generates from mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ a set of sigma points $\mathcal{Y}_\Sigma$ before applying a function $\bar{\boldsymbol{y}} = f(\boldsymbol{y}), \boldsymbol{y} \in \mathcal{Y}_\Sigma$ to these points. Finally, the UT recombines the transformed points to obtain the transformed mean $\bar{\boldsymbol{\mu}}$ and covariance $\bar{\boldsymbol{\Sigma}}$. Here, we use the symmetric set of sigma points $\mathcal{Y}_\Sigma$ with the Cholesky decomposition $LL^\intercal$ of the scaled covariance:

$$LL^\intercal = d \cdot \Sigma_s, \tag{6.102}$$

$$\mathcal{Y}_\Sigma = [\boldsymbol{\mu}_s + \boldsymbol{L}_0, \ldots, \boldsymbol{\mu}_s - \boldsymbol{L}_{d-1}], \tag{6.103}$$

with $d = 3$ and $\boldsymbol{L}_i$ referring to the ith column of L. The surfel mean $\boldsymbol{\mu}_s$ is deliberately excluded from $\mathcal{Y}_\Sigma$ to prevent previously occurring numerical round-off errors (see Eq. (30) in Wu et al. (2006)).

Thus far, we lack the sigma points' time $t_{\boldsymbol{y}}$. Hence, we transform each sigma point $\boldsymbol{y} \in \mathcal{Y}_\Sigma$ into the LiDAR frame and project using Eq. 6.1 and Eq. 6.2 with mean directional offset $\overline{o}_c$ from the surfel points:

$$t_{\boldsymbol{y}} = t(T_{\text{s,l}}^{-1}\boldsymbol{y}, \overline{o}_c). \tag{6.104}$$

Then, Eq. 6.101 and Eq. 6.6 provide the compensated sigma point $\bar{\boldsymbol{y}}_i$ with the surfels' segment time t_{seg}:

$$\Delta R_{\text{y,seg}} = \text{slerp}(\Delta\mathcal{R}_{\text{IMU}}, t_{\boldsymbol{y}})^{-1}\text{slerp}(\Delta\mathcal{R}_{\text{IMU}}, t_{\text{seg}}), \tag{6.105}$$

$$\bar{\boldsymbol{y}}_i = T_{\mathcal{X}}\left(t_{\text{seg}}\right)^{-1} T_{\mathcal{X}}\left(t_{\boldsymbol{y}}\right) \left[\Delta R_{\text{y,seg}} \boldsymbol{y}_i\right]. \tag{6.106}$$

Here, $\Delta R_{\text{y,seg}}$ counteracts the previous pre-orientation of $\widetilde{\mathcal{P}}_l$ in Eq. 6.6. Afterwards, transformed sigma points $\bar{\mathcal{Y}}_\Sigma$ are re-fused as in Wan and Van Der Merwe (2000) to obtain compensated mean $\bar{\boldsymbol{\mu}}$ and covariance $\bar{\boldsymbol{\Sigma}}$:

$$\bar{\boldsymbol{\mu}} = \sum_{\bar{\boldsymbol{y}} \in \bar{\mathcal{Y}}_\Sigma} \frac{1}{2d} \bar{\boldsymbol{y}}, \tag{6.107}$$

$$\bar{\boldsymbol{\Sigma}} = \sum_{\bar{\boldsymbol{y}} \in \bar{\mathcal{Y}}_\Sigma} \frac{1}{2d} \left(\bar{\boldsymbol{y}} - \bar{\boldsymbol{\mu}}\right)\left(\bar{\boldsymbol{y}} - \bar{\boldsymbol{\mu}}\right)^\intercal. \tag{6.108}$$

Although applicable during registration, e.g., after each LM-iteration, we found updating the sliding registration window $\mathcal{W}_l$ prior to registration to be sufficient.

11 CT-ICP (Dellenbach et al., 2022) reports an average computation time of 430 ms on Oxford Newer College (Ramezani et al., 2020) to simultaneously optimize two poses for one grid sampled and linearly interpolated scan.

6.2.10 *Keyframe Generation and Reuse*

We combine the sliding keyframe window of MARS (Ch. 5) with the keyframe storage of DLO (Chen et al., 2022) and take advantage of the regular structure of the lattice. All keyframes share an aligned common lattice with keyframe-specific shifts $\boldsymbol{\nu}_\mathrm{k} \in \mathbb{Z}^{d+1}$ to facilitate surfelwise fusion (Sec. 5.2.1) and ensure runtime efficiency without costly reintegration. Moreover, shifts allow us to maintain the local property of our map $\mathcal{M}$ and keyframes. For unseen areas, the storage approach is equivalent to the sliding keyframe window of MARS, while the traversal of known regions allows the reuse of previous keyframes.

direct LiDAR odometrys (DLOs)' adaptive distance threshold may be consistently too large in situations with obstructions, like pillars, while the LiDAR measures different surfaces with little overlap before reaching the threshold. For this reason, MAD-ICP (Ferrari et al., 2024) selects a recent scan to update its map if the percentage of successfully registered points drops below a certain threshold. Similarly, the SOD metric of AS-LIO (Zhang et al., 2024b) computes the overlap between the voxelized scan and map to adapt the length of the sliding window before merging the window into the map. Instead, we create a keyframe from the oldest scan within the registration window if less than 80 % of surfels in $\mathcal{S}$ are associated.

As the measured ranges and thus necessary map sizes vary with the surrounding environment, we adapt the map size by changing the coarse cell size c_0. While $c_0 = 4\,\mathrm{m}$ is sufficient for open surroundings, half the size (2 m) is more than enough for close quarters. This coincides by design with the coarse c_1 due to:

$$c_l = c_0 \cdot 2^{-l} \text{ with } l \in [0, \dots, 3]. \tag{6.109}$$

With four levels, the finest outdoor cell size c_3 is 0.5 m compared to 0.25 m indoors. Furthermore, both coarse cell sizes have three map resolutions in common. Thus, we seamlessly transition between narrow and wide areas by fusing map levels with the same cell size. The current coarse cell size c_0 depends on the mean surfel distance d_m filtered with an exponentially weighted moving average (EWMA):

$$d_m = \frac{1}{|S_l|} \sum_{s \in S_l} \|\boldsymbol{\mu}_s\|\,, \tag{6.110}$$

$$\bar{d}_m = \alpha \cdot d_m + (1 - \alpha) \cdot \bar{d}_m, \tag{6.111}$$

$$c_0 = \begin{cases} 4\,\mathrm{m}, & \text{if } \bar{d}_m \geq 3\,\mathrm{m}, \\ 2\,\mathrm{m}, & \text{else}, \end{cases} \tag{6.112}$$

similar to DLO's adaptive distance threshold.

After a positive keyframe decision, we correct the ego-motion for the keyframe scan pointwise with Eq. 6.101 using t_{s_j} as the reference time. In contrast to scans in $\mathcal{W}_l$, we embed each keyframe locally at the closest vertex after rotating the points into the common map frame. As a result, we set the keyframe-specific shift $\boldsymbol{\nu}_\mathrm{k}$ to the closest vertex $\boldsymbol{c}_\mathrm{o} \in \mathbb{Z}^{d+1}$. Using the coarsest level has the advantage that a shift on any finer resolution is a whole-numbered multiple of the coarse shift.

Our local map selection initially adds a third of the map window from the closest keyframes, sorted by ascending distance. For the remaining keyframes in range, we

compute the intersection over union (IoU) between a keyframe and current scan using the coarsest common map level. Embedding the current scan's coarse surfel mean $\boldsymbol{\mu}_s$ into each keyframe would be computationally involved. Instead, we embed them once into the common map lattice. Afterwards, we only require the cell shift between a keyframe's origin and the scan's origin. Adding this cell shift to the embedded indices provides the corresponding indices within the keyframe as if we embedded the coarse surfels directly. We fill the map $\mathcal{M}$ with up to F keyframes from all overlapping ones while preferring older keyframes to reduce drift over time.

6.2.11 *Implementation*

Instead of the full 3×3 covariance, we use the symmetry of the covariance matrix $\Sigma_s = \Sigma_s^\intercal$ for vectorization. This allows us to store the lower triangular matrix of Σ_s using the following $vec_L(\cdot) \in \mathbb{R}^8$ and recover it with the inverse $sym_L(\boldsymbol{a})$-operation:

$$vec_L(A) = [A_{00}, A_{11}, A_{22}, 0, A_{10}, A_{20}, A_{21}, 0]^\intercal, \tag{6.113}$$

$$sym_L(\boldsymbol{a}) = \begin{pmatrix} a_0, a_4, a_5, \\ a_4, a_1, a_6, \\ a_5, a_6, a_2 \end{pmatrix}. \tag{6.114}$$

It is easy to verify the following identities for $A = A^\intercal$:

$$sym_L(vec_L(A)) = A, \tag{6.115}$$

$$vec_L(sym_L(\boldsymbol{a})) = \boldsymbol{a}. \tag{6.116}$$

The vectorization simplifies memory alignment and use of single instruction, multiple data (SIMD) instruction sets, e.g., advanced vector extensions (AVX), with the vector class library (Fog, 2022). Addition and computation of the outer product in Eq. 5.3 become a single Fused Multiply-Add (FMA) operation per point $\boldsymbol{p}_i$.

For an efficient recomputation of $\boldsymbol{\Sigma}$ using $\widetilde{D}$ and Eq. 6.10, we make use of the Kronecker product $\otimes$ and its relation to the *vec*-function that stacks the columns of a matrix (Lancaster and Tismenetsky, 1985):

$$vec(A \cdot B \cdot C) = (C^\intercal \otimes A) \cdot vec(B). \tag{6.117}$$

Combining Eq. 6.10 and Eq. 6.14 with Eq. 6.117 gives:

$$vec(V \cdot \widetilde{D} \cdot V^\intercal) = (V \otimes V) \cdot vec(\widetilde{D}). \tag{6.118}$$

The diagonal matrix $\widetilde{D}$ has just three non-zero entries ($\widetilde{\boldsymbol{\lambda}}$). Hence, not the full $V \otimes V$ is required:

$$(V \otimes V) \cdot vec(\widetilde{D}) = \begin{bmatrix} V \cdot \left(\widetilde{\boldsymbol{\lambda}}^\intercal \odot [v_{00}, v_{01}, v_{02}]\right)^\intercal \\ V \cdot \left(\widetilde{\boldsymbol{\lambda}}^\intercal \odot [v_{10}, v_{11}, v_{12}]\right)^\intercal \\ V \cdot \left(\widetilde{\boldsymbol{\lambda}}^\intercal \odot [v_{20}, v_{21}, v_{22}]\right)^\intercal \end{bmatrix}, \tag{6.119}$$

where the Hadamard product $\odot$ (Horn and Johnson, 1991) performs element-wise multiplication. Since $V \cdot \widetilde{D} \cdot V^\intercal$ is symmetric, we rephrase Eq. 6.119 with Eq. 6.113:

$$E_V = \left[(\boldsymbol{v}_{0:} \odot \boldsymbol{v}_{1:})^\intercal, (\boldsymbol{v}_{0:} \odot \boldsymbol{v}_{2:})^\intercal, (\boldsymbol{v}_{1:} \odot \boldsymbol{v}_{2:})^\intercal \right], \tag{6.120}$$

$$vec_L(V \cdot \widetilde{D} \cdot V^\intercal) = \left[(V \odot V)^\intercal, \mathbf{0}, E_V^\intercal, \mathbf{0} \right]^\intercal \cdot \widetilde{\boldsymbol{\lambda}}, \tag{6.121}$$

where $\boldsymbol{v}_{i:}$ is the ith row vector of V. This allows an efficient SIMD implementation using two FMA and three multiply instructions. See App. B.2 for a verification script using the MATLAB symbolic toolbox.

Similar optimizations with Eq. 6.117 are applicable for Eq. 6.9 to rephrase $R(t)\Sigma_s R(t)^\intercal$ into a single matrix-vector-product. We introduce H_1 to map symmetric matrices from $vec_L(\cdot) \in \mathbb{R}^8$ to $vec(\cdot) \in \mathbb{R}^9$ and H_2 for mapping $vec(\cdot) \in \mathbb{R}^9$ to $vec_L(\cdot) \in \mathbb{R}^8$ using the basis vectors $\boldsymbol{e}_i$ with the Kronecker delta δ_{ij} :

$$\boldsymbol{e}_i = [0, \ldots, \delta_{ii}, \ldots, 0]^\intercal, \tag{6.122}$$

$$H_1 = \begin{bmatrix} \boldsymbol{e}_0 & \boldsymbol{e}_4 & \boldsymbol{e}_5 & \boldsymbol{e}_4 & \boldsymbol{e}_1 & \boldsymbol{e}_6 & \boldsymbol{e}_5 & \boldsymbol{e}_6 & \boldsymbol{e}_2 \end{bmatrix}^\intercal, \tag{6.123}$$

$$H_2 = \begin{bmatrix} \boldsymbol{e}_0 & \boldsymbol{e}_4 & \boldsymbol{e}_5 & \mathbf{0} & \boldsymbol{e}_1 & \boldsymbol{e}_6 & \mathbf{0} & \mathbf{0} & \boldsymbol{e}_2 \end{bmatrix}. \tag{6.124}$$

Thus, simplifying the use of the Kronecker product with $vec_L(\cdot)$ such that:

$$Z = H_2 \left(R(t) \otimes R(t) \right) H_1, \tag{6.125}$$

$$sym\left(\left(R(t) \otimes R(t) \right) vec\left(\Sigma_s \right) \right) = sym_L\left(Z \cdot vec_L(\Sigma_s) \right). \tag{6.126}$$

The matrix Z is constant for all surfels ($\gtrsim 100$) at time t, allowing the precomputation of Z and frequent reuse for Eq. 6.9:

$$vec_L(\Sigma_m + R(t)\Sigma_s R(t)^\intercal) = Z \cdot vec_L(\Sigma_s) + vec_L(\Sigma_m). \tag{6.127}$$

An efficient implementation needs just 6 FMA instructions since $vec_L(\cdot)$ has 2 zero-entries.

Mahalanobis Distance

Each registration iteration computes more than 10 000 Mahalanobis distances in Eq. 6.15. Every distance requires the inversion of a symmetric 3×3 matrix. In general, explicit matrix inversion is discouraged (Higham, 2002), e.g., when solving linear systems as the obtained solutions are less accurate for ill-conditioned matrices. However, a surfel integrates only local information due to the subdivision by the lattice and the fusion of a small number of spatially distributed scans. Furthermore, the GMM includes a resolution-depending scaling term $\sigma_l^2 I$ that reduces the condition number (see Eq. 2.46). Hence, some very fast analytical solutions exist for small square matrices due to the Cayley-Hamilton theorem (Visser et al., 2006) and the Leverrier-Faddeev method (Hou, 1998). As a result, the inverse of a 3×3 matrix A (Gantmakher, 1960) is:

$$A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} a_{11}a_{22} - a_{12}a_{21} & a_{12}a_{20} - a_{10}a_{22} & a_{10}a_{21} - a_{11}a_{20} \\ a_{02}a_{21} - a_{01}a_{22} & a_{00}a_{22} - a_{02}a_{20} & a_{01}a_{20} - a_{00}a_{21} \\ a_{01}a_{12} - a_{02}a_{11} & a_{02}a_{10} - a_{00}a_{12} & a_{00}a_{11} - a_{01}a_{10} \end{bmatrix}. \tag{6.128}$$

Using the symmetry of A allows some further simplification with $a_{01} = a_{10}$, $a_{02} = a_{20}$ and $a_{12} = a_{21}$:

$$A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} a_{11}a_{22} - a_{21}^2 & a_{20}a_{21} - a_{10}a_{22} & a_{10}a_{21} - a_{11}a_{20} \\ a_{20}a_{21} - a_{10}a_{22} & a_{00}a_{22} - a_{20}^2 & a_{10}a_{20} - a_{00}a_{21} \\ a_{10}a_{21} - a_{11}a_{20} & a_{10}a_{20} - a_{00}a_{21} & a_{00}a_{11} - a_{10}^2 \end{bmatrix}. \tag{6.129}$$

We rephrase Eq. 6.129 with $vec_L(\cdot)$ as:

$$\boldsymbol{m} = vec_L(A), \tag{6.130}$$
$$\boldsymbol{a}_0 = [m_1, m_0, m_0, 0, m_5, m_4, m_5, 0]^\intercal, \tag{6.131}$$
$$\boldsymbol{b}_0 = [m_2, m_2, m_1, 0, m_6, m_6, m_4, 0]^\intercal, \tag{6.132}$$
$$\boldsymbol{a}_1 = [m_6, m_5, m_4, 0, m_4, m_5, m_6, 0]^\intercal, \tag{6.133}$$
$$\boldsymbol{b}_1 = [m_6, m_5, m_4, 0, m_2, m_1, m_0, 0]^\intercal, \tag{6.134}$$
$$\boldsymbol{c} = \boldsymbol{a}_0 \odot \boldsymbol{b}_0 - \boldsymbol{a}_1 \odot \boldsymbol{b}_1, \tag{6.135}$$
$$\det(A) = m_0c_0 + m_4c_4 + m_5c_5, \tag{6.136}$$
$$A^{-1} = sym_L\left(\frac{1}{\det(A)}\boldsymbol{c}\right). \tag{6.137}$$

An efficient implementation needs one multiply and one FMA operation for Eq. 6.135 in addition to the dot product for $\det(A)$ and one division.

Embedding into the Permutohedral Lattice

The embedding of every point $\boldsymbol{p}$ into the permutohedral lattice (Sec. 5.2.1) is the most costly process during map creation. After lifting $\boldsymbol{p}$ and scaling with σ_l^{-1}, we compute the closest remainder-0 point $\mathbf{y} \in H_d$ of the simplex (see Lemma 2.9 and Fig. 3 in Adams et al. (2010)). For this, we first unroll the rank computation into $d+1$ parallel rounds. The rank $\boldsymbol{r}_H = P_H \cdot [0, \dots, d]^\intercal$ represents an unsorted permutation of $(0, \dots, d)$ with permutation matrix P_H.

The next step sets the distances $\boldsymbol{d}$ to the permuted position ($P_H\boldsymbol{b} := \boldsymbol{d}$) to obtain the barycentric coordinates. This typical "scatter"-operation ($b[ind[i]] := d[i]$) has only been recently added for memory access within AVX512 (Intel Corporation, 2023a), but remains unavailable for registers.

Instead, permuting the distances inversly[12] ($\boldsymbol{b} := P_H^\intercal\boldsymbol{d}$) allows to use efficient gather-operations ($b[i] := d[idx[i]]$), e.g., shuffle of SSE3 (Intel Corporation, 2023b) and AVX permute (Intel Corporation, 2021) instructions. We verified by enumeration for $d = 3$ (see App. B.3) that sorting the rank $\boldsymbol{r}_H$ and setting $idx[i] = ind(sorted[i])$ indeed computes the correct index. Hence, we encode the original index $(0, \dots, d)$ in the lower $\log_2(d+1) = 2$ bits after shifting the rank by the same number of bits. Then, we sort blocks of $d+1 = 4$ integers in parallel using three min and max operations.

After extracting the original index offset and adding the block offset, we shuffle according to the new index $idx[i]$. The barycentric coordinates become readily available as the difference between neighboring entries (see Proposition 4.2 in Adams et al. (2010)). We retain $\mathbf{y}$ as the one with the highest barycentric weight.

12 The inverse of a permutation matrix is its transpose (Pissanetzky, 1984).

Table 6.1: RMS-ATE [m] evaluation on the Newer College (Ramezani et al., 2020) dataset. Algorithms are grouped by LO/LIO and ordered according to publication date. An "X" marks divergence. Lower values are better (↓) with third, second and **best** highlighted.

	MARS	DLO	KISS-ICP	Traj-LO	CLINS	Fast-LIO2	Point-LIO	SLICT2	DLIO	SE-LIO	iG-LIO	LIO-MARS
Online	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓
LIO					✓	✓	✓	✓	✓	✓	✓	✓
CT*	✓			✓	✓			✓		✓		✓
01_Short	1.110	0.359	0.630	**X**	0.535	0.340	0.376	0.363	0.355	0.291	**0.282**	0.601
02_Long	3.198	0.489	**X**	**X**	0.399	0.329	**X**	0.399	0.384	**0.301**	0.338	0.396
05_Quad	0.292	0.124	0.134	0.201	**0.105**	0.111	0.171	0.109	0.120	0.113	0.109	0.132
06_Spin†	0.105	**X**	**X**	0.096	0.094	0.091	**X**	0.093	0.134	0.087	0.093	0.095
07_Park	2.278	0.213	**X**	42.321	0.202	**0.125**	0.138	0.139	0.139	0.140	0.129	0.169
Fail [%]	**0**	25.00	35.00	50.00	**0**	**0**	30.00	**0**	**0**	**0**	**0**	**0**
Avg. Rank	10.50	7.25	10.75	11.50	5.75	2.50	8.00	4.50	4.50	3.50	**2.00**	7.25
Overall	10.	7.	11.	12.	6.	2.	9.	4.	4.	3.	**1.**	7.

* Continuous-time trajectory

† Inconsistent ground-truth, excluded from ranking. Failures manually inspected and verified.

Table 6.2: RMS-ATE [m] evaluation on the Newer College extension (Zhang et al., 2021). Algorithms are grouped by LO/LIO and ordered according to publication date. An "X" marks divergence. Lower values are better (↓) with third, second and **best** highlighted.

	MARS	DLO	KISS-ICP	Traj-LO	CLINS	Fast-LIO2	Point-LIO	SLICT2	DLIO	SE-LIO	iG-LIO	LIO-MARS
Online	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓
LIO					✓	✓	✓	✓	✓	✓	✓	✓
CT*	✓			✓	✓			✓		✓		✓
Math-E	0.151	0.174	0.123	0.109	0.096	0.080	0.153	0.145	0.131	0.138	**0.062**	0.157
Math-M	0.187	2.492	3.865	0.150	0.124	0.106	0.181	0.126	0.145	0.206	**0.101**	0.141
Math-H	0.135	**X**	**X**	0.087	**0.059**	0.066	0.138	0.132	0.070	0.110	0.062	0.112
Mine-E	0.087	0.084	0.085	0.045	0.039	0.049	0.045	0.054	**0.036**	0.089	0.052	0.041
Mine-M	0.094	0.761	0.165	0.050	0.071	0.046	0.046	0.051	0.047	0.312	0.055	**0.044**
Mine-H	0.114	5.245	**X**	0.067	**X**	0.053	**0.052**	0.068	0.071	0.120	0.056	0.056
Quad-E	0.152	0.079	0.084	0.076	**0.067**	0.070	0.074	0.071	0.069	0.082	0.070	**0.067**
Stairs	**X**	0.135	**X**	**X**	**X**	**X**	0.222	**X**	0.117	**X**	0.333	**0.103**
Quad-M	0.117	0.122	15.166	0.068	**0.058**	0.060	0.066	0.064	0.062	**X**	0.062	0.061
Quad-H	0.306	4.094	**X**	0.072	**0.047**	0.056	0.066	0.103	0.069	**X**	0.062	0.063
Park	2.931	0.902	29.325	**X**	4.043	0.326	1.376	0.331	0.308	0.298	**0.263**	0.288
Cloister	0.303	0.204	**X**	0.085	0.115	0.073	0.103	0.116	0.095	0.108	**X**	**0.069**
Fail [%]	8.33	8.33	35.00	16.67	16.67	5.00	**0**	1.67	**0**	25.00	3.33	**0**
Avg. Rank	9.67	9.50	10.83	6.67	5.08	3.67	5.92	7.17	4.25	9.17	4.17	**3.50**
Overall	11.	10.	12.	7.	5.	2.	6.	8.	4.	9.	3.	**1.**

* Continuous-time trajectory

6.3 Evaluation

All experiments were conducted on a laptop with an AMD Ryzen 7 5800H and 48 GB of random access memory (RAM). We use the Oxford Newer College (Extension) dataset (Ramezani et al., 2020; Zhang et al., 2021), and our DRZ Living Lab dataset (Sec. 5.3.3) for evaluation. The selected datasets pose unique challenges due to their different characteristics. The handheld sensor motion of Newer College is, in general, slower yet more abrupt and fits through narrower passages like corridors. In contrast, unmanned aerial vehicle (UAV) flights achieve higher accelerations and rotational speeds while exhibiting more continuous movement.

We compare our system, called LIO-MARS, to in total 11 other methods, which can be divided into LiDAR odometrys (LOs) and LIOs. The real-time LO baselines are MARS (Ch. 5), DLO (Chen et al., 2022), and KISS-ICP (Vizzo et al., 2023). The

online LIO systems include Fast-LIO2 (Xu et al., 2022), DLIO (Chen et al., 2023a), SE-LIO (Yuan et al., 2025), SLICT2 (Nguyen et al., 2024b), Point-LIO (He et al., 2023) and iG-LIO (Chen et al., 2024b).

We further tested Traj-LO (Zheng and Zhu, 2024b), CLINS (Lv et al., 2021) and Coco-LIC (Lang et al., 2023). These systems only provided interfaces for offline processing of ROS bags[13]. However, in its LIO mode, Coco-LIC routinely diverged on all datasets. For better comparability, we disabled the loop-closing of SLICT2 and CLINS, as none of the other methods have explicit loop-closing. Additionally, we enabled the real-time flag for SLICT2 to maintain its real-time performance and prevent accumulation of unprocessed scans. If available, the algorithms use per dataset the recommended parameters by the respective authors. In their absence, we adapt the parameters from a similar dataset and set the intrinsics for IMU and LiDAR as well as extrinsics according to the dataset's calibration. Each algorithm uses a single parameter set per dataset without per-sequence adaptation.

Our evaluation uses Evo (Grupp, 2017) to compute the root-mean-squared (RMS)-absolute trajectory error (ATE) (see Sec. 2.7) w.r.t. the dataset's reference poses after SE(3)-alignment. Per dataset, each sequence runs in real-time with a single algorithm under evaluation. After processing the scan, we store the current pose. We repeat the evaluation 5 times for each method to report the average RMS-ATE. If, during a single run, the length of the estimated trajectory differs from the ground-truth length by more than 25 % or the mean error is above 50 m, we mark the algorithm as diverged for the sequence. Hence, we report the percentage of failed runs per dataset.

Given the large number of methods under evaluation, a clear overall winner may not be apparent. A single run with a higher error or divergence can impact the result too negatively when comparing the mean RMS-ATE over all sequences. Discarding the diverged runs would bias the comparison towards successful runs without penalty for failure. Per sequence, we thus rank the n algorithms based on their RMS-ATE from first to nth place and compute the average rank over all sequences. All diverged methods receive the nth place for the corresponding sequence.

6.3.1 *Newer College Dataset*

This dataset (Ramezani et al., 2020) contains five sequences[14] captured with a hand-held Ouster OS1-64 LiDAR with integrated IMU. Zhang et al. (2021) prepared a multi-camera-inertial-LiDAR extension with an Ouster OS0-128 for another 12 sequences, partially recreating the original dataset. During the evaluation, all algorithms receive the OS1-64 scans and its IMU for the five Newer College sequences and the OS0-128 scans with an external Alphasense IMU for the extension. The reference poses stem from ICP registration[15] against TLS point clouds. Table 6.1 presents the results for the five original sequences, whereas Tab. 6.2 shows the 12 newer sequences. The values for MARS differ w.r.t. Ch. 5 due to several bug fixes after its original publication.

13 http://wiki.ros.org/rosbag
14 The sequences are not consecutively numbered: 01, 02, 05, 06, 07.
15 Ramezani et al. (2020) do not report motion compensation.

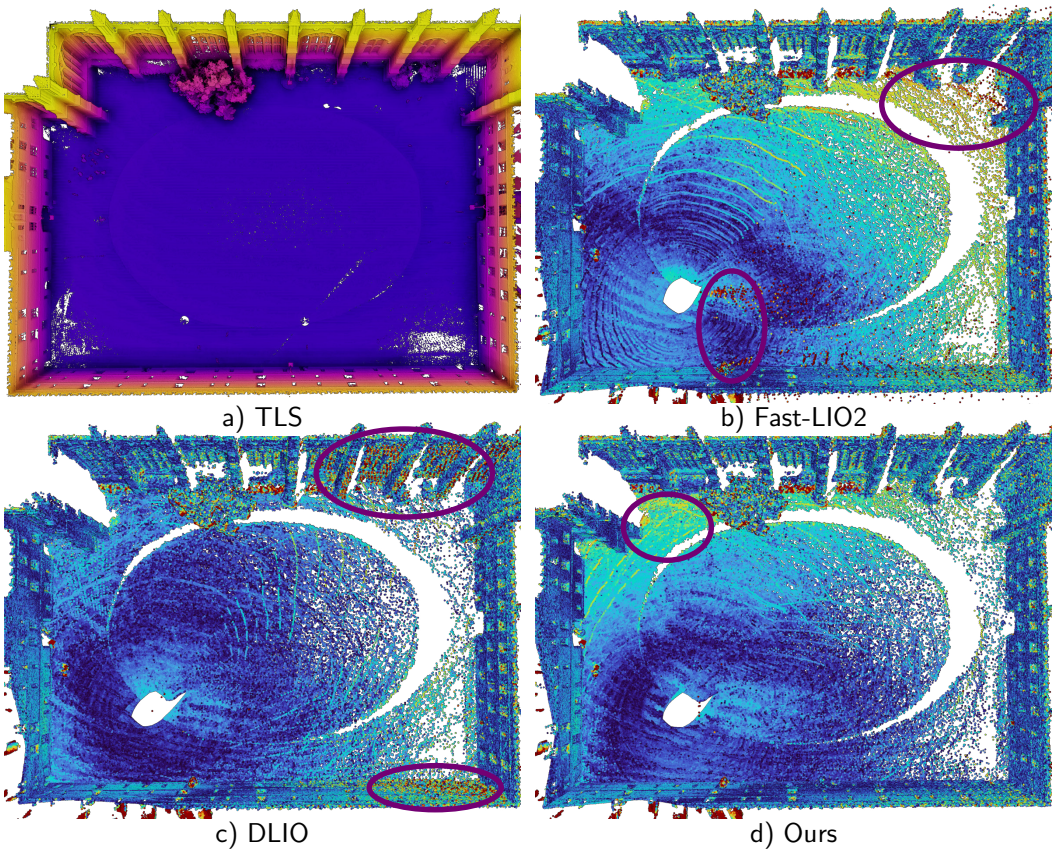

Figure 6.7: Comparison after compensation on "06_Spin" of Newer College (Ramezani et al., 2020): height-colored ground-truth point cloud from terrestrial LiDAR scanner (TLS) BLK360 [a)], aggregation of every 25th scan compensated by Fast-LIO2 [b)], DLIO [c)] and our LIO-MARS [d)]. Color in [b)-d)] encodes point-to-plane error w.r.t. ground-truth TLS map from low (blue) to high (red, $\geq 0.25\,\mathrm{m}$). Ellipses highlight areas with noticable differences between methods.

As expected, the LIO systems outperform their LO counterparts since the IMU provides valuable complementary information to LiDAR. Furthermore, methods with keyframe reuse or map reuse have an inherent advantage on the longer sequences (01, 02, and 07). The keyframe re-creation for previously visited areas in MARS (Ch. 5) leads to drift over time.

KISS-ICP instead struggles with the unsteadier characteristics of handheld sensors, which is quite different from its typical automotive driving scenario. Frequent obstruction of a small scan portion behind the sensor constitutes an additional challenge for Point-LIO and the continuous-time methods. Interestingly, the continuous-time methods perform slightly worse than the conventional methods on the first two sequences, with the only exception being SE-LIO. In these sequences, our method accumulates some drift in the parkland section leading back to the parkland mount.

The sequence "06_Spin" poses a particular challenge, as reflected by the high number of diverged solutions. The sequence contains varying high rotational velocities of up to $3.5\,\mathrm{rad\,s^{-1}}$ where motion compensation is essential. Figure 6.7 shows the aggregation of every 25th scan after compensation by Fast-LIO2, DLIO, and our approach. Each point cloud contains around 2 800 000 points. We subsample the

Table 6.3: RMS point-to-plane distance [cm] for every 25th scan for "06_Spin" (Ramezani et al., 2020) w.r.t. TLS map. Lower values are better (↓) with second and **best** highlighted.

$d_{\max}$ [cm]	Fast-LIO2	DLIO	LIO-MARS	LIO-MARS†
100	7.85	7.65	**7.59**	7.63
50	6.50	6.41	**6.31**	6.35
25	5.93	5.82	**5.73**	5.78

†Motion compensation on adaptive selected surfels between first and second registration iteration.

clouds to 5 cm resolution using CloudCompare for better alignment with the TLS map. After manual initialization, CloudCompare's ICP fine registration aligns the subsampled clouds against the TLS map with 50 000 random sampled points. We set the "final overlap" to 90 % and enabled "farthest point removal". Afterwards, we apply the estimated pose to the original cloud and compute point-to-plane errors w.r.t. the TLS map to assess the compensation and registration quality.

Since no scan is visibly missaligned, we threshold the point-to-plane errors to 0.25 m (resp. 0.5 m and 1 m) to reduce the influence of non-represented parts within the quad. Table 6.3 shows that our map consistently has the lowest RMS error (RMSE) before DLIO and Fast-LIO2 without any distinct red areas except for the growing tree and bushes. Even if we compensate during registration in between the first and second iteration instead of prior to registration, we obtain a better result than the competing methods. Fast-LIO2 exhibits some spurious measurements and incorrectly compensated scans.

The sequences of the extension (Zhang et al., 2021) showcase more variability w.r.t. the results in Tab. 6.2. Figure 6.8 shows our reconstruction using every 25th scan of the challenging "Stairs". Here, the operator moved the handheld sensor setup from a hallway through a door into a narrow staircase and multiple flights of stairs upwards before heading back down again. The close quarters make this sequence challenging as the ceiling or floor are temporarily not measured during turning. This frequently leads to underconstrained directions in the optimization. Furthermore, the LiDAR measures the stairs from above and below. This can erroneously pull both surfaces together if the map resolution is too coarse or incorrect correspondences are not rejected.

As part of their strategy to cope with the high amount of measurements per scan, most approaches use a single resolution per voxel, e.g., in conjunction with a voxel filter and random or naïve downsampling. As a result, Fast-LIO2 performs well in most cases but diverges on the "Stairs" sequence. A fine resolution (0.25 m) likely helps DLO and DLIO in this scenario, whereas a 0.4 m to 0.5 m resolution is more common. Traj-LO and KISS-ICP follow the approach of CT-ICP to store at most a fixed number of points per voxel, which effectively allows the map to represent finer details too. In contrast, SLICT2 and our method adjust data-dependently the size of a map cell dynamically.

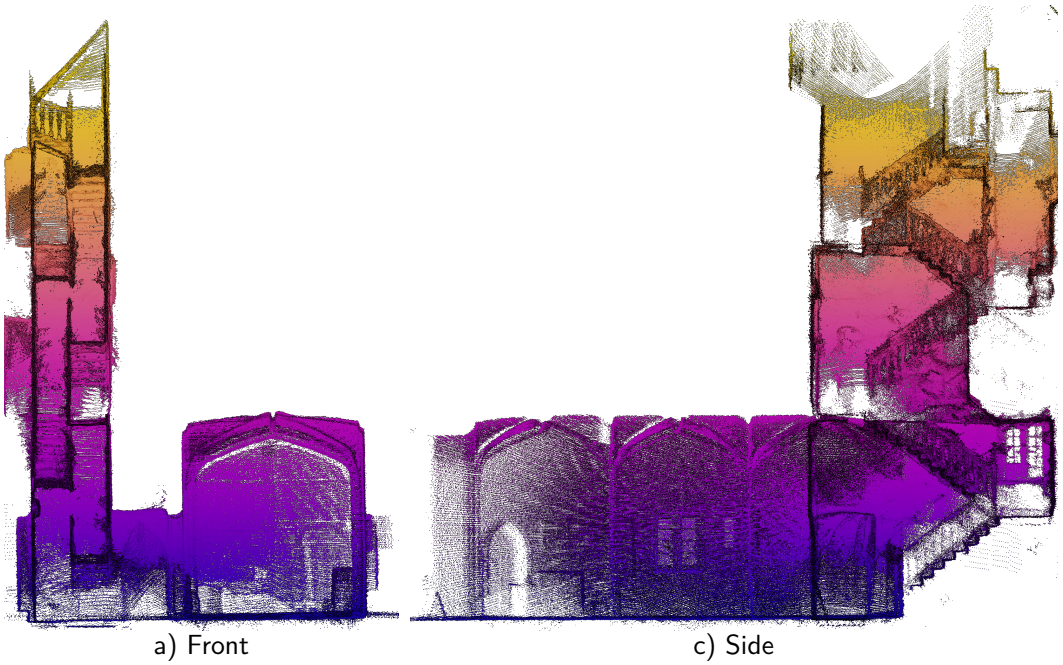

Figure 6.8: Aggregated Pointcloud for "Stairs" of Newer College (Zhang et al., 2021).

For our map (Sec. 6.2.10), adapting the coarse cell size c_0 in combination with the resolution selection (Sec. 5.2.3) facilitates the handling of different environments. Our system starts outside with its standard coarse 4 m cell size such that the finest resolution c_3 has 0.5 m cells. After walking through the door into the staircase, LIO-MARS sets c_0 to 2 m (resp. $c_3 = 0.25$ m) as the mean surfel distance reduces. The process reverses in the end after walking out into the broader hallway. Similarly, such adaptation happens when entering and leaving the narrow passage of the "Cloister" sequence.

Only three methods, DLIO, Point-LIO, and LIO-MARS, worked successfully on all sequences. At the same time, our LIO-MARS is the best performing continuous-time method and ranks best over all sequences of the extension.

6.3.2 *DRZ Living Lab*

This dataset (Sec. 5.3.3) was recorded onboard a flying DJI M210 v2 with an Ouster OS0-128 LiDAR. The DRZ Living Lab is located in a large industrial hall and accommodates facilities to test robots in difficult terrain or rescue scenarios. A Motion Capture (MoCap) system measures the UAV pose over time and provides the reference trajectories for evaluation. This packed indoor environment leads to a high number of valid measurements compared to flights in outdoor environments with open skies. Per scan, the "F2" sequence averages 92 745 valid points, which equals to 70.7 % of possible measurements.

In comparison to the previous chapter, we replace "Fast" and "Hall" with the new and more challenging sequences "H1" to "H3". Figure 6.9 shows the reconstruction of "H3" using LIO-MARS. Here, the UAV starts in the MoCap volume and flies through the hall and out the rear gate. It then flies in the alleyway between the buildings

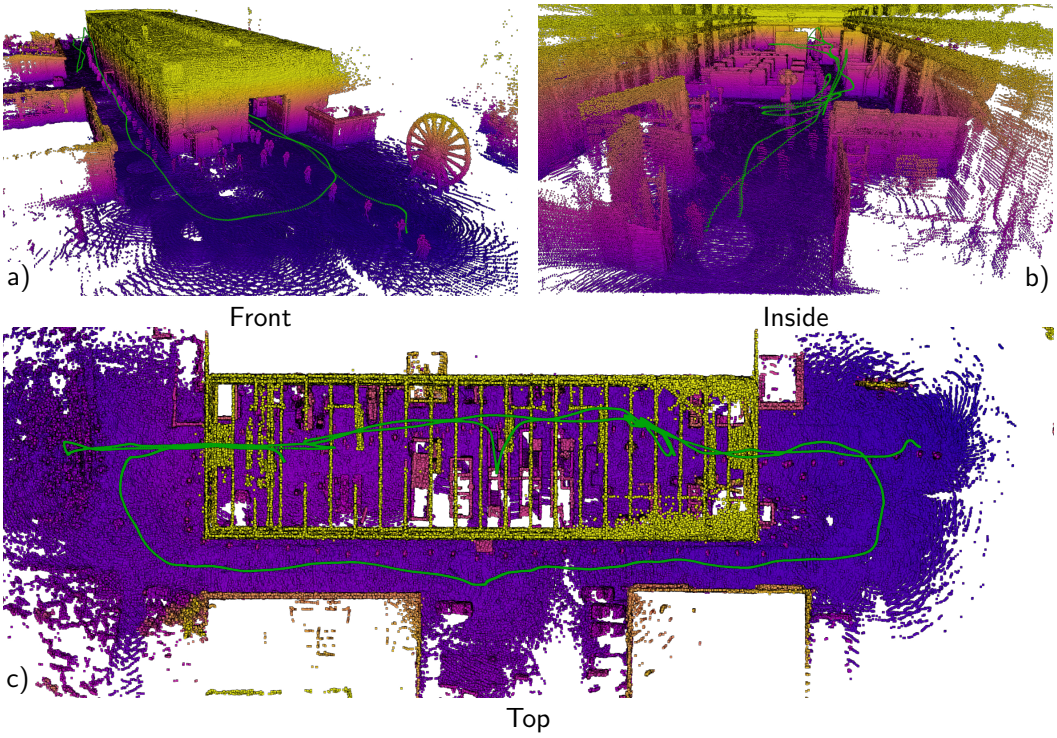

Figure 6.9: Aggregated Pointcloud for "H3" from the DRZ Living Lab dataset with overlayed trajectory (green line), colored by height from low (blue) to high (yellow). An UAV with a LiDAR started in front of the building [a)] and flew through the building [b)]. It traversed back through the alleyway and into the building a second time to land in the back. The OS-0 LiDAR provides dense 3D measurements within the hall. Roof partially removed in b) for better visualization of the interior.

Table 6.4: RMS-ATE [m] evaluation for the DRZ Living Lab dataset. Algorithms are grouped by LO/LIO and ordered according to publication date. An "X" marks divergence. Lower values are better (↓) with third, second and **best** highlighted.

	MARS	DLO	KISS-ICP	Traj-LO	CLINS	Fast-LIO2	Point-LIO	SLICT2	DLIO	SE-LIO	iG-LIO	LIO-MARS
Online	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓
LIO					✓	✓	✓	✓	✓	✓	✓	✓
CT*	✓			✓	✓			✓		✓		✓
H1	0.047	0.060	0.065	0.055	0.025	0.019	**0.018**	0.022	0.054	0.020	0.019	0.020
H2	0.048	0.060	0.059	0.060	0.028	**0.016**	0.018	0.025	0.052	0.022	0.019	0.017
H3	3.760	1.115	0.180	0.056	0.030	0.031	0.023	0.389	0.056	**X**	0.025	**0.022**
3P-S	0.018	0.020	0.027	0.020	0.011	0.012	**0.008**	0.011	0.014	0.012	0.011	0.010
3P-M	0.037	0.042	0.050	0.042	0.019	0.013	**0.010**	0.016	0.036	0.013	0.016	**0.010**
3P-F	0.056	**X**	**X**	0.087	0.039	0.039	**0.016**	0.020	0.061	0.022	0.019	0.021
S1	0.051	0.059	0.062	0.052	0.034	0.039	**0.025**	0.029	0.048	0.028	0.029	0.026
M1	0.079	0.104	**X**	0.084	0.052	0.044	0.034	0.036	0.075	0.040	0.036	**0.032**
F2	0.103	0.164	**X**	0.118	0.072	0.060	**0.054**	0.057	0.110	0.058	0.058	0.056
F3	0.068	0.093	0.098	0.081	0.055	0.041	0.027	0.030	0.068	0.030	0.030	**0.025**
Fail [%]	**0**	8.00	24.00	**0**	**0**	**0**	**0**	**0**	**0**	10.00	**0**	**0**
Avg. Rank	8.70	10.80	11.40	9.70	6.10	4.80	**1.50**	4.50	8.20	5.00	3.30	2.00
Overall	9.	11.	12.	10.	7.	5.	**1.**	4.	8.	6.	3.	2.

* Continuous-time trajectory

towards the front entrance and back into the MoCap volume. The forward direction along the alleyway is only partially constrained due to the LiDAR's limited range and the scene's geometry.

Table 6.4 shows the results for all considered methods (see Sec. 6.3). All algorithms perform reasonably well, even on the faster sequences with high rotational and linear velocities. However, LIO-MARS and Point-LIO outperform the other LOs and LIOs algorithms.

Over all three datasets, only DLIO and LIO-MARS did not fail on any sequence as shown in Tab. 6.5 and Fig. 6.10. In contrast, iG-LIO failed twice in the "Cloister". SLICT2 and Fast-LIO2 failed once, resp. thrice, in the more challenging "Stairs" sequence. Although Point-LIO performed well on the DRZ and Newer College extension sequences, it diverged multiple times on the original sequences. In total, LIO-MARS ranked best, achieving state-of-the-art performance on these UAV and hand-held sequences.

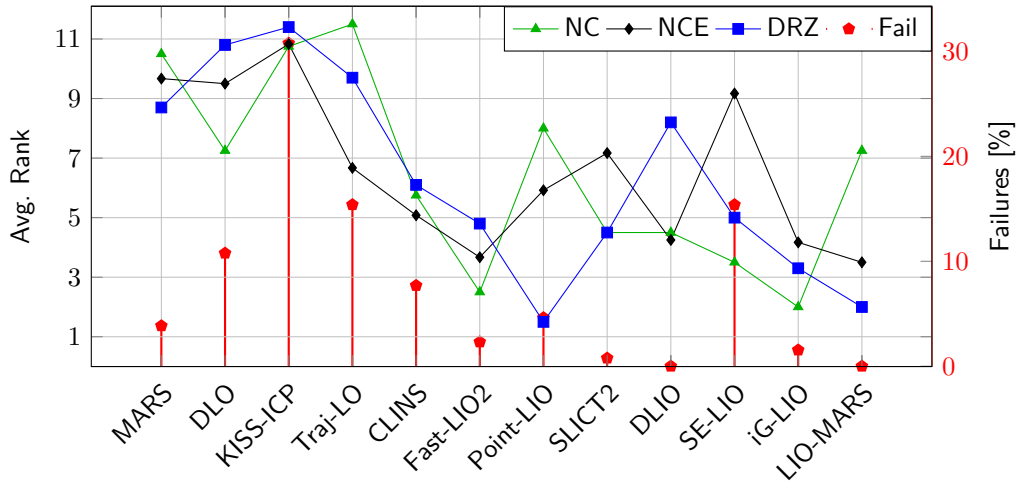

Figure 6.10: Average Ranks per algorithm for the tested datasets with failure rate. Lower values (↓) are better.

Table 6.5: Ranking over all tested datasets. Algorithms are grouped by LO/LIO and ordered according to publication date. Lower values are better (↓) with third, second and **best** highlighted.

	MARS	DLO	KISS-ICP	Traj-LO	CLINS	Fast-LIO2	Point-LIO	SLICT2	DLIO	SE-LIO	iG-LIO	LIO-MARS
Online	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓
LIO					✓	✓	✓	✓	✓	✓	✓	✓
CT*	✓			✓	✓			✓		✓		✓
Ranks	245	251	287	223	145	102	118	149	151	174	**91**	**91**
Failed†	5	14	40	20	10	3	6	1	**0**	20	2	**0**
Fail [%]	3.85	10.77	30.77	15.39	7.69	2.31	4.62	0.77	**0**	15.39	1.54	**0**
Overall	10.	11.	12.	9.	5.	3.	4.	6.	7.	8.	2.	**1.**

* Continuous-time trajectory

† Total number of failed runs over 24 sequences with 5 runs each.

Table 6.6: Statistics for varying spline parameters on the "F2" sequence in the DRZ Living Lab. Lower values are better (↓) with third, second and **best** highlighted.

Spline				RMS-ATE	Map Emb.	Comp.	Spline Init.	Reg.	Avg. Time
N	$\lvert\mathcal{W}\rvert$	$\lvert\mathcal{X}\rvert$	O	[m] (↓)	[ms] (↓)	[ms] (↓)	[ms] (↓)	[ms] (↓)	[ms] (↓)
2	3	3	8	0.0945	10.9	15.9	9.2	17.0	59.2
2	4	4	8	0.0689	11.0	17.9	12.9	21.9	68.9
2	5	5	8	0.0701	10.8	30.7	17.2	27.5	92.4
2	6	6	8	0.0713	10.9	31.5	21.1	32.9	102.8
3	2	2	2	0.0659	17.8	**15.3**	7.6	13.1	59.7
3	2	2	4	0.0567	12.2	16.4	6.5	**11.0**	51.6
3	2	2	8	0.0567	10.8	18.0	**6.3**	**11.0**	**51.5**
3	3	3	2	0.0562	17.9	16.6	10.0	18.0	68.6
3	3	3	4	0.0560	12.4	17.7	8.8	16.9	61.4
3	3	3	8	0.0562	10.8	19.1	9.4	17.1	62.4
3	3	6	2	0.0572	17.8	16.5	12.5	18.7	71.8
3	3	6	4	0.0561	10.2	18.5	7.3	17.4	60.0
3	3	6	8	0.0561	**9.1**	17.3	10.3	15.8	58.5
3	4	4	8	0.0562	10.8	20.1	13.7	22.0	72.6
3	5	5	8	**0.0559**	10.8	36.6	17.1	28.3	99.1
3	6	6	8	0.0562	10.9	37.8	21.1	33.3	109.5
4	3	6	4	0.1216	12.8	20.3	13.8	20.4	78.2
4	3	6	8	0.1026	11.1	21.9	15.3	20.3	78.6

6.3.3 *Ablation*

In the previous evaluations, LIO-MARS uses a third-order spline ($N = 3$), three scans in the current scan window $\mathcal{W}$, and six optimizable knots. Our surfel compensation applies only to the new scan whereas the two previous scans in the current window are already compensated.

To better understand the effect of our design decisions, we evaluate different spline parameters on the "F2" sequence captured within the MoCap of the DRZ Living Lab. Table 6.6 reports the results including the resp. timing for individual components.

Allocating all resources can become dangerous on a robotic system as multiple processes will interfere with each other. Hence, we limit the number of threads to four which is sufficient to process each map level in parallel. This leaves enough

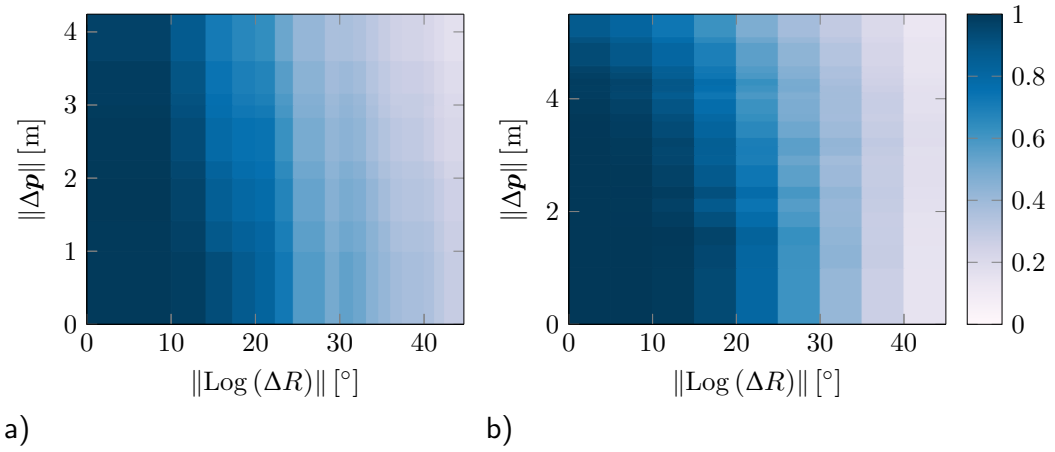

Figure 6.11: Ablation on registration offset: Success rate w.r.t. pose offset with full orientation [a)] and only yaw [b)]. The darker blue shade is better (↑).

computational resources for mapping, navigation, and robot control. However, we expect a further runtime reduction when all threads can be used.

Increasing the number of segments O reduces the time required to embed a point cloud into the map. When more segments subdivide a scan, the number of surfels per segment decreases. This, in turn, leads to faster access to individual cells since we store each segment in a separate small hash map. Evidently, this time remains constant independent of the number of scans ($|\mathcal{W}|$) within the sliding window $\mathcal{W}$, as shifting the window removes the oldest scan and adds a new scan.

Overall, our system performs best w.r.t. RMS-ATE with a spline of order $N = 3$ and an equal number of scans or more in the sliding window $|\mathcal{W}| \geq N$. Optimizing fewer than N knots can lead to oscillations and reduce accuracy. Our current implementation limits the number of optimizable knots to 6, which sets the maximum scan window size to 6. Alternatively, three scans with two knots per scan allow more flexible trajectories, e.g., as used by SLICT2. This reduces the temporal knot distance from $\approx 100\,\mathrm{ms}$ to $50\,\mathrm{ms}$. However, our results show not much of an additional benefit w.r.t. accuracy for a reduced temporal knot distance.

As expected, a larger sliding window increases the overall computation time for the compensation, initialization, and registration. Moreover, it extends the time until a previously unseen area becomes part of the map, which can degrade performance in larger or obstructed environments.

In comparison with MARS (Ch. 5), LIO-MARS can optimize more iterations, 5 instead of 3, in less time. Moreover, our system increases the map's resolution while reducing its computational cost.

6.3.3.1 *Relative Motion Constraints*

So far, our experiments only use the mixture of preintegrated and raw IMU measurements from Sec. 6.2.8. Here, we leverage the "F2" sequence from the DRZ dataset. With the standard configuration, LIO-MARS achieves an RMS-ATE of 0.056 m. Using only raw IMU without preintegration results in an RMSE of 0.057 m.

Replacing the IMU measurements with a single relative pose measurement ΔT during optimization increases the RMS-ATE to 0.093 m in our experiment. We attribute this increase to inaccuracies and sudden jumps in the height estimate when flying above the NIST crates or their side walls. The DJI M210v2 provides a magnetometer-based orientation, a horizontal position from VIO and an ultrasonic height measurement above ground. Together, the UAV's odometry has an RMS-ATE of 0.56 m including clear deviations in horizontal directions.

We already filter out spikes with high vertical velocities that are inconsistent with the motion direction. Moreover, our system only includes relative motion and no absolute position. However, smaller steps from objects on the ground still impact the result. Comparing the measured height with the local map height can potentially reduce the discrepancy, but remains out of the scope of this work.

If we instead supply only a single relative orientation measurement ΔR per scan, we achieve an RMS-ATE of 0.066 m. According to Tab. 6.4, this variant would place seventh, directly behind Fast-LIO2. Hence, we can still obtain resonable results in the absence of an IMU or improve resilience in underconstrained environments.

6.3.3.2 *Convergence Basin*

Additionally, we explore the convergence basin of our registration in Fig. 6.11 for future optimization of loop-closure candidates. Naturally, the basin depends on the environment represented by the maps under registration. Self-repetitive environments exhibit more local minima, whereas close quarters shrink the basin w.r.t. translation.

Instead of comparing close scans, we evaluate the distributed keyframes with an overlap of at most 80%. For this, we register the 26 motion-compensated keyframes against their respective local map at keyframe creation time on the "Quad-M" sequence of the Newer College extension (Zhang et al., 2021). The LiDAR predominantly measures the ground and the four vertical surfaces of the quad.

This relative registration solely relies on LiDAR to optimize a single relative pose without a continuous-time trajectory or IMU information. We create a regular grid with a 1 m stepsize to offset the initial position in x- and y-direction by ± 3 m and vary roll, pitch, and yaw by up to $\pm 40°$ with a 10° stepsize. In a second experiment, we include the height offset (± 1 m) and increase the x-y-offset to ± 4 m, while only the yaw varies by up to $\pm 45°$ in 5 degree increments. The 45° yaw limit is in place due to the right-angled repetitive walls of the quad.

For easier visualization, we combine the offsets for the positions using the distance $\|\Delta \boldsymbol{p}\|$ and similarly for the angles $\|\mathrm{Log}\,(\Delta R)\|$. The registration is successful if the final pose is within 10 cm and 5° of the original pose. As expected from a local optimization, the success rate decreases with increasing distance and angular offset, whereby the orientation is more impactful than the distance. Optimizing first on a coarse level and subsequently introducing finer levels can potentially enlarge the convergence basin. Moreover, the basin is sufficient to register scans with varying offsets, e.g., after unrotating the ground plane (Gupta et al., 2025), and select the most probable result, for example, based on ray-tracing (Quenzel et al., 2024). However, this is unnecessary during regular operation, as the IMU and prior scan motion restrict the rotation offset well.

Table 6.7: Statistics for varying motion compensation on the "F2" sequence in the DRZ Living Lab. Lower values are better (↓) with third, second and **best** highlighted.

Comp.		De-Skew	Full	RMS-ATE	Comp.	Reg.	Avg. Time
Pre.	Sel.		Cov.	[m] (↓)	[ms] (↓)	[ms] (↓)	[ms] (↓)
			✓	0.0577	**0.0**	18.4	48.5
				0.0565	**0.0**	**16.1**	**42.0**
		✓	✓	0.0572	<0.1	18.2	48.0
		✓		0.0567	<0.1	**15.6**	37.8
✓			✓	0.0570	16.5	19.0	65.9
	✓		✓	0.0570	4.4	19.0	53.3
✓				0.0562	16.4	18.5	64.7
	✓			0.0562	4.4	19.0	52.9
✓		✓	✓	0.0569	18.2	18.7	67.1
	✓	✓	✓	0.0569	4.9	18.9	53.6
✓		✓		0.0561	18.5	17.4	60.0
✓*		✓		**0.0560**	10.5	16.2	49.2
	✓	✓		0.0561	4.2	16.6	43.4

* Compensate only the new scan.

6.3.4 *Motion Compensation*

Compensating all surfels within the sliding window prior to registration is a time-costly process. Hence, we compare against compensating only the adaptively selected surfels. Additionally, we test the effect of the de-skewing and using the full covariance versus the adaptation for planar surfels (Eq. 6.14). Table 6.7 highlights the benefit of adapting the covariances of planar surfels. Furthermore, compensating selected surfels is much faster without losing accuracy.

To show the effect of our proposed UT for motion compensation, we use the Kullback-Leibler divergence (KLD) (Hershey and Olsen, 2007):

$$D_{\mathrm{KL}}\left(\mathcal{N}_q \| \mathcal{N}_r\right) = \frac{1}{2}\left[\ln\frac{|\Sigma_r|}{|\Sigma_q|} + \mathrm{Tr}\left(\Sigma_r^{-1}\Sigma_q\right) - d + \left(\boldsymbol{\mu}_q - \boldsymbol{\mu}_r\right)^{\mathsf{T}}\Sigma_r^{-1}\left(\boldsymbol{\mu}_q - \boldsymbol{\mu}_r\right)\right]. \quad (6.138)$$

The reference distributions $\mathcal{N}_r$ stem from surfel mean $\boldsymbol{\mu}_s$ and covariance $\bar{\Sigma}_s$ of compensated points using Eq. 6.101 after registration. Per surfel $s \in \mathcal{S}_l$, we compute the KLD for the raw points ($\mathcal{N}_{q_0}$), with IMU pre-orientation ($\mathcal{N}_{q_1} = (\boldsymbol{\mu}_s, \Sigma_s)$) and after surfelwise compensation ($\mathcal{N}_{q_2} = (\bar{\boldsymbol{\mu}}_s, \bar{\Sigma}_s)$). Again, we evaluate on the rotation-heavy sequence "06_spin" of the Newer College dataset (Ramezani et al., 2020).

Table 6.8: Avg. Timing [ms] for GMM computation per scan for 5 iterations on seq. "F2". Lower values are better (↓) with second and **best** highlighted.

Method \Step	Assoc.	Eval.	Grad.
orig.	3.0975	4.6433	3.4701
w/ sym.	3.0462	2.7356	2.6307
w/ vec_L	**1.8741**	**2.1111**	**1.9326**

Compared to the surfels from raw points ($\mathcal{N}_{q_0}$), our compensation reduces the KLD w.r.t. $\mathcal{N}_r$ for 76.0 % of the surfels. On average, the median KLD improves per scan in 94.6 % of the cases, with a mean reduction to 21.4 %.

The benefit is less pronounced against the pre-oriented points. After our compensation, 53.4 % of the surfels have a lower KLD w.r.t. $\mathcal{N}_r$. On average, the median KLD improves per scan in 74.5 % of the cases with a mean reduction to 91.9 %. Interestingly, the covariances $\bar{\Sigma}_s$ are more similar in 65.4 % of the cases versus 57.1 % for the mean.

6.3.5 *Influence of Symmetry*

We compare the timing for the previous GMM (Sec. 5.2.2) against a variant exploiting symmetry, e.g., in inverse computation (Eq. 6.129), and finally, our vectorized version (Sec. 6.2.11). All versions run on sequence "F2" with up to four threads and five iterations during registration. In Tab. 6.8 we report the timing for the association, gradient computation, and evaluation separately, as the GMM computation is split between them. As expected, exploiting the symmetry accelerates evaluation and gradient computation. Our vec_L version outperforms both other variants and reduces the time spent on association.

We further evaluate the timing for our optimized splatting operation from Sec. 6.2.11 on 14.195×10^6 points distributed over an area of 200 m^2. Each point is splatted onto all four levels of our surfel map (Sec. 5.2.1) with only a single thread or up to four threads in parallel. We also check whether to iterate first over all points and the level second (P→L) or vice versa. Table 6.9 shows that our optimized implementation with a single thread provides a similar speedup to using four threads for regular splatting. Moreover, our solution still profits from the higher thread count and from first processing all levels per point (L→P).

6.3.6 *Further Qualitative Examples*

To further showcase the accuracy of registered and motion-compensated point clouds, we aggregate every 25th point cloud and colorize them by height as previously done for Fig. 6.6. Recently, we deployed our UAV during a forest fire training exercise by the fire brigade of the district Viersen at the abandoned Javelin Barracks in Elmpt, Germany. Figure 6.12 shows the height-colored point clouds from multiple flights at

Table 6.9: Timing [ms] for splatting of 14.195×10^6 points (P) on 4 levels (L). Lower values are better (↓) with second and **best** highlighted.

	single (Sec. 5.2.1)		parallel (Sec. 6.2.11)	
Threads \Order	L→ P	P→ L	L→ P	P→ L
1	3475	3488	**1053**	1092
4	1034	1124	**322**	479

separate locations. Each ≈ 30 m wide row house consists of 5 attached units in a state of disrepair with many windows and doors missing. During the exercise, the trainers set the bushes in front of the left houses in Fig. 6.12 a) and close to the tree in b) on fire. After the brush fire was extinguished, our D1-UAV inspected the scene and directly provided an overview for the firefighters to assess the extent of the burned vegetation. Figure 6.12 b) shows the reconstructed LiDAR map.

Quenzel et al. (2024) further processed the sequence to align the global navigation satellite system (GNSS) poses with the georeferenced environment models. In the following Ch. 7, we additionally combine these points with thermal and color images.

6.4 Summary

In this chapter, we presented a novel continuous-time LiDAR-inertial odometry called LIO-MARS and verified our key claims. Our system extends MARS (Ch. 5) by introducing a non-uniform B-spline, active motion compensation, and tight coupling of LiDAR and IMU.

The non-uniform continuous-time B-spline adapts better to variations in scan timing without introducing additional delays. For this, we propose a novel strategy for the temporal knot placement to better represent the sliding window used for state estimation at runtime. As a consequence, we achieve better numerical stability during optimization compared to its uniform counterpart.

A timewise separation into intra-scan segments facilitates motion compensation at optimization time. Meanwhile, an unscented transform compensates individual surfels, which leads to more concise covariances for measured surfaces.

Leveraging complementary motion estimates further improves consistency and robustness. For this, we derived the analytic Jacobians for relative motion constraints, like preintegrated IMU measurements or relative poses.

Our system generates data-dependent new keyframes from the motion-compensated scans and selects a subset of keyframes for local map generation based on a surfel-dependent overlap. Moreover, the local map scale adapts to the measured distances in the sensor's vicinity to better represent open or narrow environments.

We modified the embedding of points into the permutohedral lattice using an inverse permutation and parallel min-max-sorting coupled with efficient SIMD instructions. In this way, our improved embedding achieves a ≈ 3.3-fold increase in throughput without changing the thread count. Furthermore, we exploit the inherent symmetry within

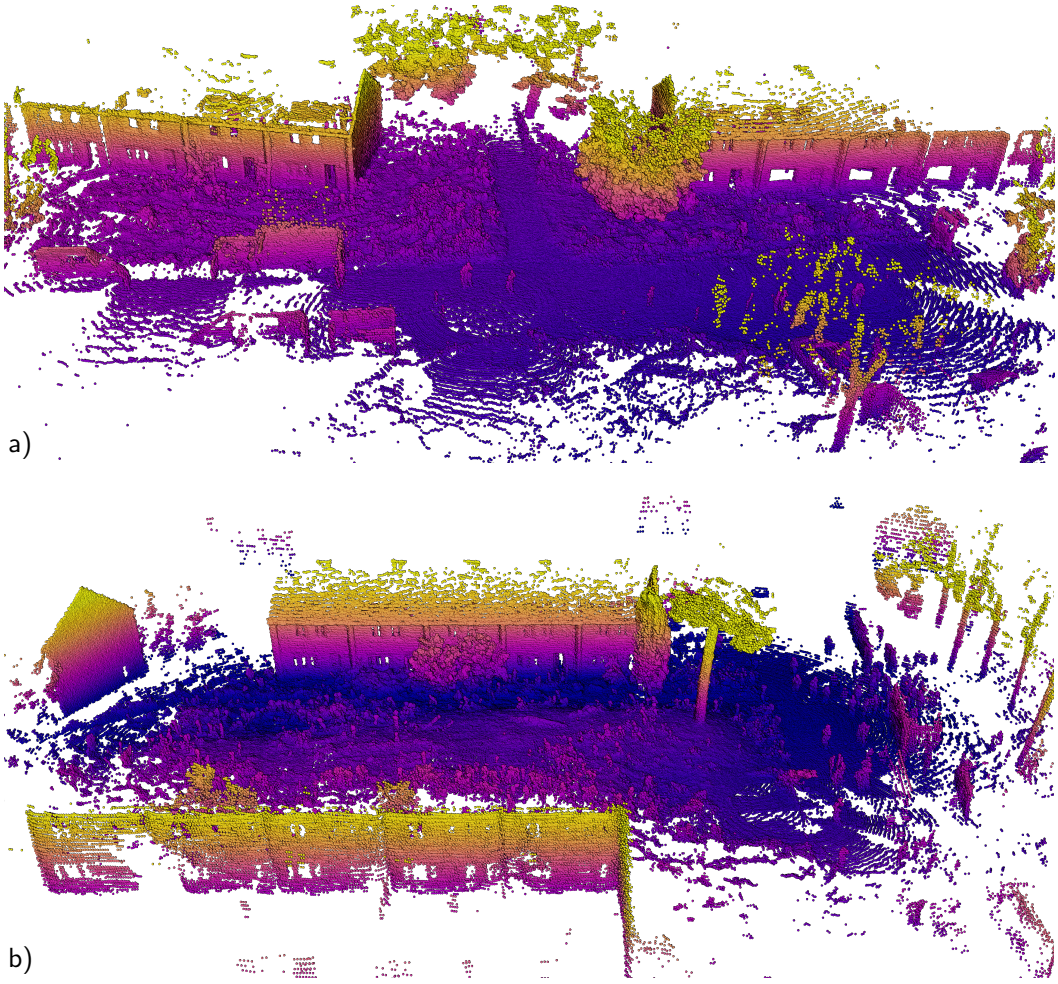

Figure 6.12: Aggregated pointclouds from separate locations recorded during a fire fighting exercise with the fire brigade of the district Viersen at the abandoned Javelin Barracks in Elmpt, Germany. The visible row houses consist of 5 attached units with a combined width of $\approx 30\,\mathrm{m}$. Maps are colored by height from low (blue) to high (yellow).

surfel covariances and the Gaussian mixture model computation and rephrase both using Kronecker sums and products for more efficient calculation. As a consequence, the real-time performance per iteration improves on average by a factor of 2.

In total, we tested 11 other current LO and LIO algorithms on multiple LiDAR datasets in UAV and handheld sensor scenarios. Overall, our system is the best-performing continuous-time method delivering state-of-the-art performance in real-time.

7

Dense Multi-Modal Mapping

Autonomous robotic systems heavily depend on knowledge about their environment to safely navigate, interact and perform inspection and search and rescue (SAR) tasks in real-time. For this, state estimation and mapping capabilities are key building blocks. Ideally, such a map contains the observed structures with their dimensions, color, thermal signatures (Rosu et al., 2019a), and the semantics to know what is visible (Behley et al., 2019; Rosu et al., 2019b; Bultmann et al., 2023). At the same time, an operator supervising the unmanned aerial vehicle (UAV) profits from improved situational awareness as the map depicts the whereabouts of the UAV and what is in the drone's vicinity, thus giving a better sense of the surrounding environment.

In this chapter, we will focus on the fusion of preprocessed multi-modal data for fine-grained dense mapping, as visualized in Fig. 7.1. Since LiDARs provide accurate distance measurements around the robot even in texture-less or dark environments independent of ambient illumination, we rely on LiDAR scans to generate the represented 3D structure. For this, the continuous-time trajectory of Ch. 6 enables accurate dense reconstructions to be enriched with color, thermal, and semantic information from multiple complementary modalities, such as LiDAR, RGB-D, and thermal cameras. Multi-rate systems benefit from such a late fusion approach, increasing adaptability to changing sensor configurations and enabling pipelining for efficient hardware usage.

In previous chapters, mapping is a means to improve the calibration and estimated state. As such, only a selected subset of the available information was used with a varying degree of density, from sparse triangulated features (see Ch. 3) to dense keyframes with coarse surfels (see Ch. 5). Consequently, the fine-grained dense map creation needs to be fast to provide valuable real-time feedback. Depending on the platform, we do not have the resources to run the fusion on the GPU, as the GPU already computes the semantics and object detection (Bultmann* et al., 2021, 2023). Other processes with real-time constraints also run onboard the UAV. Hence, we aimed for an efficient single/multi-core CPU implementation with sparse volumes. Instead of the lattice from Ch. 6, we adapt the sparse voxel grid from Ch. 5. The main reason is the faster index computation. Furthermore, we only require access to some neighboring voxels to map the occupancy.

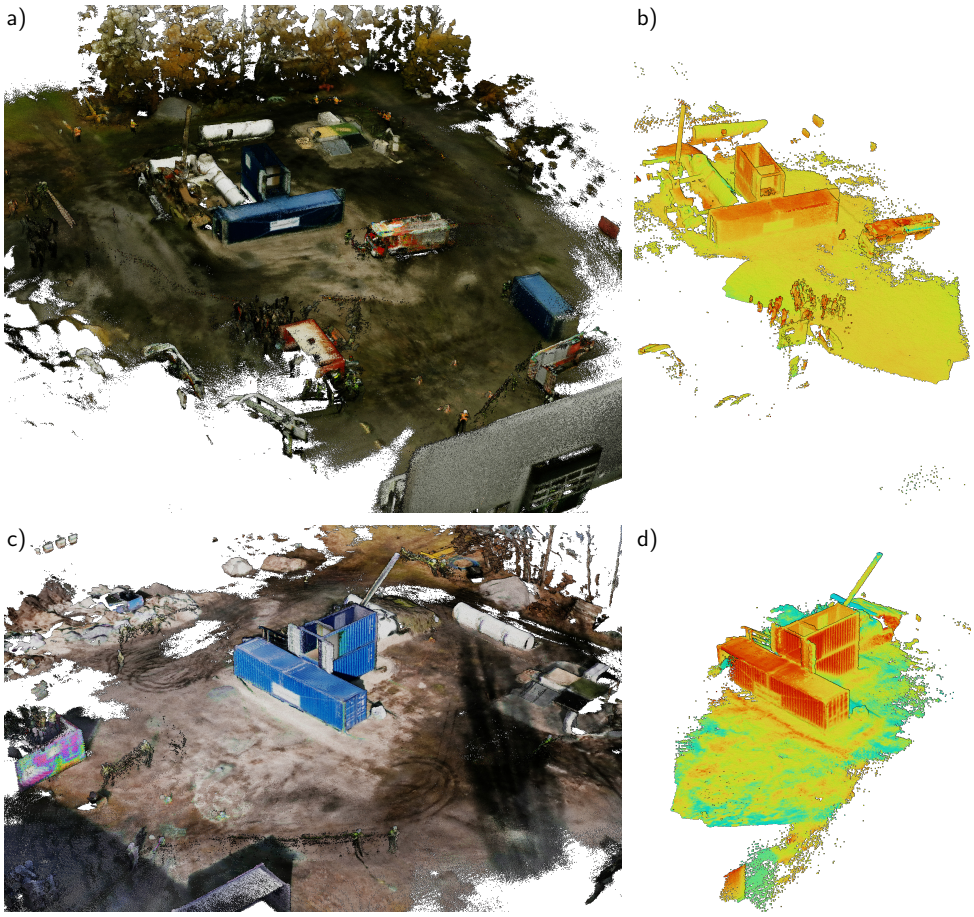

Figure 7.1: Colored and thermal maps acquired from two test runs of a collapsed building scenario at the DRZ Living Lab.

In summary, our key claims for our mapping approach are:

- First, the continuous-time trajectory allows us to integrate multi-modal information temporally correct over multiple views into a common map.
- Second, rephrasing the probabilistic semantic fusion in its log form improves numerical stability.
- Third, the proposed integrated system proved useful in a number of applications with real-world UAV experiments.

Preface

This chapter is partially adapted from Bultmann* et al. (2021), previously published by IEEE and presented at the European Conference on Mobile Robots (ECMR 2021), and its extension (Bultmann et al., 2023), previously published by Elsevier in the Robotics and Autonomous Systems journal.

Parts of this chapter are adapted from Schleich et al. (2021), previously published by IEEE and presented at the International Conference on Unmanned Aircraft Systems (ICUAS 2021).

Statement of Personal Contribution

> "The author of this thesis [Jan Quenzel] substantially contributed to the following aspects of the previous publication (Schleich et al., 2021), including the conception, design, and implementation of the proposed methods for LiDAR odometry and occupancy environment mapping, the preparation and conduct of experiments for the outdoor datasets, drafting the manuscript, and helped with the manuscript revision. He further contributed to the literature survey, evaluation of the proposed approach and the analysis and interpretation of the experimental results."

> "The author of this thesis [Jan Quenzel] substantially contributed to the following aspects of the previous publication (Bultmann* et al., 2021), including the conception, literature survey, design, and implementation of the proposed methods for point cloud fusion and semantic mapping, the preparation and conduct of experiments for the outdoor datasets, drafting the manuscript, and helped with the manuscript revision. He further contributed to the evaluation of the proposed approach and the analysis and interpretation of the experimental results."

> "The author of this thesis [Jan Quenzel] substantially contributed to the following aspects of the previous publication (Bultmann et al., 2023), including the conception, literature survey, design, and implementation of the proposed methods for point cloud fusion and semantic mapping, the preparation and conduct of experiments for the outdoor datasets, and helped with drafting the manuscript and manuscript revision. He further contributed to the evaluation of the proposed approach and the analysis and interpretation of the experimental results."

The content presented in this chapter, unless otherwise stated, is the contribution of the author of this thesis.

The proposed method for occupancy mapping was developed by the author of this thesis and integrated into the high-level planning by the co-author Daniel Schleich. Hence, Sec. 7.2.1 was adapted to match the thesis author's contributions, and a reference to the original publication (Schleich et al., 2021) is given for a detailed description of the implementation.

The methods for semantic segmentation and label propagation were provided by the co-author Simon Bultmann. Hence, Sec. 7.2.4 was significantly adapted to match the thesis author's contributions, and a reference to the original publications (Bultmann* et al., 2021, 2023) is given for a detailed description of the implementation.

7.1 Related Work

3D reconstruction has been extensively studied over the last two decades (Kostavelis and Gasteratos, 2015; Zollhöfer et al., 2018; Chen and Wang, 2024; Dalal et al., 2024).

Most approaches can be grouped based on their map representation and spatial subdivision into point-, voxel-, surfel-, mesh-based, and (neural-)implicit methods.

Point-based methods maintain a sparse set of individual surface measurements. Volume element (voxel)-based approaches subdivide the scene into volumetric elements, e.g., using regular grids (Oleynikova et al., 2017), a permutohedral lattice (see Sec. 5.2.1) or an octree (Hornung et al., 2013). Surface element (surfel)-based systems use either discs (Whelan et al., 2015; Behley and Stachniss, 2018; Park et al., 2018) or Gaussians (see Sec. 5.2; Kerbl et al. (2023)) to include measurement uncertainty in a probabilistic fashion. In contrast, mesh-based methods (Rosu et al., 2019b; Vizzo et al., 2021) represent surfaces as a set of polygons with texture applied to the polygon faces. Implicit models estimate a function whose zero-level set coincides with the surface, e.g., using signed distance functions (SDFs) (Oleynikova et al., 2017), Hermite radial basis functions (RBFs) (Liu et al., 2016) or learned features (Mildenhall et al., 2021; Zhong et al., 2023).

Each representation comes with its own advantages and drawbacks. Point clouds are easy to handle and can naturally express structures with differing levels of detail. Nevertheless, holes remain in between points while the amount of points quickly becomes difficult to process. Additionally, finding correspondences requires elaborate nearest neighbor (NN) searches, such as using a k-d tree (Blanco and Rai, 2014; Xu and Zhang, 2021). voxels directly reduce the total number of elements since a voxel represents all points within its volume. This reduction is generally fast and simplifies the scene at the cost of discretization errors. However, neighborhood searches are faster. Surfels follow the surface more closely with less discretization error while initially requiring more computation than voxels. Meshing (Kazhdan and Hoppe, 2013; Vizzo et al., 2021) is time-consuming and resource-intensive while naturally hole-filling. Fine structures require an increased number of polygon vertices. At the same time, the overlaying texture allows the decoupling of geometry and visual appearance (Rosu et al., 2019b), while modern graphics hardware enables fast rendering.

In practice, hybrid volumetric representations are very common, with map storage being (block-)sparse (Hornung et al., 2013; Oleynikova et al., 2017; Xu and Zhang, 2021; Yuan et al., 2022; Sec. 5.2.1) instead of dense. CT-ICP (Dellenbach et al., 2022) maintains a small number of actual points per voxel in a sparse voxel grid, which was subsequently adopted by KISS-ICP (Vizzo et al., 2023) and SE-LIO (Yuan et al., 2025). Droeschel and Behnke (2016) use dense voxel grids on multiple resolutions where a surfel represents the points per voxel. Similarly, we combine permutohedral lattices with surfels in Sec. 5.2.1. SHINE-mapping (Zhong et al., 2023) stores incrementally learned features in an octree and uses a shared multi-layer perceptron (MLP) to decode the SDF. Similarly, PIN-SLAM (Pan et al., 2024) uses point-wise latent features to represent the underlying geometry and predicts the SDF from multiple nearby voxel hashed points.

Occupancy Mapping

In order for robots to safely navigate within the environment, it is of utmost importance to differentiate between free space and occupied areas. Probabilistic modeling (Thrun et al., 2005) prevailed here as sensor measurements and pose estimates are inherently noisy. In 3D, octrees (Hornung et al., 2013; Vespa et al., 2018; Duberg and Jensfelt, 2020; Funk et al., 2021) commonly store the log-odds representation for a voxel's occupancy probability. Log-odds only require a summation instead of the

multiplication for the Bayes update. OctoMap (Hornung et al., 2013) has been widely adapted for robotic applications (Lluvia et al., 2021). Instead, OFusion (Vespa et al., 2018) proposes a more compact storage and efficient tree traversal using a Morton code. Additionally, a quadratic B-spline is used to model the range noise for RGB and depth (RGB-D) sensors. UFOMap (Duberg and Jensfelt, 2020) also employs a Morton code for traversal while focusing more on faster integration. Funk et al. (2021) build upon OFusion and introduce a more sensor-specific volume allocation. In contrast, Voxblox (Oleynikova et al., 2017) focuses on an efficient merging strategy and the creation of an Euclidean SDF (ESDF) from the underlying truncated SDF (TSDF) for planning. Recently, nvblox (Millane et al., 2024) adapts the approach for GPU, whereas Dynablox (Schmid et al., 2023) aims to detect moving objects within the TSDF map.

Reactivity is important in changing environments, especially in the presence of people or moving objects, and with planning for autonomous robots in mind. Sun et al. (2018) extend OctoMap with a long short-term memory (LSTM) per cell to account for long-term changes like parking cars. Reacting to short-term events requires some time to transition from occupied to free space. Moreover, adjusting the sensor model parameters towards a desired reaction time might not be straightforward. For this, OFusion (Vespa et al., 2018) introduced a time-based moving average to enable forgetting. Instead, we introduce a signum occupancy function, which allows us to maintain small time windows and fast computation.

Peopleremover (Schauer and Nüchter, 2018) generates a voxel grid from registered panoramic terrestrial LiDAR scanner (TLS) clouds. It then marks "see-through" voxels by ray-tracing the TLS points, starting with points closest to the sensor and omitting shadowed points. Points belonging to the marked voxels are identified as dynamic. In contrast, Removert (Kim and Kim, 2020) detects dynamic points based on discrepancies between range images of projected sub-maps with an iterative refinement over multiple resolutions. Erasor (Lim et al., 2021) restricts itself to an automotive scenario by assuming points above the ground plane are dynamic. After binning based on horizontal range and angle, it detects dynamic candidate bins by thresholding the height difference between the map and the query scan. For these candidates, Erasor computes the local ground plane and marks points above as moving. This poses a problem in indoor scenarios, with foliage or underpasses, as removing these points from the map may harm the localization quality.

While 3D semantic occupancy prediction (Zhang et al., 2023b; Wang et al., 2025) is related to semantic and occupancy mapping, the prediction and completion are outside the scope of this thesis.

Color Mapping

Individual map elements typically store additional information like color (Oleynikova et al., 2017), thermal (Rosu et al., 2019a), occupancy (Hornung et al., 2013), the signed distance (SDF) to the closest surface (Daun et al., 2021; Splietker and Behnke, 2023), semantic (McCormac et al., 2017), a density (Kerbl et al., 2023), latent (Yuan and Nüchter, 2024) or learned features (Zhong et al., 2023; Pan et al., 2024).

The TSDF remained popular (Zollhöfer et al., 2018) for 3D reconstruction with RGB-D or colorized point clouds. Minimizing the signed distance is convenient for

registration, while the marching cubes algorithm (Lorensen and Cline, 1987) directly extracts a mesh for visualization. TSDF methods (Dai et al., 2017) are predominantly GPU-based to achieve real-time performance. The limited GPU memory restricted the scene size (Newcombe et al., 2011a) or required shifting (Whelan et al., 2013) of the dense volume. Nießner et al. (2013) tackled this with sparse voxel hashing to maintain voxels only around measured points.

However, Steinbrücker et al. (2014) instead proposed an efficient CPU-based TSDF update using SIMD instructions within an octree structure. Voxblox (Oleynikova et al., 2017) incrementally constructs the Euclidean SDF on the CPU to establish synergies between mapping and planning. Recently, nvblox (Millane et al., 2024) adapts Voxblox to NVIDIA GPUs for faster computation due to higher parallelization while also providing an occupancy integrator.

Over the last four years, neural field approaches (Xia and Xue, 2023; Dalal et al., 2024; Irshad et al., 2024) have received much attention motivated by the image-based task of novel view synthesis. Neural radiance field (NeRF) (Mildenhall et al., 2021) established radiance field methods as the state-of-the-art for high-fidelity reconstructions. It trains a shallow MLP to predict the color and density for a given 3D point and viewing direction. NeRF renders a novel view by sampling points along each pixel's ray direction and alpha-blending the predicted colors using the density. Although querying the MLP is fast, NeRF requires very long training times ranging typically from a couple of hours up to multiple days. Plenoxels (Fridovich-Keil et al., 2022) and DVGO (Sun et al., 2022) showed voxel-based radiance fields without NNs that provide similar quality within minutes. Instant-NGP (Müller et al., 2022) dramatically reduced NeRF's training time using a multi-resolution hash encoding on the order of minutes rather than hours. For this, Instant-NGP maintains a multiscale occupancy grid derived from the NeRF density estimate to focus on sampling close to the surface during training. SiLVR (Tao et al., 2024) conditions an Instant-NGP NeRF (Müller et al., 2022) on depth from LiDAR to obtain sub-maps. Neuralangelo (Li et al., 2023) adapts the multi-resolution hash encoding to estimate a neural SDF in a coarse-to-fine manner.

As an alternative to MLP's, Gaussian splatting (GS) (Kerbl et al., 2023) combines unordered 3D Gaussians with radiance fields where individual Gaussians are alpha-blended. Follow-up works proposed using 2D surfels (Huang et al., 2024), Gaussian opacity fields (Yu et al., 2024) or smooth convex shapes (Held et al., 2025). However, these methods require potent compute capabilities.

Thermal Mapping

Incorporating other modalities like thermal or multi-spectral cameras enables a broad range of inspection and mapping applications, e.g., from solar panels (Meribout et al., 2023) and building inspection (Kim et al., 2023; Parracho et al., 2023), to landslide (Sun et al., 2024) or volcanic mapping (Irmisch et al., 2021) and archeological surveying of heritage sides (Sutherland et al., 2023).

A typical workflow for UAV-based thermography (Parracho et al., 2023), building information model (BIM) generation, or building inspection involves pre-planning a GNSS-based flight path, automatic image capturing during the flight and post-processing of color and thermal images after landing with structure-from-motion (SfM)

and multi-view stereo (MVS) or orthophoto generation (OpenDroneMap Authors, 2020; Kim et al., 2023). Such an SfM-based workflow generally processes color imagery first before integrating thermal images since the reconstruction accuracy benefits from the higher resolution of color cameras.

Archeological surveys (Sutherland et al., 2023) mainly rely on TLSs to generate dense 3D point clouds or meshes with thermal images applied as texture. Similarly, Rosu et al. (2019a) reconstruct a mesh (Kazhdan and Hoppe, 2013) from registered LiDAR scans acquired by a UAV. Afterwards, projecting mesh faces into the thermal images allows the merging of different viewpoints in a high-resolution thermal texture on coarse geometry. The textured mesh further enables the detection and localization of heat sources.

Irmisch et al. (2021) combine a thermal camera with an inertial measurement unit (IMU), a GPS, and stereo color cameras for volcanic mapping and BIM generation (Schischmanow et al., 2022). Here, a voxel grid merges the 3D point clouds from dense stereo matching of the color cameras with projected thermal information. The system triggers the reconstruction in regular time intervals or based on the distance traveled according to visual-inertial odometry (VIO) or SfM. Semantic object masks help to remove moving objects during offline processing.

Recent works (Hassan et al., 2025; Lu et al., 2025) transfer radiance field approaches like NeRF and GS to thermal data. Ye et al. (2024) adapt Nerfacto (Tancik et al., 2023) for thermal-only NeRF with normalized intensities and structural thermal constraints. In contrast, Hassan et al. (2025) predict the temperature directly from features of the density MLP shared across modalities. Lin et al. (2024b) use a separate thermal density and mutually regularize the respective densities across channels. Like these NeRF-based approaches, Lu et al. (2025) evaluate multiple strategies supervised with an additional smoothness term for GS. This includes fine-tuning of thermal Gaussians after initialization from color, separate Gaussians per modality and adding separate spherical harmonics on a single Gaussian. Instead, Chen et al. (2024a) present a physically motivated adaptation for GS. An MLP modifies the spherical harmonics to model atmospheric attenuation during transmission. A learned module based on the 2D-Laplacian directly addresses blurring due to thermal conduction. As before, these methods are unsuitable for reconstruction in real-time.

Semantic Mapping

Environment knowledge helps in diverse tasks through a more complete scene understanding. Nguyen et al. (2019) perform semantic segmentation onboard a UAV to inspect penstocks. Bartolomei et al. (2020) classify an occupancy map and sparse VIO landmarks as (un-/)informative to steer path planning towards well-textured directions. Furthermore, semantic predictions improve registration (Bao and Savarese, 2011; Zaganidis et al., 2018) or allow to exclude dynamic objects for robustness (Chen et al., 2019). Long-term semantic correspondences (Lianos et al., 2018) help to reduce drift as the semantic notion of an object, e.g., a car, remains consistent over a wide variety of conditions where visual features become too dissimilar. A semantic map further enables domain adaptation from one modality to another, e.g., from RGB camera to LiDAR (Rosu et al., 2019b), between sensors of the same modality with

differing properties or transfer knowledge between different datasets (Bultmann et al., 2023).

Identifying instances of items (Civera et al., 2011; McCormac et al., 2018; Grinvald et al., 2019) allows one to find points of interest and reconstruct individual objects (Kong et al., 2023; Wen et al., 2023; Liao et al., 2024). This makes parts of the map reusable, as different instances of the same object can share and complete a single representation (Salas-Moreno et al., 2013). While object-level mapping and shape completion are beneficial in many robotic applications, these topics remain outside the scope of this thesis.

Semantic segmentation of point clouds has advanced tremendously in recent years since the introduction of benchmark datasets like SemanticKITTI (Behley et al., 2019), nuScenes (Caesar et al., 2020) or Paris-CARLA-3D (Deschaud et al., 2021). However, labels may change depending on the view angle for partially observed structures or due to increasing sparsity at higher ranges (Zhu et al., 2022a). Hence, they require recomputation as soon as more data is available.

In practice, we measure the environment continuously and have to enforce temporal consistency. Hermans et al. (2014) use a conditional random field (CRF) for spatial regularization to smooth semantic labels throughout an aggregated point cloud. Kundu et al. (2014) jointly infer occupancy and semantic category from sparse Visual-SLAM (VSLAM) points using a CRF for a dense voxel map.

Instead, Stückler et al. (2014) aggregate frame-wise semantics of RGB-D images towards keyframes in an octree on the CPU. A Bayesian update fuses voxel-wise class predictions using the depth from the RGB-D sensor, which results in higher accuracy for the back-projected labels compared to instantaneous segmentation. Yang et al. (2017) improve consistency after incremental Bayes update using a hierarchical CRF model on the 3D voxel grid.

Valentin et al. (2013) extract a mesh from a TSDF of aggregated depth images. Their classifier then infers the semantics per mesh face, which directly ties the semantic and geometric resolution together. A CRF regularizes neighboring faces without considering the per-class likelihood or newer information.

Alternatively, Kimera-Semantic (Rosinol et al., 2020) approximates the log-class probabilities to Voxblox (Oleynikova et al., 2017) by fusing the frequency of the class labels for points falling into the same voxel. Instead, SemanticFusion (McCormac et al., 2017) maintains the full class probabilities for each surfel on the GPU, and the Bayesian update fuses semantics into visible surfels. It tends to replicate surfaces on multiple scales when measured from different ranges, thus wasting limited GPU memory and prohibiting large-scale reconstruction.

Rosu et al. (2019b) decouple geometry and semantics with a textured mesh. The texture stores the class probabilities only for the most likely classes, with further weights taking the sensor distance and viewing angle into account. In this way, a geometrically simple wall can have a higher-resolution semantic texture to distinguish between the wall and an attached poster, or the opposite case where a facade with intricate geometry has a simple semantic texture. Moreover, adding additional texture channels, e.g., for thermal mapping (Rosu et al., 2019a), is straightforward once the mesh exists.

Our CPU-based real-time mapping combines multiple modalities in a common surfel map by adding modality-specific channels per surfel. The continuous-time trajectory (see Ch. 6) enables the augmentation of point clouds with temporally close color and thermal information (see Sec. 7.2.3). We reuse the surfel map with sparse voxel grids (see Ch. 5) with our efficient surfel aggregation (see Ch. 6). For semantics, we rephrase the Bayesian update of SemanticFusion (McCormac et al., 2017) in its log form for improved numerical stability. A signum occupancy function allows the integration of an efficient occupancy estimation with a fixed horizon for adjustable reactivity and allows for simple integer arithmetic.

7.2 Our Method

Our mapping builds upon the previously presented map representations in Ch. 5. We maintain a sparse voxel map $\mathcal{M}$ with a single resolution and small cell size for a higher level of detail. Again, each voxel stores a surfel to fuse points in its vicinity. The surfel is beneficial for occupancy mapping in Sec. 7.2.1 and easy to visualize using splatting (Pfister et al., 2000; Botsch et al., 2005; Kerbl et al., 2023). Furthermore, each voxel fuses additional application-specific information such as color (see Sec. 7.2.2), thermal (see Sec. 7.2.3), and semantics (see Sec. 7.2.4).

In the previous chapters, the keyframe creation naturally restricted the number of scans in the map. In contrast, we densely aggregate as many scans as possible here, even though the occupancy mapping may discard intermediate scans to satisfy real-time constraints. The newest scan keeps the map up-to-date, which is important for obstacle avoidance and planning (Schleich et al., 2021).

7.2.1 *Occupancy Mapping*

We define the occupancy of a voxel as either occupied (w_{occ}), free (w_{free}), or unknown (w_{unk}). In Schleich et al. (2021), we simplify the textbook case (Thrun et al., 2005) for efficiency reasons to the thresholded values as a signum occupancy function:

$$w_{\text{occ}} = +1, \tag{7.1}$$

$$w_{\text{unk}} = 0, \tag{7.2}$$

$$w_{\text{free}} = -1. \tag{7.3}$$

We maintain per voxel a ring buffer $\mathcal{W}$ of the last $n_{\mathcal{W}} = 16$ occupancy measurements (w_{occ}, w_{unk}, w_{free}), initialized as unknown. Such a small fixed-size ring buffer reduces the time needed to measure an old obstacle location as free. Hence, the sum over $\mathcal{W}$ gives the occupancy f_{occ} as:

$$s_w = \sum_{w \in \mathcal{W}} w, \tag{7.4}$$

$$f_{\text{occ}}(\mathcal{W}) = \begin{cases} w_{\text{occ}} & \text{, if } s_w \geq \theta_{\text{occ}} \\ w_{\text{free}} & \text{, if } s_w \leq \theta_{\text{free}} \\ w_{\text{unk}} & \text{, else.} \end{cases} \tag{7.5}$$

We first transform the point cloud $\mathcal{P}$ with the sensor pose into the map frame. Then, binning retains the set of indices $\mathcal{I}_{\text{occ}} \subset \mathbb{N}^3$ for all occupied voxels with a predefined side length. We maintain per voxel a surfel (see Ch. 5) for the binned points. Using the index $\boldsymbol{i} \in \mathbb{N}^3$ has the advantage of requiring only integer arithmetic. Additionally, we create sets of indices for free $\mathcal{I}_{\text{free}}$ and unknown voxels $\mathcal{I}_{\text{unk}}$. Then, we ray-trace with a 3D version of the Bresenham line search (Amanatides and Woo, 1987). The traversal starts at the index corresponding to the sensor position $\boldsymbol{o}_{\text{w}}$ and ends n_{unk} voxels behind the binned index. We add indices between the sensor and binned voxel to the free set $\mathcal{I}_{\text{free}}$ and those behind to the unknown set $\mathcal{I}_{\text{unk}}$. Due to the binning, some new voxels of $\mathcal{I}_{\text{occ}}$ may end up being traversed as either free or unknown. In that case, we remove $\boldsymbol{i} \in \mathcal{I}_{\text{occ}}$ from $\mathcal{I}_{\text{free}}$ and $\mathcal{I}_{\text{unk}}$. Similarly, removing $\mathcal{I}_{\text{free}}$ from $\mathcal{I}_{\text{unk}}$ ensures precedence of free space over unknown.

Measurements under a grazing angle often traverse many voxels and incorrectly classify them as free. Hence, we ignore free space elements of I_{free} if the dot product between surfel normal and view direction is below a threshold $\theta_\alpha = 0.3$. For all indices, new occupancy measurements are added to the corresponding voxel's ring buffer $\mathcal{W}$ according to the relevant set ($\mathcal{I}_{\text{occ}}$, $\mathcal{I}_{\text{free}}$, or $\mathcal{I}_{\text{unk}}$). We recompute f_{occ} and check if the updated voxel transitions its occupancy state to clear transitioned surfels of previously occupied and newly free/unknown voxels. Then, the map $\mathcal{M}$ merges the current scan's surfels corresponding to the occupied voxels of $\mathcal{I}_{\text{occ}}$ with their previously mapped counterpart.

We use this approach as the basis for high-level obstacle avoidance and trajectory planning (Schleich et al., 2021), with a voxel size of 25 cm. Similarly, Bultmann and Behnke (2022) adapt it to update dynamic objects within a semantic map fused from distributed smart edge sensors.

The following section details how additional color information enriches our dense map.

7.2.2 *Color Mapping*

Our continuous-time trajectory $T_{\mathcal{X}}(t)$ (see Sec. 6.2.1) readily provides the sensor poses at a points' capture time $t_{\boldsymbol{p}}$ and at the camera time t_{cam}. There, both poses are w.r.t. the common map frame w. Additionally, we address the time offset Δt_{cam} between the camera sensors and the LiDAR. This directly improves the quality of the fused colors and semantics as these heavily depend on the accuracy of sensor poses for projection (see Sec. 7.4 in Rosu et al. (2019b)).

For wide field-of-view (FoV) cameras, we employ the double sphere camera model (DS) (Usenko et al., 2018) with parameters $[f_x, f_y, c_x, c_y, \xi, \alpha]^\intercal$:

$$d_1 = \sqrt{p_x^2 + p_y^2 + p_z^2}, \tag{7.6}$$

$$d_2 = \sqrt{p_x^2 + p_y^2 + (\xi d_1 + p_z)^2}, \tag{7.7}$$

$$\pi_{\text{DS}}(\boldsymbol{p}) = \pi \left(\begin{bmatrix} p_x \\ p_y \\ \alpha \cdot d_2 + (1-\alpha) \cdot (\xi d_1 + p_z) \end{bmatrix} \right). \tag{7.8}$$

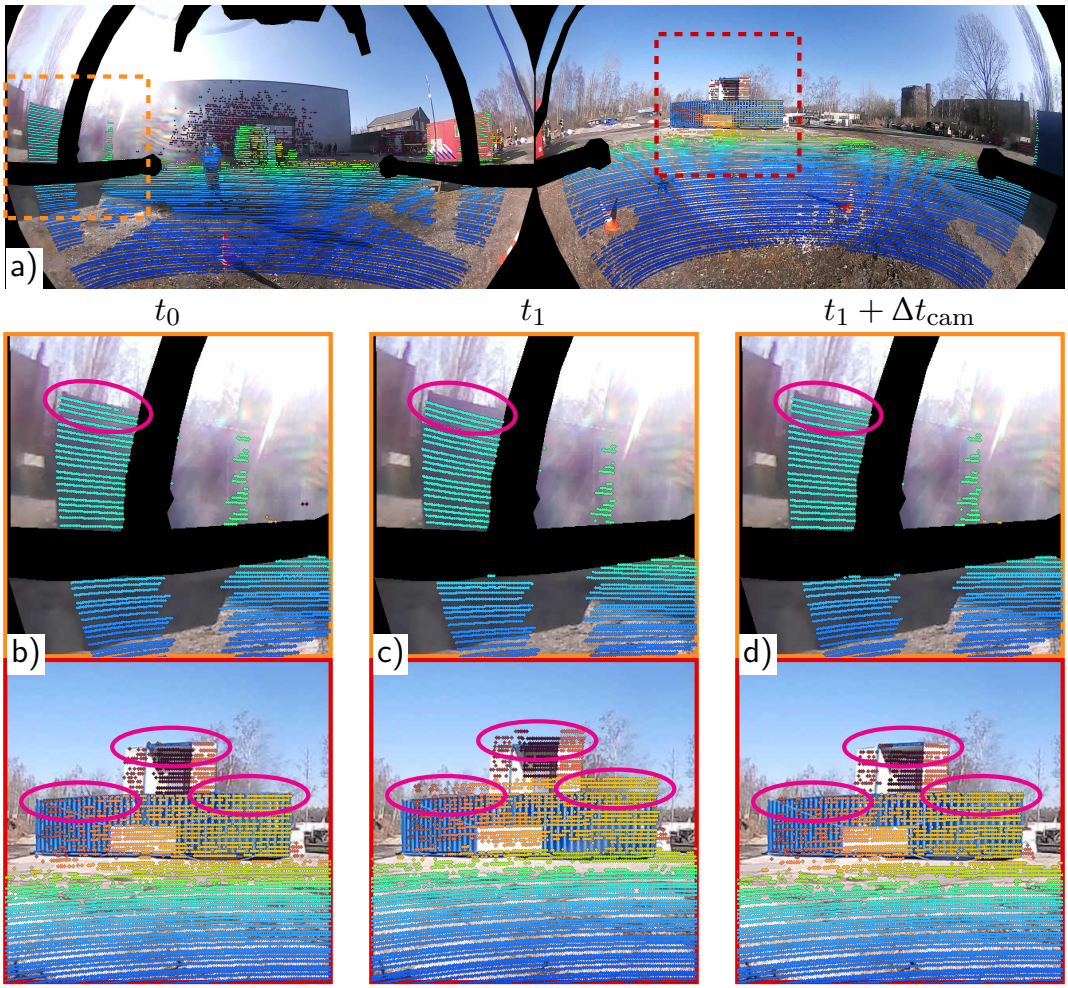

Figure 7.2: Projecting LiDAR points into the camera [a)] allows us to colorize the point cloud. For improper synchronized sensors, ego-motion leads to inconsistent projections [b),c)]. Ellipses highlight the discrepancies in the close ups. The correspondences remain consistent after compensation [d)] with the correct timing offset Δt_{cam}.

Fortunately, the pinhole camera model (see Sec. 2.5) is a special case of the DS model with $\xi = 0$ and $\alpha = 0$.

Together with both sensor poses, this allows us to define the following projection for the cameras in a unified manner:

$$\boldsymbol{p}_{\text{w}} = T_{\mathcal{X}}\left(t_{\boldsymbol{p}}\right) \cdot \boldsymbol{p}, \tag{7.9}$$

$$\boldsymbol{p}_{\text{c}} = T_{\text{c,i}} T_{\mathcal{X}}\left(t_{\boldsymbol{p}} + \Delta t_{\text{cam}}\right)^{-1} \cdot \boldsymbol{p}_{\text{w}}, \tag{7.10}$$

$$\boldsymbol{u}_{\boldsymbol{p}} = \pi_{\text{DS}}\left(\boldsymbol{p}_{\text{c}}\right). \tag{7.11}$$

In contrast to Rosu et al. (2019b), we only project the current point cloud into timewise close color or thermal images, as shown in Fig. 7.2 and Fig. 7.3. In this way, the viewpoints of the camera and LiDAR are similar, which reduces occlusions. Hence, the shadow mapping-based visibility check becomes unnecessary.

Depending on the target application, we require a colored map, e.g., for visual inspection. After projection (see Eq. 7.11), we bilinearly interpolate at the projected

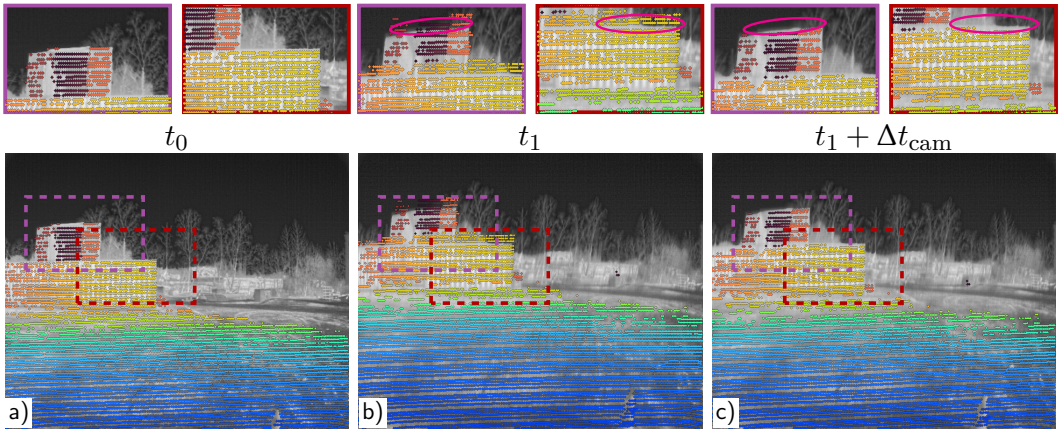

Figure 7.3: Motion leads to incorrect projections from a previous frame at t_0 [a)] to the next at t_1 [b)] without timing offset compensation. Measurements on the foreground, e.g., the container, project to the background, e.g., into the sky as highlighted by ellipses in the closeup. With offset Δt_{cam} [c)], the projection remains consistent.

pixel. As common knowledge, directly mixing RGB colors leads to suboptimal results (Kahu et al., 2019). Hence, we convert the color $\boldsymbol{c}$ into HSV color space using OpenCV[1]. We reuse the incremental update from Ch. 5 with Eq. 5.2 and Eq. 5.4 to fuse the weighted mean color with weight w_c in hue, saturation, value (HSV) color space:

$$w_s = w_s + w_c, \tag{7.12}$$

$$\overline{\boldsymbol{s}} = \overline{\boldsymbol{s}}_c + w_c \cdot \boldsymbol{c}_{\text{HSV}}, \tag{7.13}$$

$$\overline{\boldsymbol{c}}_{\text{HSV}} = \frac{1}{w_s} \overline{\boldsymbol{s}}_c. \tag{7.14}$$

The weight w_c (Rosu et al., 2019b) takes the distance and the viewing angle into account:

$$w_c = w_{\text{dist}} \cdot w_{\text{view}}, \tag{7.15}$$

$$w_{\text{dist}} = 1 - \frac{\|\boldsymbol{p}_c\|_2}{r_{\text{max}}}, \tag{7.16}$$

$$w_{\text{view}} = |(\boldsymbol{o}_{\text{w}} - \boldsymbol{p}_{\text{w}}) \cdot \boldsymbol{n}|. \tag{7.17}$$

Here, w_{dist} reduces linearly with increasing distance to the camera. This favors spatially closer frames and improves the details of the fused colors. The term w_{view} prefers frontal perspectives over oblique viewing directions.

For visualization, we convert the mean HSV color back to red, green, blue (RGB).

7.2.3 *Thermal Mapping*

Finding heat sources is a recurring task in SAR, as well as inspection. For this, thermal imagers are the tool of choice. However, the measurement accuracy strongly depends on the incidence angle on a surveyed surface. Under an oblique angle, a surface may

1 https://docs.opencv.org/4.9.0/de/d25/imgproc_color_conversions.html

reflect the heat signature from the environment. Nonetheless, measuring perpendicular to the surface can lead to reflections of the camera system or the person holding the camera. Low emissivity and high reflectivity of a surface intensify this problem. Hence, inspectors should survey under a small incidence angle, non-perpendicular to the surface (FLIR Systems Inc., 2016).

Following Rosu et al. (2019a), we model this behavior as:

$$w_{\text{thermal}} = w_{\text{view}} \cdot \left(1 - \exp\left(-\frac{1}{2\sigma^2} \arccos\left(w_{\text{view}}\right)^2\right)\right), \tag{7.18}$$

with $\sigma = 0.05$. Here, the dot product in w_{view} naturally decreases towards oblique angles, whereas the Gaussian downweighs nearly perpendicular observations. Apart from that, we reuse the projection (see Eq. 2.78, see Fig. 7.3) and weighted sum (see Eq. 7.14) to compute the mean thermal intensity.

As shown in Fig. 7.1, we apply a color map such as Turbo[2] to the mean thermal intensity for visualization purposes.

7.2.4 *Semantic Mapping*

Many tasks require knowledge about the semantic meaning of objects or surfaces in the environment. The robot should recognize the obstacles' whereabouts in the scene and understand whether those obstacles are cars, pedestrians, walls, or other objects. The semantic knowledge may stem from a projection of the point cloud into pixelwise segmented RGB images, from object detections (Bultmann* et al., 2021), directly from LiDAR segmentation (Cortinhal et al., 2020; Rosu et al., 2020), or a combination thereof (Bultmann et al., 2023).

Our input is a semantically segmented point cloud $\mathcal{Y}_k$ with per class probability $p\left(l_i|\mathcal{Y}_k\right)$ and semantic class label l_i. Each voxel fuses the semantics of all points in its vicinity probabilistically. Our probabilistic fusion scheme follows the reasoning of SemanticFusion (McCormac et al., 2017). Assuming independence between semantic segmentations $p\left(l_i|\mathcal{Y}_k\right)$, we use Bayes' Rule:

$$p(l_i|\mathcal{Y}_{1:k}) = \frac{p\left(l_i|\mathcal{Y}_{1:k-1}\right) p\left(l_i|\mathcal{Y}_k\right)}{\sum_i p\left(l_i|\mathcal{Y}_{1:k-1}\right) p\left(l_i|\mathcal{Y}_k\right)}. \tag{7.19}$$

A naive implementation, as in SemanticFusion, suffers from numerical instability due to the finite precision of the multiplication result. In practice, this leads to all class probabilities being close to zero, e.g., when class probabilities strongly vary:

$$p\left(l_i|\mathcal{Y}_k\right) \approx 1 \text{ and } p\left(l_i|\mathcal{Y}_{k+1}\right) \approx 0, \tag{7.20}$$

$$p\left(l_j|\mathcal{Y}_k\right) \approx 0 \text{ and } p\left(l_j|\mathcal{Y}_{k+1}\right) \approx 1, \tag{7.21}$$

such that both class-wise products will be almost zero. This results in a loss of information, which the normalization term can not recover. Thus, SemanticFusion required continuous reinitialization of the mapped probabilities $p(l_i|\mathcal{Y}_{1:k})$.

2 https://blog.research.google/2019/08/turbo-improved-rainbow-colormap-for.html

Hence, we switch to log probabilities:

$$L_{i,1:k} = \log\left(p\left(l_i|\mathcal{Y}_{1:k}\right)\right), \tag{7.22}$$
$$L_{i,1:k-1} = \log\left(p\left(l_i|\mathcal{Y}_{1:k-1}\right)\right), \tag{7.23}$$
$$L_{i,k} = \log\left(p\left(l_i|\mathcal{Y}_{k}\right)\right), \tag{7.24}$$
$$S_{1:k} = \log\left(\sum_i p\left(l_i|\mathcal{Y}_{1:k-1}\right) p\left(l_i|\mathcal{Y}_{k}\right)\right). \tag{7.25}$$

Each voxel now stores $L_{i,1:k}$ instead of $p\left(l_i|\mathcal{Y}_{1:k}\right)$. Equation 7.19 simplifies in log form to:

$$S_{i,1:k} = L_{i,1:k-1} + L_{i,k}, \tag{7.26}$$
$$L_{i,1:k} = S_{i,1:k} - S_{1:k}. \tag{7.27}$$

We further improve numerical stability with the following logarithm identity for $\log\left(\sum_i x_i\right)$ by factorizing the largest summand (x_m) out:

$$m = \arg\max_i(x_i), \tag{7.28}$$
$$\log\left(\sum_i x_i\right) = \log\left(x_m\right) + \log\left(1 + \sum_{i\neq m} \frac{x_i}{x_m}\right). \tag{7.29}$$

Many programming languages[3] provide for $\log(1+a)$ with $a \gtrapprox 0$ more accurate implementations via $\mathrm{log1p}(a)$ without explicitly applying the logarithm to the sum $(1+a)$.

Replacing the division in Eq. 7.29 with its log form results in:

$$\log\left(\sum_i x_i\right) = \log\left(x_m\right) + \log\left(1 + \sum_{i\neq m} \exp^{\log(x_i)-\log(x_m)}\right). \tag{7.30}$$

Hence, we compute $S_{1:k}$ as follows:

$$S_{1:k} = S_{m,1:k} + \log\left(1 + \sum_{i\neq m} \exp^{S_{i,1:k}-S_{m,1:k}}\right). \tag{7.31}$$

As common in segmentation, the last network layer uses a softmax activation function to normalize the predicted distribution. Ideally, we would skip the last layer and directly fuse unnormalized network outputs to save additional exp and log computations. However, this step is necessary since the individual outputs may be arbitrarily scaled. As an additional precaution, our co-author Simon Bultmann introduced a temperature parameter (LeCun et al., 2006) in the softmax computation for the journal version (Bultmann et al., 2023). The temperature adjusts the sharpness of the networks' output distribution, either shifting towards a single peak in $(p(l_i|\mathcal{Y}_k) \approx 1)$ or increasing uniformity. While we observed the described numerical

3 https://numpy.org/doc/stable/reference/generated/numpy.log1p.html, https://en.cppreference.com/w/cpp/numeric/math/log1p, https://de.mathworks.com/help/matlab/ref/log1p.html

instability (see Eq. 7.21) in Bultmann* et al. (2021) and with SemanticFusion in Rosu et al. (2019b), we did not encounter the instability with the temperature softmax.

An infinite time horizon of the semantic map, fusing all scans, may not be necessary or wanted—depending on the use case, e.g., for global vs. local planning. Hence, we store the log probabilities per scan in a fixed-size double-ended queue (deque) per voxel, as described in Ch. 5. Fusing these per-scan log probabilities yields the voxels' class probabilities for the shorter time horizon with at most N scans. Older scans are either fused into the infinite time horizon estimate or removed entirely.

Most points have a high likelihood for a limited number of classes. Hence, it is unnecessary to store all class probabilities. For memory constraint systems, it is beneficial to retain the most likely class and probability $\max_i p(l_i|\mathcal{Y}_k)$ per point. As a result, only a sparse subset requires storage. For more details on that approach, we refer the reader to Rosu et al. (2019b).

7.3 Evaluation

We evaluate the mapping components on data captured during multiple flights with modified DJI M210v2 UAVs. The UAV is equipped with an Intel NUC for onboard processing and data storage. An Ouster OS0-128 LiDAR provides distance measurements for assistance functions, mapping, and higher-level autonomy. A FLIR Boson thermal camera takes long-wave infrared images in front of the drone for thermal mapping and person detection. Additionally, a 360° FoV Insta360 Air camera allows us to colorize all LiDAR points.

For the first sequence, multiple pedestrians walk around the Campus Poppelsdorf of the University of Bonn. At the same time, the operator manually steers the UAV from in between three buildings to an open field. Two more sequences were captured during autonomous exploration tests at the DRZ Living Lab in Dortmund. The scenario recreated a collapsed building with debris and rubble. During these exercises, MARS (see Sec. 5) localized the UAV and provided poses for the online occupancy mapping. The resulting occupancy was the basis for the planning pipeline to realize the autonomous flights. Schleich et al. (2021) describe the UAV setup and its capabilities in more detail.

For the following evaluation, we reprocess each sequence with LIO-MARS (see Ch. 6) and pass the motion-compensated point cloud to the mapping component. At first, we examine the occupancy mapping (see Sec. 7.3.1). Then, we jointly evaluate the color and thermal mapping (see Sec. 7.3.2) before continuing with the semantic mapping (see Sec. 7.3.3).

All experiments were conducted on a laptop with an AMD Ryzen 7 5800H and 48 GB of random access memory (RAM). The mapping components are restricted to use at most four threads to reduce interference at runtime with other components like our LiDAR-inertial odometry (LIO) or when deployed on a robot with planning and obstacle avoidance.

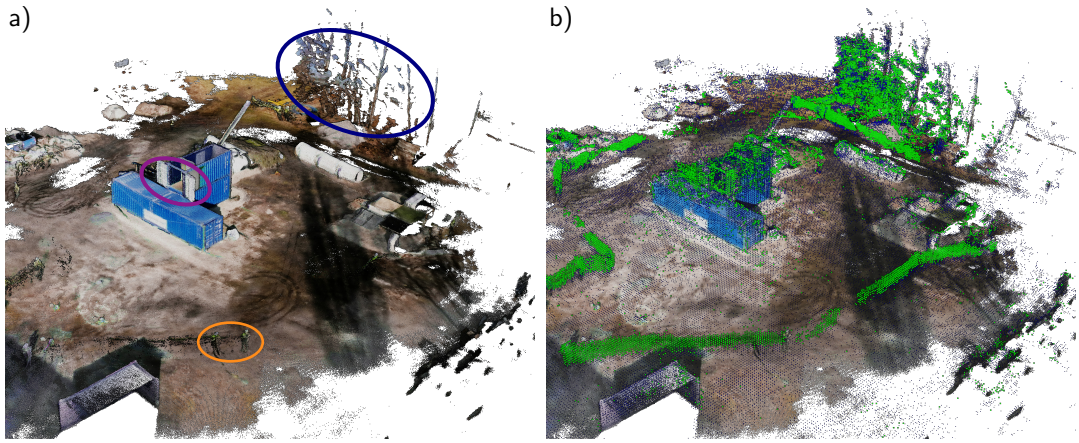

Figure 7.4: Colored map [a)] overlayed in [b)] with freed voxels (green) during a test run in a collapsed building scenario at the DRZ Living Lab. Freed voxels mostly correspond to trees (blue), the swinging container doors (violet) and the safety-operator (orange) along his path.

7.3.1 *Occupancy Mapping*

We evaluate our occupancy on the first sequence captured at the Campus Poppelsdorf of the University of Bonn. For this, we compare the occupancy against manually annotated semantic ground-truth classes from the aggregated and voxelized point cloud (25 cm) after registration with MARS. Here, the occupancy cell size is set to 25 cm to match the coarse semantic resolution and the cell size of the high-level planning (Schleich et al., 2021). Since the "person" class is the only moving class, we check for each "person" voxel if it was marked as occupied and freed later on. Using the intersection over union (IoU) would be inaccurate, as the "person" class represents only a small subset of all free voxels. After processing the whole dataset, the occupancy mapping marked 7822 out of 7840 "person"-voxels (99.77 %) as free again. The remaining 18 voxels mostly correspond to current measurements of people. The exception is one pedestrian moving in the opposite direction of the UAV and leaving the sensor range. These remain occupied since we only ray-trace valid measurements for runtime efficiency.

In Schleich et al. (2021), we use the occupancy mapping for high-level planning during numerous test flights at the DRZ Living Lab and on the Campus Poppelsdorf. Since the occupancy maps were not recorded during these exercises, we rerun the occupancy mapping using the motion-compensated points and poses from LIO-MARS for two autonomous test runs at the DRZ Living Lab. Here, our occupancy computation switches to a cell size of 20 cm for easier subdivision, which equals 4 times the color map's resolution (5 cm). For better visibility, we overlay previously occupied and subsequently freed voxels (green) on the colored map from Sec. 7.3.2.

The safety pilot's path stands out in Fig. 7.4 as multiple elongated green segments around the scenario site. The path appears non-continuous due to the limited sensor range. Additionally, due to windy conditions, many freed voxels group around the free-swinging container doors and the tree branches in the back.

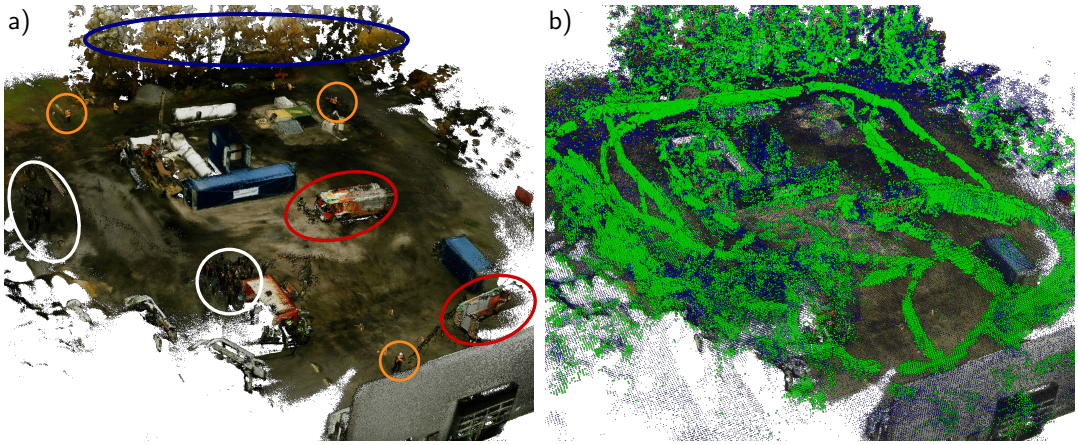

Figure 7.5: Colored map [a)] overlayed in [b)] with freed voxels (green) during a joint exercise with fire fighters (red) in a collapsed building scenario at the DRZ Living Lab. Freed voxels stem from multiple fire fighters including a fire truck and the safety pilot (orange). A group of observers (white) moved from the container to the left during the exercise.

For this sequence, adding newly measured occupancy voxels took 2.41 ms whereas ray tracing took, on average, 9.22 ms. Inserting free measurements took another 3.53 ms. Afterwards, merging new voxels into the map took 4.55 ms, including the normal computation. In total, a scan took, on average, around 19.93 ms using up to four threads, resp. 24.80 ms with a single thread. Without grazing angle rejection and normal computation, the time further reduces to 13.29 ms for four threads or 20.00 ms with a single thread.

For timing comparison, we supply the same point clouds to the popular voxblox (Oleynikova et al., 2017). Its successor, nvblox (Millane et al., 2024), requires a GPU and is thus omitted since the GPU might not be available or is preoccupied with object detection or semantic computation. Furthermore, OctoMap is excluded since its reportedly[4] slower than voxblox by a factor of 2.

Voxblox's integration method is set to "fast" with their iterative closest point (ICP) being deactivated for fairness. We exclude mesh updating and publishing from the analysis. Voxblox (Oleynikova et al., 2017) takes at a resolution of 20 cm with carving enabled around 21.26 ms compared to our 19.93 ms resp. 13.29 ms.

The second run exhibits much more movement, as shown in Fig. 7.5. Multiple firefighters participated in the exercise while a group of observer was present at the red container. The firefighter truck started on the right and turned left to the center. As before, the safety pilot followed the UAV around the compound. During the second half, the observer group moved to the left side behind fluttering barrier tape.

7.3.2 *Color and Thermal Mapping*

In order to demonstrate the color improvement due to correct time projection, we compute per surfel the fused colors' mean $\boldsymbol{\mu}_c$ and covariance Σ_c.

4 https://voxblox.readthedocs.io/en/latest/pages/Performance.html

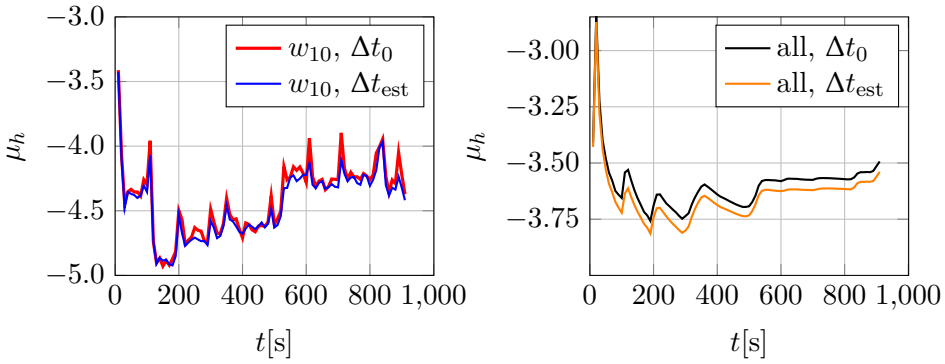

Figure 7.6: Incorrect projection time offset (Δt_0) increases the mean differential entropy μ_h for the color covariance of mapped surfels. The correct offset Δt_{est} provides lower entropy over non-overlapping 10 s windows w_{10} (left) and over the whole sequence (right).

We evaluate the differential entropy $h(\mathcal{N})$ as in Cover and Thomas (2005) (Eq. 8.43) for the fused color Gaussian $\mathcal{N}_c(\boldsymbol{\mu}_c, \Sigma_c)$ and the mean value μ_h over all surfels $\mathcal{C}$:

$$h(\mathcal{N}_c(\boldsymbol{\mu}_c, \Sigma_c)) = \frac{1}{2} \ln(2\pi e)^3 \det \Sigma_c, \tag{7.32}$$

$$\mu_h = \sum_{c \in \mathcal{C}} h(\mathcal{N}_c). \tag{7.33}$$

A lower differential entropy corresponds to less uncertainty and less variation in the fused colors. To get an estimate over all surfels, we compute the mean value μ_h over $h(\mathcal{N}_c)$ for the uncorrected $\Delta t = 0$ and the corrected Δt_{est} for non-overlapping 10 s windows and a complete flight at the DRZ Living Lab. The resulting mean differential entropy μ_h, as shown in Fig. 7.6, underlines that compensating for time offsets reduces variations in fused colors. Furthermore, this emphasizes the benefit of our continuous-time trajectory for multi-rate systems with differing acquisition times. This helps to obtain higher-quality reconstructions. Figure 7.1 shows the resulting color and thermal maps with closeups of the scenario structure in Fig. 7.7.

Adding color and thermal measurements to the point cloud takes around 1.28 ms. At a resolution of 5 cm, surfeling and weighting the augmented cloud takes another 3.51 ms for up to four threads, resp. 9.32 ms for a single thread. Merging the cloud's surfels with the map and updating the changed map takes another 12.13 ms, resp. 10.85 ms. In total, the fusion procedure for the augmented point cloud takes around 15.63 ms using up to 4 threads or 20.17 ms for a single thread.

In comparison, Voxblox still takes 25.78 ms on the same 5 cm resolution with disabled carving and using a restricted number of 4 threads. Again, this excludes updating and publishing of the mesh.

Additionally, we deployed our UAV during a forest fire training exercise by the fire brigade of the district Viersen. During the exercise, the fire brigade extinguished multiple controlled vegetation fires at the abandoned Javelin Barracks in Elmpt, Germany. Figure 7.8 shows one such fire's color and thermal map, which we recorded directly after the flames went out.

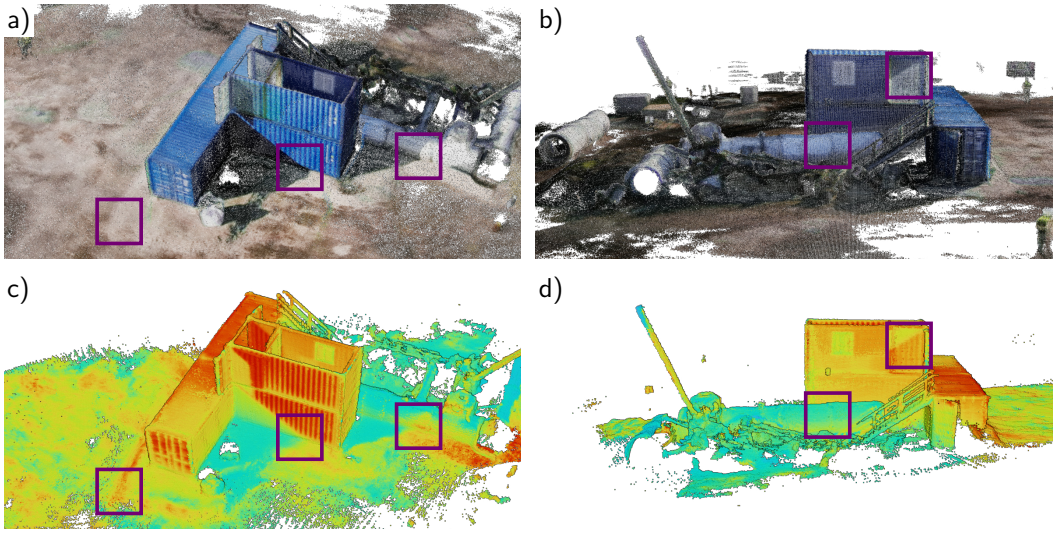

Figure 7.7: Closeups of colored and thermal maps of second sequence at the DRZ Living Lab. Shadowed and brightly lid areas (rectangles) in the color map are clearly discernable in thermal maps.

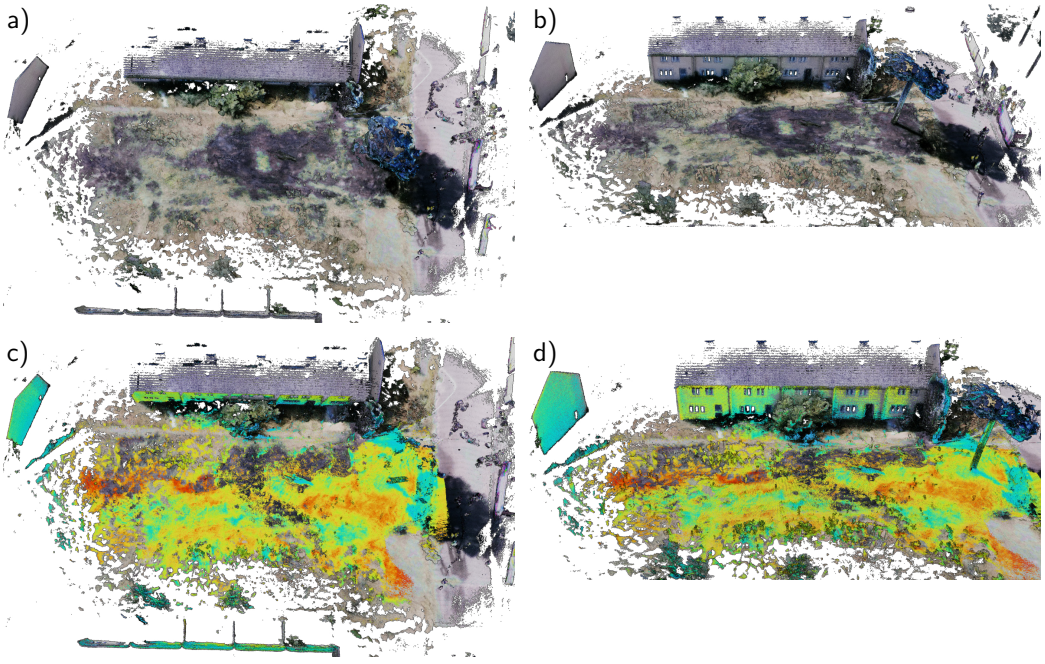

Figure 7.8: Colored and thermal maps of forest fire training exercise by the fire brigade of the district Viersen at the abandoned Javelin Barracks in Elmpt, Germany.

Table 7.1: Comparison of the IoU [%] for semantic labels of single LiDAR scans and our fused map for different variants of a retrained SalsaNext (Cortinhal et al., 2020). Higher values are better (↑) with second and **best** highlighted.

Method	building	road	person	vegetation	bicycle	vehicle	object	Mean IoU
Single	32.39	72.99	2.67	2.17	3.93	0.60	3.40	16.88
Map	23.58	66.94	0.25	0.45	0.00	1.40	3.65	13.75
Single w/ z, I	83.15	75.55	16.90	2.12	7.47	2.87	5.51	27.65
Map w/ z, I	86.66	65.08	27.28	4.62	0.21	5.84	10.05	28.53
Single w/ LP, I	95.82	88.71	49.45	77.58	14.40	56.19	22.77	57.85
Map w/ LP, I	96.14	**89.05**	53.25	**80.39**	20.92	81.31	**44.23**	66.47
Single w/ LP	95.80	88.26	62.54	74.54	14.57	60.70	26.47	60.41
Map w/ LP	**96.35**	88.51	**73.49**	78.85	**22.85**	**82.51**	43.96	**69.50**

7.3.3 *Semantic Mapping*

We evaluate our semantic fusion against the manually annotated ground-truth classes on the first sequence captured at the Campus Poppelsdorf of the University of Bonn. Here, SalsaNext (Cortinhal et al., 2020) provides the semantic annotations for the LiDAR scans and runs in real-time onboard the UAV. Cortinhal et al. trained SalsaNext on SemanticKITTI (Behley et al., 2019) with differing LiDAR, FoV, and sensor geometry. Hence, the network required adaptation to our setup as described in Bultmann et al. (2023) by rescaling the height (w/ z), inclusion of the intensity channel (w/ I), or retraining with label propagation (w/ LP). For more details, we refer the reader to the corresponding publication.

Table 7.1 shows the resulting IoU for single scans and the fused maps with the proposed scheme (see Sec. 7.2.4). All numbers have been recalculated after several bug fixes and may deviate slightly from the initially reported results in Bultmann* et al. (2021) and Bultmann et al. (2023). We keep the coarse and fine voxel sizes (25 cm, resp. 6.25 cm) consistent with the initial results for better comparison, even if these sizes differ from the previous sections. Originally, the size was chosen for efficiency reasons to coincide with the occupancy resolution used by the high-level planning.

Our fusion promotes consistency between scans, which ideally improves the mean IoU. However, this also reinforces incorrect decisions as seen for the original unadapted SalsaNext. After adaptation, all maps exhibit higher mean IoU than single scan

Table 7.2: Ablation on Mean IoU for SalsaNext. Second and **best** values are highlighted.

SalsaNext	Resolution [cm]	Mean IoU [%] (↑)	Avg. Time [ms] (↓)
orig.	25	12.07	43.7
orig.	6.25	13.75	60.8
w/ z, I	25	23.36	**32.7**
w/ z, I	6.25	28.53	54.6
w/ LP, I	25	69.46	33.4
w/ LP, I	6.25	66.47	55.2
w/ LP	25	**69.81**	47.7
w/ LP	6.25	69.50	55.4

evaluation. Here, the smaller classes, such as “person” or “bicycle”, benefit more than “building” or “road”.

Table 7.2 shows the mean IoU and average computation time for coarse and fine voxel resolutions (25 cm, resp. 6.25 cm) with our logarithmic fusion formulation (see Eq. 7.27). The mean IoU between different fusion schemes (see Eq. 7.19 and Eq. 7.27) was insignificant for the retrained networks due to the temperature adjustment within the softmax activation function. As expected, the fusion takes longer for the finer resolution as more voxels are in the sparse volume. Nevertheless, all variants run in real time using a single thread with insignificant differences after LP retraining w.r.t. the resolution.

The consistent estimate of the scene semantics enables us to enforce temporal and spatial consistency between individual class predictions (Rosu et al., 2019b; Bultmann et al., 2023). The fused label is useful for fine-tuning (Rosu et al., 2019b) for novel viewpoints or the domain adaptation between different sensors and modalities (Bultmann et al., 2023). Although designed for use with LiDAR, our formulation is general enough for use with depth from RGB-D cameras. As such, Bultmann and Behnke (2022) adapt our semantic map to a distributed smart edge sensor network.

7.4 Summary

In this chapter, we presented a multi-modal fusion for CPU-based dense environment mapping in real-time and verified our key claims.

Our sparse voxel map stores per voxel a surfel, the voxel's occupancy, its color, the thermal signature, and the voxel's semantics. We reuse the vectorized representation and computation from Sec. 6.2.11 for the surfel. A ternary signum occupancy function allows fast reactive occupancy estimation per voxel using efficient summation over a small fixed-size ring buffer.

The continuous-time trajectory, estimated by LIO-MARS (see Ch. 6), enables us to aggregate multi-modal sensor data temporally correct in a common map even within a multi-rate system. Using a motion-compensated image projection, we enrich point clouds with additional channels from temporally close multi-modal images. Our fusion further integrates the respective thermal and color channels into the map under consideration of the view dependency of the respective sensors. As a result, our fast CPU-based multi-view aggregation provides more consistent maps at runtime while keeping the GPU free for other purposes like semantic segmentation and object detection. To merge semantic estimates, we reformulate the probabilistic fusion using Bayes' rule in its logarithmic form for improved numerical stability. As a result, fused semantic labels enforce temporal and spatial consistency between individual class predictions. In addition, these consistent labels enable fine-tuning on novel viewpoints and domain adaptation between different sensors and modalities (Bultmann et al., 2023).

During real-world experiments, our color and thermal maps provided direct feedback for the UAV operator during teleoperation and allowed monitoring of autonomous missions. Moreover, the mapped occupancy enabled high-level planning and autonomous exploration in dynamic environments (Schleich et al., 2021), as validated in numerous autonomous flights.

8 Conclusion

8.1 Summary

In this thesis, we presented novel real-time approaches for calibration, odometry, and mapping using multi-modal sensor setups onboard various UAVs. Our solutions laid the foundation for autonomous assistance functions, including obstacle avoidance, navigation, and exploration, that fostered safer manual and repeatedly enabled autonomous drone operations (Schleich et al., 2021). All methods work in close proximity to structures and are independent of external positioning like global navigation satellite system (GNSS) or preexisting and possibly outdated maps. As such, our created maps facilitate inspection (Quenzel et al., 2019) and SAR while improving decision-making for further downstream tasks. Moreover, the presented methods are applicable for general dense 3D mapping and localization with, e.g., car-, robot-mounted (Ch. 5), or handheld sensor suites (Ch. 6).

Our joint photometric calibration (Ch. 3) successfully estimates the camera's vignetting, response function CRF and exposure ratio together with the map point radiance. It handles natural scenes without uniform illumination or known calibration targets while running in real-time on a laptop CPU. Small oriented patches around the sparsely distributed ORB features form the basis for the keyframe-based optimization without requiring multiple per-pixel measurements. For vignetting, a 2D thin plate spline (TPS) captures local deformations, whilst a sixth-order polynomial represents the general shape. Similarly, we model the camera response function (CRF) using a combination of a 1D TPS with border conditions and a Gamma curve. Furthermore, the mapped radiance enables estimation of the current exposure ratio w.r.t. the tracked map at the sensor's frame rate.

Inspired by the necessity for constant brightness in computer vision, our novel dissimilarity metric (Ch. 4) replaces the standard photometric error in direct image alignment. Already slight variations in lighting drastically degrade the performance of the photometric error. In contrast, using our measure leads to more accurate and robust depth in stereo disparity tasks as well as improved camera trajectories. Our metric combines a gradient-based orientation measure with a magnitude-dependent scaling term, which is easy to integrate into various existing visual odometry (VO) systems while retaining real-time performance at typical camera frame rates.

To localize the UAV in flight, a novel LiDAR odometry, called MARS (Ch. 5), jointly registers multiple scans along a continuous-time B-spline trajectory in real-time without requiring prior maps. Moreover, our MARS delivered reliable UAV poses during numerous autonomous test flights with state-of-the-art accuracy on various datasets, including handheld and automotive setups. We accelerate the GMM-based registration by embedding scans in multi-resolution surfel maps with sparse permutohedral lattices and adaptively selecting the appropriate surfel resolution.

For high-fidelity reconstructions, our extension LIO-MARS (Ch. 6) directly optimizes intra-scan motion during registration without costly pointwise surfel reintegration due to UT-based de-skewing. Resilience increases further by leveraging timewise non-uniformly spaced B-spline knots and relative motion constraints on the continuous-time trajectory, e.g., from robot odometry or preintegrated IMU measurements. Real-time performance improves by rephrasing the Gaussian mixture model (GMM) and covariances with Kronecker sum and products.

Despite the multi-modal sensors' differing acquisition times, we enrich LiDAR point clouds pointwise with color and thermal signatures from time-compensated projective correspondences (Ch. 7) while considering their reliability w.r.t. scene geometry. Our real-time mapping combines the annotated scans, including semantic categories, into dense 3D maps using sparse volumes. The semantic fusion ensures greater numerical stability by rephrasing the Bayesian fusion in its logarithmic form.

For high-level planning in dynamic environments, we improve reactivity with a signum occupancy function to maintain a small temporal occupancy window in real-time. Moreover, occupancy computation becomes more efficient due to simple integer summation. The window length balances reactivity to changes detected by ray-tracing and retaining previous knowledge of the environment.

Keeping a situational overview updated during a crisis is a major challenge for first responders. A task that is nowadays unimaginable without UAVs. With the contributions presented throughout this thesis, UAV operation becomes easier than ever while delivering broad overviews together with closeup inspection for difficult-to-reach and GNSS-denied environments in real-time.

8.2 Outlook and Future Work

The proposed works naturally present numerous possible future research directions.

Minimizing our dissimilarity metric, e.g., between the image and the LiDAR's intensity or reflectivity channel or reprojection residuals for image lines and 3D edges, could enable joint calibration of extrinsic parameters and time-offsets for color and thermal cameras. These additional parameters may also be directly introduced into the LIO optimization. Compensating rolling shutter effects in real-time is another possibility.

Combining camera and LiDAR for localization (Zheng et al., 2022) balances and complements the individual sensors' strengths and weaknesses. On an open field, the LiDAR may only measure the ground, whereas the camera retains visible landmarks (Shan et al., 2021) beyond the LiDAR's range. Dark environments let color cameras struggle, while thermal and LiDAR work well. For overlapping FoV, LiDAR points may initialize image features with reliable depth or guide temporal MVS to even denser maps with increased fidelity in the reconstruction.

Scene motion violates the static world assumption inherent in most odometry systems. Explicitly incorporating the relative motion of tracked objects (Judd et al., 2018) may improve resilience and allow for more accurate localization in otherwise underconstrained situations.

Although the scanning pattern of some LiDARs, like the Livox Horizon, is non-repetitive, some surfaces are sampled more than once during the scanning time.

This would lead to additional intra-segment constraints per scan. However, the time estimation for sigma points in the proposed unscented transform (UT) does not support these patterns. Further LiDAR-only constraints arise from intensity and reflectivity channels (Zhang et al., 2023a; Pfreundschuh et al., 2024), where different surfaces and materials appear discernable. However, both channels require compensation (Höfle and Pfeifer, 2007) for scene geometry and vignetting similar to cameras.

LiDARs directly provide distance measurements over a broad range with low relative error. Diffusion models presented first promising results for denoising (Chang et al., 2023; Nakashima and Kurazume, 2024) yet remain too time-consuming and costly for real-time application in actual SAR scenarios.

Furthermore, the noise becomes more pronounced when dealing with shorter measurements. Conversely, range noise dominates the surfel covariance when the surfel size is too small, or points sample a surface in an unfavorable pattern, e.g., a straight line. In both cases, the estimated normal direction is perpendicular to the actual surface normal. This affects registration as well as mapping. Extending our surfel-based registration with range-image (Di Giammarino et al., 2022; Qu et al., 2022) or mean-based point registration (Xu et al., 2022; Vizzo et al., 2023) provides a suitable fallback strategy for these adverse surfels or for very high ranges with too sparse measurements.

Supplementing our surfel maps with other environment features may help to model the underlying geometry more accurately. Spheroids or cylinders fit better to partially observed round objects like lamp posts (Dong et al., 2023) or tree trunks (Proudman et al., 2022). Edges directly constrain two directions, but curvature-based detection on raw point clouds is unreliable (Xu et al., 2022). Normal-based detection between neighboring surfels is a promising alternative. Similarly, knowing where a wall ends or a window begins helps in otherwise unrestricted scenarios, e.g., free-standing buildings measured from only one side.

In multi-story buildings and staircases, LiDARs commonly measure corresponding floors and ceilings or walls from both sides. Improper handling tends to contract both mapped surfaces into one (Chung and Kim, 2024). Under the reasonable assumption of uniform thickness for individual walls, e.g., due to building codes and materials (DIN 4109-2:2018-01), knowledge of one side allows us to assume the opposite orientation and possibly estimate the wall thickness as well.

An iterated error-state Kalman filter (IEKF) is a popular tool (Geneva et al., 2020; Qin et al., 2020; Xu and Zhang, 2021; He et al., 2023) in modern state estimation due to its accuracy and the maintained uncertainty estimate. However, the IEKF requires specialized derivation for spline-based trajectory estimation. The uncertainty may increase resilience when dealing with unrestricted or uninformative directions during optimization or resolve such later on.

Further extensions for LiDAR-inertial odometry with MARS maps (LIO-MARS) include loop-closing and pose graph optimization (PGO) to improve consistency and reduce accumulated errors. PGO could directly optimize the whole spline trajectory (Quenzel et al., 2024) and align the per-scan estimated gravity globally.

Recent advances promise vast potential in learned implicit map representations (Hong et al., 2024; Pan et al., 2024; Tao et al., 2024). These could tackle

some of the shortcomings of point-, surfel- or TSDF-based maps and enable joint optimization of map and trajectory. Currently, learned methods (Mildenhall et al., 2021; Kerbl et al., 2023) are too computationally involved or time-consuming which is disadvantageous for robotic systems with limited resources.

However, a straightforward extension of our mapping (Ch. 7) is the subsequent use of Gaussian splatting (GS) (Kerbl et al., 2023; Jiang et al., 2025) on an operator's station for more visually pleasing maps. We already have accurate and meaningful 3D Gaussians with channel-wise information, e.g., color, which is an ideal initialization for GS and should allow much faster convergence. Furthermore, the operator may directly inspect the map while incremental optimization on some selected keyframes continues in parallel.

A

Incorporated Publications

A.1 Keyframe-based Photometric Online Calibration and Color Correction

This publication constitues the main part on color calibration in Ch. 3.

> The publication has been removed from the online version of the thesis and can be accessed via the DOI link above.

A.2 Beyond Photometric Consistency: Gradient-based Dissimilarity for Improving Visual Odometry and Stereo Matching

This publication constitues the core part on gradient-based dissimilarity in Ch. 4.

> The publication has been removed from the online version of the thesis and can be accessed via the DOI link above.

A.3 Real-time Multi-Adaptive-Resolution-Surfel 6D LiDAR Odometry using Continuous-time Trajectory Optimization

This publication constitues the core part on LiDAR odometry in Ch. 5 and lays the foundation for Ch. 6.

 "Real-time Multi-Adaptive-Resolution-Surfel 6D LiDAR Odometry using Continuous-time Trajectory Optimization." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS51168.2021.9636763.

> The publication has been removed from the online version of the thesis and can be accessed via the DOI link above.

A.4 Autonomous Flight in Unknown GNSS-denied Environments for Disaster Examination

This publication covers parts on the occupancy mapping in Ch. 7.

> The publication has been removed from the online version of the thesis and can be accessed via the DOI link above.

A.5 Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles

This publication covers parts of Ch. 7.

DOI: 10.1109/ECMR50962.2021.9568812.

> The publication has been removed from the online version of the thesis and can be accessed via the DOI link above.

A.6 Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles with Label Propagation for Cross-Domain Adaptation

This publication covers parts of Ch. 7, extending the conference version with approaches for label propagation for cross-domain supervision and additional experiments.

First published in *Journal of Robotics and Autonomous Systems*, vol. 159, 2023, by Elsevier. Reproduced with permission from Elsevier. Reprinted from S. Bultmann, J. Quenzel, and S. Behnke (2023). "Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles with Label Propagation for Cross-Domain Adaptation." In: *Journal of Robotics and Autonomous Systems* 159. DOI: 10.1016/j.robot.2022.104286.

> The publication has been removed from the online version of the thesis and can be accessed via the DOI link above.

B

Additional Derivations

B.1 Preintegration

The Jacobians for Eq. 6.38 are as follows:

$$\overline{\boldsymbol{a}}_{\mathrm{w}} = -R_{\mathrm{w}}\overline{\boldsymbol{a}}_{\mathrm{m}_{j+1}}\Delta t_{\mathrm{m}}, \tag{B.1}$$

$$\boldsymbol{J}_{\Delta_{\mathrm{pre}_j}}^{\Delta_{\mathrm{pre}_{j+1}}} = \begin{bmatrix} I_3 & [\overline{\boldsymbol{a}}_{\mathrm{w}}\frac{\Delta t_{\mathrm{m}}}{2}]_\times & \Delta t_{\mathrm{m}} I_3 \\ 0_3 & I_3 & 0_3 \\ 0_3 & [\overline{\boldsymbol{a}}_{\mathrm{w}}]_\times & I_3 \end{bmatrix}, \tag{B.2}$$

$$\boldsymbol{J}_{\overline{\boldsymbol{a}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}} = \begin{bmatrix} \frac{\Delta t_{\mathrm{m}}^2}{2} R_{\mathrm{w}} \\ 0_3 \\ \Delta t_{\mathrm{m}} R_{\mathrm{w}} \end{bmatrix}, \tag{B.3}$$

$$\boldsymbol{J}_{\overline{\boldsymbol{\omega}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}} = \begin{bmatrix} \frac{\Delta t_{\mathrm{m}}^2}{4}[\overline{\boldsymbol{a}}_{\mathrm{w}}]_\times R_{\mathrm{w}} J_r\left(\frac{\Delta t_{\mathrm{m}}}{2}\boldsymbol{\omega}_{\mathrm{m}_{j+1}}\right) \\ \Delta t_m \Delta R_{\mathrm{m}_j} J_r\left(\Delta t_{\mathrm{m}}\boldsymbol{\omega}_{\mathrm{m}_{j+1}}\right) \\ \frac{\Delta t_{\mathrm{m}}}{2}[\overline{\boldsymbol{a}}_{\mathrm{w}}]_\times R_{\mathrm{w}} J_r\left(\frac{\Delta t_{\mathrm{m}}}{2}\boldsymbol{\omega}_{\mathrm{m}_{j+1}}\right) \end{bmatrix}, \tag{B.4}$$

$$\begin{aligned} \Sigma^{\Delta_{\mathrm{pre}_{j+1}}} &= \left(\boldsymbol{J}_{\Delta_{\mathrm{pre}_j}}^{\Delta_{\mathrm{pre}_{j+1}}}\right)\Sigma^{\Delta_{\mathrm{pre}_j}}\left(\boldsymbol{J}_{\Delta_{\mathrm{pre}_j}}^{\Delta_{\mathrm{pre}_{j+1}}}\right)^{\mathsf{T}} \\ &+ \left(\boldsymbol{J}_{\overline{\boldsymbol{a}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}}\right)\Sigma_{\mathrm{acc}}\left(\boldsymbol{J}_{\overline{\boldsymbol{a}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}}\right)^{\mathsf{T}} \\ &+ \left(\boldsymbol{J}_{\overline{\boldsymbol{\omega}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}}\right)\Sigma_{\mathrm{gyr}}\left(\boldsymbol{J}_{\overline{\boldsymbol{\omega}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}}\right)^{\mathsf{T}} \end{aligned} \tag{B.5}$$

$$\boldsymbol{J}_{\boldsymbol{b}_{\mathrm{acc}},i}^{\Delta_{\mathrm{pre}_{j+1}}} = -\boldsymbol{J}_{\overline{\boldsymbol{a}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}} + \boldsymbol{J}_{\Delta_{\mathrm{pre}_j}}^{\Delta_{\mathrm{pre}_{j+1}}}\boldsymbol{J}_{\mathrm{acc},i}^{\Delta_{\mathrm{pre}}}, \tag{B.6}$$

$$\boldsymbol{J}_{\boldsymbol{b}_{\mathrm{gyr}},i}^{\Delta_{\mathrm{pre}_{j+1}}} = -\boldsymbol{J}_{\overline{\boldsymbol{\omega}}_{j+1}}^{\Delta_{\mathrm{pre}_{j+1}}} + \boldsymbol{J}_{\Delta_{\mathrm{pre}_j}}^{\Delta_{\mathrm{pre}_{j+1}}}\boldsymbol{J}_{\overline{\boldsymbol{\omega}}_{j+1},i}^{\Delta_{\mathrm{pre}}}. \tag{B.7}$$

B.2 VECTORIZATION

We used the Symbolic Math toolbox[1] in Matlab R2021b for verification of Eq. 6.121:

```
%% verification script for vectorization using symbolic toolbox
% We skip the filler '0' entries in vecL and 0-rows in H2.
% => 6 instead of 8 values / rows
%
% linear indexing from 3x3 matrix:
% !!! matlab starts indexing with 1 !!!
sym_ind_L = [1,5,9,2,3,6]';
vec = @(x) x(:); % R^9
vecL = @(x) x(sym_ind_L); % R^6

% diagonal matrix D in R^3x3, 3 non-zero entries along diagonal
D00 = sym('D00'); D11 = sym('D11'); D22 = sym('D22');
D = [D00,0,0;0,D11,0;0,0,D22];
assume(D,'real');
% Eigenvectors V are orthogonal but not necessarily symmetric!
v00 = sym('v00'); v10 = sym('v10'); v20 = sym('v20');
v01 = sym('v01'); v11 = sym('v11'); v21 = sym('v21');
v02 = sym('v02'); v12 = sym('v12'); v22 = sym('v22');
V = [v00,v01,v02;v10,v11,v12;v20,v21,v22]; % R^3x3
assume(V,'real');

% 00,10,20,01,11,21,20,21,22
H2 = [1,0,0,0,0,0,0,0,0;
      0,0,0,0,1,0,0,0,0;
      0,0,0,0,0,0,0,0,1;
      0,1,0,0,0,0,0,0,0;
      0,0,1,0,0,0,0,0,0;
      0,0,0,0,0,1,0,0,0]; % R^6x9
% verify H2:
simplify((H2 * vec(V)) == vecL(V)) % equal
% verify kron prod:
simplify(vecL(V*D*V') == vecL(kron(V,V)*vec(D))) % equal

% compute optimized form:
% !!! no zero filler rows added !!!
Ev = [V(1,:).*V(2,:);V(1,:).*V(3,:); V(2,:).*V(3,:)];
vdvt = [(V.*V);Ev] * diag(D);
% compare against original VDV':
simplify(vecL(V*D*V') == vdvt) % equal
% compare against kron prod version:
simplify(vecL(kron(V,V)*vec(D)) == vdvt) % equal
```

1 https://www.mathworks.com/products/symbolic.html

B.3 Permutation

We use the following Matlab script for verification in Sec. 6.2.11 and verified in Matlab R2021b that this works up to $d = 9$. The number of permutations becomes the limiting factor with increasing dimension d.

```
%% verification script for sorting lattice ranks
% !!! matlab starts indexing with 1 !!!
clear all; clc;
d = 3; % dimension
n = d+1; % lifted dimension
i_to_d = 0:d; % sorted positions
ranks = flipud(perms(i_to_d)); % all permutations of ranks
dist = rand(n,1); % some rand vals to check equality
N = 2^nextpow2(n); % next higher multiple of 2, used for shifting
lowBitVal = N-1; % all lower bits set, used to extract idx back
v1 = zeros(size(ranks,1),1); v2 = v1; % store correctness
% check every permutation:
for num = 1 : size(ranks,1)
    b = ranks(num,:); % current rank permutation
    ind = b + 1; % +1 for matlab indexing
    % init variables:
    P = zeros(n,n); sb = zeros(size(dist)); gb = sb;
    % compute permutation matrix
    for i = 1 : (d+1) % shifted by 1 for matlab indexing
        P(i_to_d(i)+1,ind(i)) = 1;
    end
    % compute w/ permutation matrix for verification
    %  ! Mat-Mul is not efficient for this problem !
    cb = P' * dist;
    % compute idx with sorting
    % 1. shift rank left, add position in lowerbits
    ir = uint16(b) * uint16(N) + uint16(i_to_d);
    % 2. sort and extract lowerbits -> idx
    %    real impl should use parallel sorting, w/ min,max
    idx = bitand(uint16(sort(ir(:))'),uint16(lowBitVal))+1;
    % compute scattering / gathering:
    for i = 1 : (d+1) % shifted by 1 for matlab indexing
        % b[ind[i]] := d[i]
        sb(ind(i)) = dist(i);
        % b[i] := d[idx[i]]
        gb(i) = dist(idx(i));
    end
    % verify by checking distances:
    v1(num) = 1e-6 > norm(sb(:) - gb(:));
    v2(num) = 1e-6 > norm(sb(:) - cb(:));
end
% check all are correct:
(sum(v1) == length(v1)) && (sum(v2) == length(v2))
```

List of Figures

List of Tables

Bibliography

Abboud, N., M. Sayour, I. H. Elhajj, J. Zelek, and D. Asmar (2024). "Inline Photometrically Calibrated Hybrid Visual SLAM." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS58592.2024.10802153.

Adams, A., J. Baek, and M. A. Davis (2010). "Fast High-Dimensional Filtering Using the Permutohedral Lattice." In: *Computer Graphics Forum* 29.2. DOI: 10.1111/j.1467-8659.2009.01645.x.

Agarwal, S., K. Mierle, and The Ceres Solver Team (2022). *Ceres Solver*. Version 2.1. URL: https://github.com/ceres-solver/ceres-solver.

Alexandrov, S., J. Prankl, M. Zillich, and M. Vincze (2016). "Calibration and Correction of Vignetting Effects with an Application to 3D Mapping." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2016.7759621.

Ali, S. A., D. Aouada, G. Reis, and D. Stricker (2023). "DELO: Deep Evidential LiDAR Odometry using Partial Optimal Transport." In: *Workshop Proceedings of the IEEE International Conference on Computer Vision (ICCV Workshops)*. DOI: 10.1109/ICCVW60793.2023.00486.

Alon, O., S. Rabinovich, C. Fyodorov, and J. R. Cauchard (2021). "Drones in Firefighting: A User-Centered Design Perspective." In: *Proceedings of the International Conference on Mobile Human-Computer Interaction (MobileHCI)*. DOI: 10.1145/3447526.3472030.

Amanatides, J. and A. Woo (1987). "A fast voxel traversal algorithm for ray tracing." In: *Eurographics*. Vol. 87. 3. DOI: 10.2312/egtp.19871000.

Anderson, S., T. Barfoot, C. Tong, and S. Särkkä (2015). "Batch nonlinear continuous-time trajectory estimation as exactly sparse Gaussian process regression." In: *AR* 39.3. DOI: 10.1007/s10514-015-9455-y.

Arnold, E., J. Wynn, S. Vicente, G. Garcia-Hernando, Á. Monszpart, V. A. Prisacariu, D. Turmukhambetov, and E. Brachmann (2022). "Map-free Visual Relocalization: Metric Pose Relative to a Single Image." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-031-19769-7_40.

Bao, S. Y. and S. Savarese (2011). "Semantic structure from motion." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2011.5995462.

Barron, J. T. (2019). "A General and Adaptive Robust Loss Function." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2019.00446.

Bartolomei, L., L. Teixeira, and M. Chli (2020). "Perception-aware Path Planning for UAVs using Semantic Segmentation." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS45743.2020.9341347.

Behley, J., M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall (2019). "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2019.00939.

Behley, J. and C. Stachniss (2018). "Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments." In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2018.XIV.016.

Bergmann, P., R. Wang, and D. Cremers (2018). "Online Photometric Calibration of Auto Exposure Video for Realtime Visual Odometry and SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/LRA.2017.2777002.

Bergstra, J., D. Yamins, and D. Cox (2013). "Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures." In: *Proceedings of the IEEE International Conference on Machine Learning (ICML)*. URL: https://proceedings.mlr.press/v28/bergstra13.html.

Besl, P. J. and N. D. McKay (1992). "A method for registration of 3-D shapes." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 14.2. DOI: 10.1109/34.121791.

Beul, M., D. Droeschel, M. Nieuwenhuisen, J. Quenzel, S. Houben, and S. Behnke (2018). "Fast Autonomous Flight in Warehouses for Inventory Applications." In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2018.2849833.

Beul, M., N. Krombach, Y. Zhong, D. Droeschel, M. Nieuwenhuisen, and S. Behnke (2015). "A High-performance MAV for Autonomous Navigation in Complex 3D Environments." In: *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*. DOI: 10.1109/ICUAS.2015.7152417.

Bhat, S., R. Birkl, D. Wofk, P. Wonka, and M. Müller (2023). *ZoeDepth: Zero-shot Transfer by Combining Relative and Metric Depth*. arXiv: 2302.12288 [cs.CV].

Blanco, J.-L. (2010). *A tutorial on SE(3) transformation parameterizations and on-manifold optimization*. Tech. rep. 012010. University of Malaga. URL: http://ingmec.ual.es/~jlblanco/papers/jlblanco2010geometry3D_techrep.pdf.

Blanco, J.-L. and P. K. Rai (2014). *nanoflann: a C++ header-only fork of FLANN, a library for Nearest Neighbor (NN) with KD-trees*. https://github.com/jlblancoc/nanoflann.

Bleyer, M., C. Rhemann, and C. Rother (2011). "PatchMatch Stereo - Stereo Matching with Slanted Support Windows." In: *Proceedings of the British Machine Vision Conference (BMVC)*. DOI: 10.5244/C.25.14.

Bochkovskiy, A., A. Delaunoy, H. Germain, M. Santos, Y. Zhou, S. Richter, and V. Koltun (2025). "Depth Pro: Sharp Monocular Metric Depth in Less Than a Second." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=aueXfY0Clv.

Botsch, M., A. Hornung, M. Zwicker, and L. Kobbelt (2005). "High-quality surface splatting on today's GPUs." In: *Proceedings Eurographics/IEEE VGTC Symposium Point-Based Graphics*. DOI: 10.1109/PBG.2005.194059.

Bultmann, S. and S. Behnke (2022). "3D Semantic Scene Perception using Distributed Smart Edge Sensors." In: *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS)*. DOI: 10.1007/978-3-031-22216-0_22.

Bultmann*, S., J. Quenzel*, and S. Behnke (2021). "Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles." In: *Proceedings of the European Conference on Mobile Robots (ECMR)*. DOI: 10.1109/ECMR50962.2021.9568812.

– (2023). "Real-Time Multi-Modal Semantic Fusion on Unmanned Aerial Vehicles with Label Propagation for Cross-Domain Adaptation." In: *Journal of Robotics and Autonomous Systems* 159. DOI: 10.1016/j.robot.2022.104286.

Burnett, K., A. P. Schoellig, and T. D. Barfoot (2024). "Continuous-Time Radar-Inertial and Lidar-Inertial Odometry Using a Gaussian Process Motion Prior." In: *IEEE Transactions on Robotics (T-RO)* 41. DOI: 10.1109/TRO.2024.3521856.

Burri, M., J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart (2016). "The EuRoC micro aerial vehicle datasets." In: *The International Journal of Robotics Research (IJRR)*. DOI: 10.1177/0278364915620033.

Burri, M., H. Oleynikova, M. W. Achtelik, and R. Siegwart (2015). "Real-Time Visual-Inertial Mapping, Re-localization and Planning Onboard MAVs in Unknown Environments." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2015.7353622.

Caesar, H., V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom (2020). "nuScenes: A Multimodal Dataset for Autonomous Driving." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR42600.2020.01164.

Calonder, M., V. Lepetit, C. Strecha, and P. Fua (2010). "BRIEF: Binary Robust Independent Elementary Features." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-642-15561-1_56.

Campos, C., R. Elvira, J. Rodríguez, J. Montiel, and J. Tardós (2021). "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM." In: *IEEE Transactions on Robotics (T-RO)* 37.6. DOI: 10.1109/TRO.2021.3075644.

Cao, C., X. Ren, and Y. Fu (2024). "MVSFormer++: Revealing the Devil in Transformer's Details for Multi-View Stereo." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=wXWfvSpYHh.

Chang, J.-H. R., W.-Y. Chen, A. Ranjan, K. M. Yi, and O. Tuzel (2023). "Pointersect: Neural Rendering with Cloud-Ray Intersection." In: *CVPR*. DOI: 10.1109/CVPR52729.2023.00808.

Chen, G.-r. (1992). "OPTIMAL INTERPOLATION OF SCATTERED DATA ON A CIRCULAR DOMAIN WITH BOUNDARY CONDITIONS." In: *Journal of Computational Mathematics* 10.4. URL: http://www.jstor.org/stable/45340251.

Chen, G. and W. Wang (2024). *A Survey on 3D Gaussian Splatting*. arXiv: 2401.03890 [cs.CV].

Chen, K., B. Lopez, A.-a. Agha-mohammadi, and A. Mehta (2022). "Direct LiDAR Odometry: Fast Localization With Dense Point Clouds." In: *IEEE Robotics and Automation Letters (RA-L)* 7.2. DOI: 10.1109/LRA.2022.3142739.

Chen, K., R. Nemiroff, and B. T. Lopez (2023a). "Direct LiDAR-Inertial Odometry: Lightweight LIO with Continuous-Time Motion Correction." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48891.2023.10160508.

Chen, Q., S. Shu, and X. Bai (2024a). "Thermal3D-GS: Physics-Induced 3D Gaussians for Thermal Infrared Novel-View Synthesis." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-031-73383-3_15.

Chen, X., A. Milioto, E. Palazzolo, P. Giguère, J. Behley, and C. Stachniss (2019). "SuMa++: Efficient LiDAR-based Semantic SLAM." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS40897.2019.8967704.

Chen, Y., X. Chen, X. Wang, Q. Zhang, Y. Guo, Y. Shan, and F. Wang (2023b). "Local-to-Global Registration for Bundle-Adjusting Neural Radiance Fields." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52729.2023.00799.

Chen, Z., Y. Xu, S. Yuan, and L. Xie (2024b). "iG-LIO: An Incremental GICP-based Tightly-coupled LiDAR-inertial Odometry." In: *IEEE Robotics and Automation Letters (RA-L)* 9.2. DOI: 10.1109/LRA.2024.3349915.

Chung, D. and J. Kim (2024). "NV-LIOM: LiDAR-Inertial Odometry and Mapping Using Normal Vectors Towards Robust SLAM in Multifloor Environments." In: *IEEE Robotics and Automation Letters (RA-L)* 9.11. DOI: 10.1109/LRA.2024.3457373.

Civera, J., D. Gálvez-López, L. Riazuelo, J. Tardós, and J. Montiel (2011). "Towards semantic SLAM using a monocular camera." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2011.6094648.

Cortinhal, T., G. Tzelepis, and E. E. Aksoy (2020). "SalsaNext: Fast, Uncertainty-Aware Semantic Segmentation of LiDAR Point Clouds." In: *Advances in Visual Computing*. DOI: 10.1007/978-3-030-64559-5_16.

Cover, T. M. and J. A. Thomas (2005). "Differential Entropy." In: *Elements of Information Theory*. John Wiley & Sons, Ltd. Chap. 8. ISBN: 9780471748823. DOI: https://doi.org/10.1002/047174882X.ch8.

Dai, A., M. Nießner, M. Zollöfer, S. Izadi, and C. Theobalt (2017). "BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration." In: *ACM Transactions on Graphics*. DOI: 10.1145/3054739.

Dalal, A., D. Hagen, K. G. Robbersmyr, and K. M. Knausgård (2024). "Gaussian Splatting: 3D Reconstruction and Novel View Synthesis: A Review." In: *IEEE Access* 12. DOI: 10.1109/ACCESS.2024.3408318.

Das, M. P., L. Matthies, and S. Daftry (2021). "Online Photometric Calibration of Automatic Gain Thermal Infrared Cameras." In: *IEEE Robotics and Automation Letters (RA-L)* 6.2. DOI: 10.1109/LRA.2021.3061401.

Daun, K., M. Schnaubelt, S. Kohlbrecher, and O. von Stryk (2021). "HectorGrapher: Continuous-time Lidar SLAM with Multi-resolution Signed Distance Function Registration for Challenging Terrain." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR53300.2021.9597690.

Debevec, P. E. and J. Malik (1997). "Recovering High Dynamic Range Radiance Maps from Photographs." In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. DOI: 10.1145/258734.258884.

Dellenbach, P., J.-E. Deschaud, B. Jacquet, and F. Goulette (2022). "CT-ICP: Real-time Elastic LiDAR Odometry with Loop Closure." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA46639.2022.9811849.

Demmel, N., D. Schubert, C. Sommer, D. Cremers, and V. Usenko (2021). "Square Root Marginalization for Sliding-Window Bundle Adjustment." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV48922.2021.01301.

Deschaud, J.-E., D. Duque, J. P. Richa, S. Velasco-Forero, B. Marcotegui, and F. Goulette (2021). "Paris-CARLA-3D: A Real and Synthetic Outdoor Point Cloud Dataset for Challenging Tasks in 3D Mapping." In: *Remote Sensing* 13.22. DOI: 10.3390/rs13224713.

Dexheimer, E. and A. Davison (2024). "COMO: Compact Mapping and Odometry." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-031-73001-6_20.

Di Giammarino, L., L. Brizi, T. Guadagnino, C. Stachniss, and G. Grisetti (2022). "MD-SLAM: Multi-cue Direct SLAM." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS47612.2022.9981147.

DIN 4109-2:2018-01 (2018). *Schallschutz im Hochbau - Teil 2: Rechnerische Nachweise der Erfüllung der Anforderungen*. Norm. DOI: 10.31030/2764609.

Dong, H., X. Chen, S. Särkkä, and C. Stachniss (2023). "Online pole segmentation on range images for long-term LiDAR localization in urban environments." In: *Journal of Robotics and Autonomous Systems* 159. DOI: 10.1016/j.robot.2022.104283.

Dosovitskiy, A. et al. (2021). "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=YicbFdNTTy.

Droeschel, D. and S. Behnke (2016). "MRSLaserMap: Local Multiresolution Grids for Efficient 3D Laser Mapping and Localization." In: *Proceedings of the RoboCup International Symposium*. DOI: 10.1007/978-3-319-68792-6_26.

– (2018). "Efficient Continuous-time SLAM for 3D Lidar-based Online Mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2018.8461000.

Droeschel, D., M. Schwarz, and S. Behnke (2017). "Continuous Mapping and Localization for Autonomous Navigation in Rough Terrain using a 3D Laser Scanner." In: *Journal of Robotics and Autonomous Systems*. DOI: 10.1016/j.robot.2016.10.017.

Duberg, D. and P. Jensfelt (2020). "UFOMap: An Efficient Probabilistic 3D Mapping Framework That Embraces the Unknown." In: *IEEE Robotics and Automation Letters (RA-L)* 5.4. DOI: 10.1109/LRA.2020.3013861.

Engel, J., V. Koltun, and D. Cremers (2018). "Direct Sparse Odometry." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 40.3. DOI: 10.1109/TPAMI.2017.2658577.

Engel, J., T. Schöps, and D. Cremers (2014). "LSD-SLAM: Large-scale direct monocular SLAM." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-319-10605-2_54.

Engel, J., J. Stückler, and D. Cremers (2015). "Large-Scale Direct SLAM with Stereo Cameras." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2015.7353631.

Engel, J., V. Usenko, and D. Cremers (2016). *A Photometrically Calibrated Benchmark For Monocular Visual Odometry*. arXiv: 1607.02555 [cs.CV].

Fan, Z. et al. (2024). *InstantSplat: Unbounded Sparse-view Pose-free Gaussian Splatting in 40 Seconds*. arXiv: 2403.20309 [cs.CV].

Ferrari, S., L. D. Giammarino, L. Brizi, and G. Grisetti (2024). "MAD-ICP: It is All About Matching Data – Robust and Informed LiDAR Odometry." In: *IEEE Robotics and Automation Letters (RA-L)* 9.11. DOI: 10.1109/LRA.2024.3456509.

FLIR Systems Inc. (2016). *5 Factors Influencing Radiometric Temperature Measurements*. Tech. rep. FLIR Systems Inc. URL: https://www.flir.com/discover/oem/cores/5-factors-influencing-radiometric-temperature-measurements/.

Fog, A. (2022). *VCL - C++ vector class library manual*. https://www.agner.org/optimize/vcl_manual.pdf.

Forster, C., M. Pizzoli, and D. Scaramuzza (2014). "SVO: Fast Semi-Direct Monocular Visual Odometry." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2014.6906584.

Forster, C., Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza (2017). "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems." In: *IEEE Transactions on Robotics (T-RO)* 33.2. DOI: 10.1109/TRO.2016.2623335.

Fridovich-Keil, S., A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa (2022). "Plenoxels: Radiance Fields without Neural Networks." In: *CVPR*. DOI: 10.1109/CVPR52688.2022.00542.

Fu, Y., X. Wang, S. Liu, A. Kulkarni, J. Kautz, and A. Efros (2024). "COLMAP-Free 3D Gaussian Splatting." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.01965.

Funk, N., J. Tarrio, S. Papatheodorou, M. Popović, P. F. Alcantarilla, and S. Leutenegger (2021). "Multi-Resolution 3D Mapping With Explicit Free Space Representation for Fast and Accurate Mobile Robot Motion Planning." In: *IEEE Robotics and Automation Letters (RA-L)* 6.2. DOI: 10.1109/LRA.2021.3061989.

Furgale, P., J. Rehder, and R. Siegwart (2013). "Unified temporal and spatial calibration for multi-sensor systems." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2013.6696514.

Gamache, O., J.-M. Fortin, M. Boxan, M. Vaidis, F. Pomerleau, and P. Giguère (2024). "Exposing the Unseen: Exposure Time Emulation for Offline Benchmarking of Vision Algorithms." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS58592.2024.10803057.

Gantmakher, F. R. (1960). *The theory of matrices*. Vol. 131. American Mathematical Society.

Geiger, A., P. Lenz, and R. Urtasun (2012). "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2012.6248074.

Geneva, P., K. Eckenhoff, W. Lee, Y. Yang, and G. Huang (2020). "OpenVINS: A Research Platform for Visual-Inertial Estimation." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9196524.

Goldman, D. and J. Chen (2005). "Vignette and exposure calibration and compensation." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2005.249.

Grinvald, M., F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto (2019). "Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery." In: *IEEE Robotics and Automation Letters (RA-L)* 4.3. DOI: 10.1109/LRA.2019.2923960.

Grossberg, M. and S. Nayar (2004). "Modeling the space of camera response functions." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 26.10. DOI: 10.1109/TPAMI.2004.88.

Grupp, M. (2017). *evo: Python package for the evaluation of odometry and SLAM.* https://github.com/MichaelGrupp/evo.

Gu, X., Z. Fan, S. Zhu, Z. Dai, F. Tan, and P. Tan (2020). "Cascade Cost Volume for High-Resolution Multi-View Stereo and Stereo Matching." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR42600.2020.00257.

Guadagnino, T., B. Mersch, I. Vizzo, S. Gupta, M. Malladi, L. Lobefaro, G. Doisy, and C. Stachniss (2025). "Kinematic-ICP: Enhancing LiDAR Odometry with Kinematic Constraints for Wheeled Mobile Robots Moving on Planar Surfaces." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA55743.2025.11128503.

Gupta, S., T. Guadagnino, B. Mersch, N. Trekel, M. Malladi, and C. Stachniss (2025). *Efficiently Closing Loops in LiDAR-Based SLAM Using Point Cloud Density Maps.* arXiv: 2501.07399 [cs.RO].

Haarbach, A., T. Birdal, and S. Ilic (2018). "Survey of Higher Order Rigid Body Motion Interpolation Methods for Keyframe Animation and Continuous-Time Trajectory Estimation." In: *Proceedings of the International Conference on 3D Vision (3DV)*. DOI: 10.1109/3DV.2018.00051.

Haber, E. and J. Modersitzki (2006). "Intensity gradient based registration and fusion of multi-modal images." In: *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. DOI: 10.1007/11866763_89.

Haidar, J., D. Khalil, and D. Asmar (2024). "OSPC: Online Sequential Photometric Calibration." In: *Pattern Recognition Letters* 181. DOI: https://doi.org/10.1016/j.patrec.2024.03.005.

Handa, A., T. Whelan, J. McDonald, and A. J. Davison (2014). "A Benchmark for RGB-D Visual Odometry, 3D Reconstruction and SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2014.6907054.

Hassan, M., F. Forest, O. Fink, and M. Mielle (2025). "ThermoNeRF: A multimodal Neural Radiance Field for joint RGB-thermal novel view synthesis of building

facades." In: *Advanced Engineering Informatics* 65. DOI: 10.1016/j.aei.2025.103345.

He, D., W. Xu, N. Chen, F. Kong, C. Yuan, and F. Zhang (2023). "Point-LIO: Robust High-Bandwidth Light Detection and Ranging Inertial Odometry." In: *Advanced Intelligent Systems* 5.7. DOI: 10.1002/aisy.202200459.

Held, J., R. Vandeghen, A. Hamdi, A. Deliege, A. Cioppa, S. Giancola, A. Vedaldi, B. Ghanem, and M. Van Droogenbroeck (2025). "3D Convex Splatting: Radiance Field Rendering with 3D Smooth Convexes." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52734.2025.01990.

Hermans, A., G. Floros, and B. Leibe (2014). "Dense 3D semantic mapping of indoor scenes from RGB-D images." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2014.6907236.

Hershey, J. R. and P. A. Olsen (2007). "Approximating the Kullback Leibler divergence between Gaussian mixture models." In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. DOI: 10.1109/ICASSP.2007.366913.

Higham, N. J. (1989). "Matrix Nearness Problems and Applications." In: *Applications of Matrix Theory*. Oxford University Press. DOI: 10.2307/3619410.

– (2002). *Accuracy and Stability of Numerical Algorithms*. Second. Society for Industrial and Applied Mathematics. DOI: 10.1137/1.9780898718027.

Hirschmüller, H. (2011). "Semi-global matching-motivation, developments and applications." In: *Photogrammetric Week 11*. URL: http://www.ifp.uni-stuttgart.de/publications/phowo11/index.html.

Hodson, T. (2022). "Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not." In: *Geoscientific Model Development* 15.14. DOI: 10.5194/gmd-15-5481-2022.

Höfle, B. and N. Pfeifer (2007). "Correction of laser scanning intensity data: Data and model-driven approaches." In: *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)* 62.6. DOI: 10.1016/j.isprsjprs.2007.05.008.

Holland, P. W. and R. E. Welsch (1977). "Robust regression using iteratively reweighted least-squares." In: *Communications in Statistics - Theory and Methods* 6.9. DOI: 10.1080/03610927708827533.

Holz, D., A. Ichim, F. Tombari, R. Rusu, and S. Behnke (2015). "Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D." In: *IEEE Robotics & Automation Magazine* 22.4. DOI: 10.1109/MRA.2015.2432331.

Hong, S., J. He, X. Zheng, and C. Zheng (2024). "LIV-GaussMap: LiDAR-Inertial-Visual Fusion for Real-Time 3D Radiance Field Map Rendering." In: *IEEE Robotics and Automation Letters (RA-L)* 9.11. DOI: 10.1109/LRA.2024.3400149.

Hong, S., J. Jung, H. Shin, J. Han, J. Yang, C. Luo, and S. Kim (2025). "PF3plat: Pose-Free Feed-Forward 3D Gaussian Splatting for Novel View Synthesis." In: *Proceedings of the IEEE International Conference on Machine Learning (ICML)*. URL: https://openreview.net/forum?id=VjI1NnsW4t.

Horn, R. A. and C. R. Johnson (1991). "Matrix equations and the Kronecker product." In: *Topics in Matrix Analysis*. Cambridge University Press. DOI: 10.1017/CBO9780511840371.

Hornung, A., K. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard (2013). "OctoMap: An efficient probabilistic 3D mapping framework based on octrees." In: *Autonomous Robots* 34.3. DOI: 10.1007/s10514-012-9321-0.

Hou, S.-H. (1998). "Classroom Note: A Simple Proof of the Leverrier–Faddeev Characteristic Polynomial Algorithm." In: *SIAM Review* 40.3. DOI: 10.1137/S003614459732076X.

Houben, S., J. Quenzel, N. Krombach, and S. Behnke (2016). "Efficient multi-camera visual-inertial SLAM for micro aerial vehicles." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2016.7759261.

Huai, Z. and G. Huang (2024). "A Consistent Parallel Estimation Framework for Visual-Inertial SLAM." In: *IEEE Transactions on Robotics (T-RO)* 40. DOI: 10.1109/TRO.2024.3433868.

Huang, B., Z. Yu, A. Chen, A. Geiger, and S. Gao (2024). "2D Gaussian Splatting for Geometrically Accurate Radiance Fields." In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. DOI: 10.1145/3641519.3657428.

Huang, G. P., A. I. Mourikis, and S. I. Roumeliotis (2008). "Analysis and improvement of the consistency of extended Kalman filter based SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.2008.4543252.

Huber, P. J. (1964). "Robust Estimation of a Location Parameter." In: *The Annals of Mathematical Statistics* 35.1. DOI: 10.1214/aoms/1177703732.

Intel Corporation (2021). *Intrinsics for Permute Operations*. https://www.intel.com/content/www/us/en/docs/cpp-compiler/developer-guide-reference/2021-10/intrinsics-for-permute-operations-001.html.

– (2023a). *Intrinsics for Integer Gather and Scatter Operations*. https://www.intel.com/content/www/us/en/docs/cpp-compiler/developer-guide-reference/2021-8/intrinsics-for-int-gather-and-scatter-ops.html.

– (2023b). *Shuffle Intrinsics*. https://www.intel.com/content/www/us/en/docs/cpp-compiler/developer-guide-reference/2021-10/shuffle-intrinsics.html.

Irmisch, P., I. Ernst, D. Baumbach, M. M. Linkiewicz, V. Unnithan, F. Sohl, J. Wohlfeil, and D. Grießbach (2021). "A Hand-Held Sensor System for Exploration and Thermal Mapping of Volcanic Fumarole Fields." In: *Proceedings of the International Symposium on Geometry and Vision (ISGV)*. DOI: 10.1007/978-3-030-72073-5_6.

Irshad, M. Z., M. Comi, Y.-C. Lin, N. Heppert, A. Valada, R. Ambrus, Z. Kira, and J. Tremblay (2024). *Neural Fields in Robotics: A Survey*. arXiv: 2410.20220 [cs.RO].

Isaacson, S., P.-C. Kung, M. Ramanagopal, R. Vasudevan, and K. A. Skinner (2023). "LONER: LiDAR Only Neural Representations for Real-Time SLAM." In: *IEEE Robotics and Automation Letters (RA-L)* 8.12. DOI: 10.1109/LRA.2023.3324521.

Jiang, C., R. Gao, K. Shao, Y. Wang, R. Xiong, and Y. Zhang (2025). "LI-GS: Gaussian Splatting with LiDAR Incorporated for Accurate Large-Scale Reconstruction."

In: *IEEE Robotics and Automation Letters (RA-L)* 10.2. DOI: 10.1109/LRA.2024.3522846.

Jordan, S., J. Moore, S. Hovet, J. Box, J. Perry, K. Kirsche, D. Lewis, and Z. Tse (2018). "State-of-the-art technologies for UAV inspections." In: *IET Radar, Sonar & Navigation* 12.2. DOI: 10.1049/iet-rsn.2017.0251.

Judd, K. M., J. D. Gammell, and P. Newman (2018). "Multimotion Visual Odometry (MVO): Simultaneous Estimation of Camera and Third-Party Motions." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8594213.

Julier, S. J. and J. K. Uhlmann (1997). "New extension of the Kalman filter to nonlinear systems." In: *Signal Processing, Sensor Fusion, and Target Recognition VI*. Vol. 3068. DOI: 10.1117/12.280797.

Jung, M., S. Jung, and A. Kim (2023). "Asynchronous Multiple LiDAR-Inertial Odometry Using Point-Wise Inter-LiDAR Uncertainty Propagation." In: *IEEE Robotics and Automation Letters (RA-L)* 8.7. DOI: 10.1109/LRA.2023.3281264.

Kahu, S. Y., R. B. Raut, and K. M. Bhurchandi (2019). "Review and evaluation of color spaces for image/video compression." In: *Color Research & Application* 44.1. DOI: 10.1002/col.22291.

Kazhdan, M. and H. Hoppe (2013). "Screened Poisson Surface Reconstruction." In: *ACM Transactions on Graphics* 32.3. DOI: 10.1145/2487228.2487237.

Kerbl, B., G. Kopanas, T. Leimkuehler, and G. Drettakis (2023). "3D Gaussian Splatting for Real-Time Radiance Field Rendering." In: *ACM Transactions on Graphics* 42.4. DOI: 10.1145/3592433.

Kerbl, B., A. Meuleman, G. Kopanas, M. Wimmer, A. Lanvin, and G. Drettakis (2024). "A Hierarchical 3D Gaussian Representation for Real-Time Rendering of Very Large Datasets." In: *ACM Transactions on Graphics* 43.4. DOI: 10.1145/3658160.

Kerl, C., J. Sturm, and D. Cremers (2013a). "Dense Visual SLAM for RGB-D Cameras." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2013.6696650.

– (2013b). "Robust Odometry Estimation for RGB-D Cameras." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2013.6631104.

Khattak, S., H. Nguyen, F. Mascarich, T. Dang, and K. Alexis (2020). "Complementary Multi–Modal Sensor Fusion for Resilient Robot Pose Estimation in Subterranean Environments." In: *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*. DOI: 10.1109/ICUAS48674.2020.9213865.

Kim, G. and A. Kim (2020). "Remove, then Revert: Static Point cloud Map Construction using Multiresolution Range Images." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS45743.2020.9340856.

Kim, G., Y. S. Park, Y. Cho, J. Jeong, and A. Kim (2020). "MulRan: Multimodal Range Dataset for Urban Place Recognition." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9197298.

Kim, H., N. Lamichhane, C. Kim, and R. Shrestha (2023). "Innovations in Building Diagnostics and Condition Monitoring: A Comprehensive Review of Infrared Thermography Applications." In: *Buildings* 13.11. DOI: 10.3390/buildings13112829.

Kim, M. G. (2000). "Multivariate outliers and decompositions of mahalanobis distance." In: *Communications in Statistics - Theory and Methods* 29.7. DOI: 10.1080/03610920008832559.

Kingma, D. P. and J. Ba (2015). "Adam: A Method for Stochastic Optimization." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=8gmWwjFyLj.

Klein, G. and D. Murray (2007). "Parallel tracking and mapping for small AR workspaces." In: *Proceeding of the IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*. DOI: 10.1109/ISMAR.2007.4538852.

Knutson, A. and T. Tao (2001). "Honeycombs and sums of Hermitian matrices." In: *Notices of the American Mathematical Society* 48.2. DOI: 10.48550/arXiv.math/0009048.

Kong, X., S. Liu, M. Taher, and A. J. Davison (2023). "vMAP: Vectorised Object Mapping for Neural Field SLAM." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52729.2023.00098.

Konolige, K. (2010). "Projected texture stereo." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ROBOT.2010.5509796.

Kostavelis, I. and A. Gasteratos (2015). "Semantic mapping for mobile robotics tasks: A survey." In: *Journal of Robotics and Autonomous Systems* 66. DOI: 10.1016/j.robot.2014.12.006.

Kruijff-Korbayova, I. et al. (2021). "German Rescue Robotics Center (DRZ): A Holistic Approach for Robotic Systems Assisting in Emergency Response." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR53300.2021.9597869.

Kundu, A., Y. Li, F. Dellaert, F. Li, and J. Rehg (2014). "Joint Semantic Segmentation and 3D Reconstruction from Monocular Video." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-319-10599-4_45.

Kung, P.-C., C.-C. Wang, and W.-C. Lin (2021). "A Normal Distribution Transform-Based Radar Odometry Designed For Scanning and Automotive Radars." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14417–14423. DOI: 10.1109/ICRA48506.2021.9561413.

Lancaster, P. and M. Tismenetsky (1985). *The Theory of Matrices: With Applications*. second. Computer Science and Scientific Computing. Elsevier Science.

Lang, X., C. Chen, K. Tang, Y. Ma, J. Lv, Y. Liu, and X. Zuo (2023). "Coco-LIC: Continuous-Time Tightly-Coupled LiDAR-Inertial-Camera Odometry Using Non-Uniform B-Spline." In: *IEEE Robotics and Automation Letters (RA-L)* 8.11. DOI: 10.1109/LRA.2023.3315542.

Lattimer, B. Y. et al. (2023). "Use of Unmanned Aerial Systems in Outdoor Firefighting." In: *Fire Technology* 59. DOI: 10.1007/s10694-023-01437-0.

Lauterbach, H. A., D. Borrmann, and A. Nüchter (2017). "Towards Radiometrical Alignment of 3D Point Clouds." In: *International Arch. Photogramm. Remote Sens.*

Spatial Inf. Sci. (ISPRS) XLII-2/W3. DOI: 10.5194/isprs-archives-XLII-2-W3-419-2017.

LeCun, Y., S. Chopra, R. Hadsell, M. Ranzato, and F. J. Huang (2006). "A Tutorial on Energy-based Learning." In: *Predicting structured data*. Vol. 1. The MIT Press. Chap. 10.

Lee, D., H. Lim, and S. Han (2025). "GenZ-ICP: Generalizable and Degeneracy-Robust LiDAR Odometry Using an Adaptive Weighting." In: *IEEE Robotics and Automation Letters (RA-L)* 10.1. DOI: 10.1109/LRA.2024.3498779.

Leroy, V., Y. Cabon, and J. Revaud (2024). "Grounding Image Matching in 3D with MASt3R." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-031-73220-1_5.

Leutenegger, S., S. Lynen, M. Bosse, R. Siegwart, and P. Furgale (2015). "Keyframe-based visual–inertial odometry using nonlinear optimization." In: *The International Journal of Robotics Research (IJRR)* 34.3. DOI: 10.1177/0278364914554813.

Li, J., R. Zhen, and R. Stevenson (2024). "Automatic exposure strategy network for robust visual odometry in environments with high dynamic range." In: *Machine Vision and Applications (MVA)* 36 (1). DOI: 10.1007/s00138-024-01623-2.

Li, K., M. Li, and U. D. Hanebeck (2021). "Towards High-Performance Solid-State-LiDAR-Inertial Odometry and Mapping." In: *IEEE Robotics and Automation Letters (RA-L)* 6.3. DOI: 10.1109/LRA.2021.3070251.

Li, Z., T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin (2023). "Neuralangelo: High-Fidelity Neural Surface Reconstruction." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52729.2023.00817.

Li, Z. and N. Snavely (2018). "MegaDepth: Learning Single-View Depth Prediction from Internet Photos." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2018.00218.

Lianos, K.-N., J. Schönberger, M. Pollefeys, and T. Sattler (2018). "VSO: Visual Semantic Odometry." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-030-01225-0_15.

Liao, Z., J. Yang, J. Qian, A. P. Schoellig, and S. L. Waslander (2024). "Uncertainty-aware 3D Object-Level Mapping with Deep Shape Priors." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA57147.2024.10611206.

Lim, H., S. Hwang, and H. Myung (2021). "ERASOR: Egocentric Ratio of Pseudo Occupancy-Based Dynamic Object Removal for Static 3D Point Cloud Map Building." In: *IEEE Robotics and Automation Letters (RA-L)* 6.2. DOI: 10.1109/LRA.2021.3061363.

Lin, C.-H., W.-C. Ma, A. Torralba, and S. Lucey (2021). "BARF: Bundle-Adjusting Neural Radiance Fields." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV48922.2021.00569.

Lin, J. et al. (2024a). "VastGaussian: Vast 3D Gaussians for Large Scene Reconstruction." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.00494.

Lin, J. and F. Zhang (2024). "$R^3$3LIVE++: A Robust, Real-Time, Radiance Reconstruction Package With a Tightly-Coupled LiDAR-Inertial-Visual State Estimator."

In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 46.12. DOI: 10.1109/TPAMI.2024.3456473.

Lin, T.-Y., P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie (2017). "Feature Pyramid Networks for Object Detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2017.106.

Lin, Y., X.-Y. Pan, S. Fridovich-Keil, and G. Wetzstein (2024b). "ThermalNeRF: Thermal Radiance Fields." In: *Proceedings of the IEEE International Conference on Computational Photography (ICCP)*. DOI: 10.1109/ICCP61108.2024.10644336.

Lindenberger, P., P.-E. Sarlin, and M. Pollefeys (2023). "LightGlue: Local Feature Matching at Light Speed." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV51070.2023.01616.

Liso, L., E. Sandström, V. Yugay, L. Van Gool, and M. Oswald (2024). "Loopy-SLAM: Dense Neural SLAM with Loop Closures." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.01925.

Liu, A., W. Lin, and M. Narwaria (2012). "Image Quality Assessment Based on Gradient Similarity." In: *IEEE Transactions on Image Processing* 21.4. DOI: 10.1109/TIP.2011.2175935.

Liu, J., Y. Zhang, X. Zhao, Z. He, W. Liu, and X. Lv (2024). "Fast and Robust LiDAR-Inertial Odometry by Tightly-Coupled Iterated Kalman Smoother and Robocentric Voxels." In: *IEEE Transactions on Intelligent Transportation Systems* 25.10. DOI: 10.1109/TITS.2024.3391291.

Liu, Q., Z. Wang, and H. Wang (2021). "PC-SD-VIO: A constant intensity semi-direct monocular visual-inertial odometry with online photometric calibration." In: *Journal of Robotics and Autonomous Systems* 146. DOI: 10.1016/j.robot.2021.103877.

Liu, S., C. C. Wang, G. Brunnett, and J. Wang (2016). "A closed-form formulation of HRBF-based surface reconstruction by approximate solution." In: *Computer-Aided Design* 78. DOI: 10.1016/j.cad.2016.05.001.

Lluvia, I., E. Lazkano, and A. Ansuategi (2021). "Active Mapping and Robot Exploration: A Survey." In: *Sensors* 21.7. DOI: 10.3390/s21072445.

Lopez-Fuentes, L., G. Oliver, and S. Massanet (2015). "Revisiting Image Vignetting Correction by Constrained Minimization of Log-Intensity Entropy." In: *Proceedings of the International Work-Conference on Artificial Neural Networks (IWANN)*. DOI: 10.1007/978-3-319-19222-2_38.

Lorensen, W. E. and H. E. Cline (1987). "Marching cubes: A high resolution 3D surface construction algorithm." In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. DOI: 10.1145/37401.37422.

Loshchilov, I. and F. Hutter (2019). "Decoupled Weight Decay Regularization." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

Lottes, P., R. Khanna, J. Pfeifer, R. Siegwart, and C. Stachniss (2017). "UAV-based crop and weed classification for smart farming." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2017.7989347.

Lovegrove, S., A. Patron-Perez, and G. Sibley (2013). "Spline Fusion: A continuous-time representation for visual-inertial fusion with application to rolling shutter cameras." In: *Proceedings of the British Machine Vision Conference (BMVC)*. DOI: 10.5244/C.27.93.

Lowe, D. G. (2004). "Distinctive Image Features from Scale-Invariant Keypoints." In: *International Journal of Computer Vision (IJCV)* 60.2. DOI: 10.1023/B:VISI.0000029664.99615.94.

Lu, R., H. Chen, Z. Zhu, Y. Qin, M. Lu, L. Zhang, C. Yan, and A. Xue (2025). "ThermalGaussian: Thermal 3D Gaussian Splatting." In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=ybFRoGxZjs.

Luo, D., Y. Zhuang, and S. Wang (2022). "Hybrid sparse monocular visual odometry with online photometric calibration." In: *The International Journal of Robotics Research (IJRR)* 41.11-12. DOI: 10.1177/02783649221107703.

Lv, J., K. Hu, J. Xu, Y. Liu, X. Ma, and X. Zuo (2021). "CLINS: Continuous-Time Trajectory Estimation for LiDAR-Inertial System." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS51168.2021.9636676.

Lv, J., X. Lang, J. Xu, M. Wang, Y. Liu, and X. Zuo (2023). "Continuous-Time Fixed-Lag Smoothing for LiDAR-Inertial-Camera SLAM." In: *IEEE/ASME Transactions on Mechatronics* 28.4. DOI: 10.1109/TMECH.2023.3241398.

Madsen, K., H. B. Nielsen, and O. Tingleff (2004). *Methods for nonlinear least squares problems*. second. Informatics and Mathematical Modelling, Technical University of Denmark.

Magnusson, M., A. Lilienthal, and T. Duckett (2007). "Scan registration for autonomous mining vehicles using 3D-NDT." In: *Journal of Field Robotics (JFR)* 24.10. DOI: 10.1002/rob.20204.

Manzini, T., R. Murphy, and D. Merrick (2023). "Quantitative Data Analysis: CRASAR Small Unmanned Aerial Systems at Hurricane Ian." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR59696.2023.10499943.

Martin-Brualla, R., N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth (2021). "NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR46437.2021.00713.

McCormac, J., R. Clark, M. Bloesch, A. Davison, and S. Leutenegger (2018). "Fusion++: Volumetric Object-Level SLAM." In: *Proceedings of the International Conference on 3D Vision (3DV)*. DOI: 10.1109/3DV.2018.00015.

McCormac, J., A. Handa, A. Davison, and S. Leutenegger (2017). "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2017.7989538.

Menze, M., C. Heipke, and A. Geiger (2015). "Joint 3D Estimation of Vehicles and Scene Flow." In: *ISPRS Workshop on Image Sequence Analysis (ISA)*. DOI: 10.5194/isprsannals-II-3-W5-427-2015.

Meribout, M., V. K. Tiwari, J. Herrera, and A. Baobaid (2023). "Solar panel inspection techniques and prospects." In: *Measurement* 209. DOI: 10.1016/j.measurement.2023.112466.

Miao, R., P. Liu, F. Wen, Z. Gong, W. Xue, and R. Ying (2022). "R-SDSO: Robust stereo direct sparse odometry." In: *The Visual Computer* 38 (6). DOI: 10.1007/s00371-021-02278-0.

Miao, Y. and M. Yamaguchi (2021). "Photometric Calibration for Stereo Camera with Gamma-like Response Function in Direct Visual Odometry." In: *Sensors* 21.21. DOI: 10.3390/s21217048.

Mildenhall, B., P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng (2021). "NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis." In: *ECCV*. DOI: 10.1145/3503250.

Millane, A., H. Oleynikova, E. Wirbel, R. Steiner, V. Ramasamy, D. Tingdahl, and R. Siegwart (2024). "nvblox: GPU-Accelerated Incremental Signed Distance Field Mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA57147.2024.10611532.

Moulon, P., P. Monasse, and R. Marlet (2013). "Global fusion of relative motions for robust, accurate and scalable structure from motion." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/ICCV.2013.403.

Moulon, P., P. Monasse, R. Perrot, and R. Marlet (2016). "OpenMVG: Open multiple view geometry." In: *International Workshop on Reproducible Research in Pattern Recognition (RRPR)*. DOI: 10.1007/978-3-319-56414-2_5.

Müller, T., A. Evans, C. Schied, and A. Keller (2022). "Instant Neural Graphics Primitives with a Multiresolution Hash Encoding." In: *ACM Transactions on Graphics* 41.4. DOI: 10.1145/3528223.3530127.

Mur-Artal, R., J. Montiel, and J. Tardós (2015). "ORB-SLAM: A Versatile and Accurate Monocular SLAM System." In: *IEEE Transactions on Robotics (T-RO)* 31.5. DOI: 10.1109/TRO.2015.2463671.

Mur-Artal, R. and J. Tardós (2017). "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras." In: *IEEE Transactions on Robotics (T-RO)* 33.5. DOI: 10.1109/TRO.2017.2705103.

Murphy, R., D. Merrick, J. Adams, J. Broder, A. Bush, L. Hart, and R. Hawkins (2021). "How Robots helped out after the Surfside Condo Collapse: Responders flew drones night and day to survey the collapse and search for survivors." In: *IEEE Spectrum*.

Nakashima, K. and R. Kurazume (2024). "LiDAR Data Synthesis with Denoising Diffusion Probabilistic Models." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA57147.2024.10611480.

Nashed, S., J. Jin Park, R. Webster, and J. Durham (2021). "Robust Rank Deficient SLAM." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS51168.2021.9636443.

Newcombe, R. A., S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon (2011a). "KinectFusion: real-time dense surface mapping and tracking." In: *Proceeding of the IEEE and ACM International*

Symposium on Mixed and Augmented Reality (ISMAR). DOI: 10.1109/ISMAR.2011.6092378.

Newcombe, R. A., S. J. Lovegrove, and A. J. Davison (2011b). "DTAM: Dense tracking and mapping in real-time." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2011.6126513.

Ng, T.-T., S.-F. Chang, and M.-P. Tsui (2007). "Using geometry invariants for camera response function estimation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2007.383000.

Nguyen, T.-M., Z. Cao, K. Li, S. Yuan, and L. Xie (2024a). *GPTR: Gaussian Process Trajectory Representation for Continuous-Time Motion Estimation*. arXiv: 2410.22931 [cs.RO].

Nguyen, T.-M., D. Duberg, P. Jensfelt, S. Yuan, and L. Xie (2023). "SLICT: Multi-Input Multi-Scale Surfel-Based Lidar-Inertial Continuous-Time Odometry and Mapping." In: *IEEE Robotics and Automation Letters (RA-L)* 8.4. DOI: 10.1109/LRA.2023.3246390.

Nguyen, T.-M., X. Xu, T. Jin, Y. Yang, J. Li, S. Yuan, and L. Xie (2024b). "Eigen Is All You Need: Efficient Lidar-Inertial Continuous-Time Odometry With Internal Association." In: *IEEE Robotics and Automation Letters (RA-L)* 9.6. DOI: 10.1109/LRA.2024.3391049.

Nguyen, T. et al. (2019). "MAVNet: An Effective Semantic Segmentation Micro-Network for MAV-Based Tasks." In: *IEEE Robotics and Automation Letters (RA-L)* 4.4. DOI: 10.1109/LRA.2019.2928734.

Nießner, M., M. Zollhöfer, S. Izadi, and M. Stamminger (2013). "Real-time 3D reconstruction at scale using voxel hashing." In: *ACM Transactions on Graphics* 32.6. DOI: 10.1145/2508363.2508374.

Nistér, D. (2004). "An efficient solution to the five-point relative pose problem." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 26.6. DOI: 10.1109/TPAMI.2004.17.

Nocedal, J. and S. J. Wright (2006). *Numerical Optimization*. Second. Springer. DOI: 0.1007/978-0-387-40065-5.

Oleynikova, H., Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto (2017). "Voxblox: Incremental 3D Euclidean Signed Distance Fields for On-Board MAV Planning." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2017.8202315.

Ollero, A., J. Martínez-de-Dios, and L. Merino (2006). "Unmanned aerial vehicles as tools for forest-fire fighting." In: *Forest Ecology and Management* 234. DOI: 10.1016/j.foreco.2006.08.292.

OpenDroneMap Authors (2020). *ODM - A command line toolkit to generate maps, point clouds, 3D models and DEMs from drone, balloon or kite images*. https://github.com/OpenDroneMap/ODM.

Oquab, M. et al. (2024). "DINOv2: Learning Robust Visual Features without Supervision." In: *Transactions on Machine Learning Research (TMLR)*. URL: https://openreview.net/forum?id=a68SUt6zFt.

Ovrén, H. and P.-E. Forssén (2019). "Trajectory representation and landmark projection for continuous-time structure from motion." In: *The International Journal of Robotics Research (IJRR)* 38.6. DOI: 10.1177/0278364919839765.

Pan, Y., X. Zhong, L. Wiesmann, T. Posewsky, J. Behley, and C. Stachniss (2024). "PIN-SLAM: LiDAR SLAM Using a Point-Based Implicit Neural Representation for Achieving Global Map Consistency." In: *IEEE Transactions on Robotics (T-RO)* 40. DOI: 10.1109/TRO.2024.3422055.

Park, C., P. Moghadam, S. Kim, A. Elfes, C. Fookes, and S. Sridharan (2018). "Elastic LiDAR Fusion: Dense Map-Centric Continuous-Time SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2018.8462915.

Park, S., T. Schöps, and M. Pollefeys (2017). "Illumination Change Robustness in Direct Visual SLAM." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2017.7989525.

Parracho, D. F. R., J. Poças Martins, and E. Barreira (2023). "A Workflow for Photogrammetric and Thermographic Surveys of Buildings with Drones." In: *New Advances in Building Information Modeling and Engineering Management*. Springer Nature Switzerland. ISBN: 978-3-031-30247-3. DOI: 10.1007/978-3-031-30247-3_5.

Pascoe, G., W. Maddern, M. Tanner, P. Piniés, and P. Newman (2017). "NID-SLAM: Robust Monocular SLAM Using Normalised Information Distance." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2017.158.

Patchou, M., J. Tiemann, C. Arendt, S. Böeker, and C. Wietfeld (2022). "Realtime Wireless Network Emulation for Evaluation of Teleoperated Mobile Robots." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR56537.2022.10018773.

Pfister, H., M. Zwicker, J. van Baar, and M. Gross (2000). "Surfels: surface elements as rendering primitives." In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. DOI: 10.1145/344779.344936.

Pfreundschuh, P., H. Oleynikova, C. Cadena, R. Siegwart, and O. Andersson (2024). "COIN-LIO: Complementary Intensity-Augmented LiDAR Inertial Odometry." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA57147.2024.10610938.

Piccinelli, L., Y.-H. Yang, C. Sakaridis, M. Segu, S. Li, L. V. Gool, and F. Yu (2024). "UniDepth: Universal Monocular Metric Depth Estimation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.00963.

Pissanetzky, S. (1984). "Linear Algebraic Equations." In: *Sparse Matrix Technology*. Academic Press. Chap. 2. ISBN: 978-0-12-557580-5. DOI: 10.1016/B978-0-12-557580-5.50007-7.

Polizzi, V., R. Hewitt, J. Hidalgo-Carrió, J. Delaune, and D. Scaramuzza (2022). "Data-Efficient Collaborative Decentralized Thermal-Inertial Odometry." In: *IEEE Robotics and Automation Letters (RA-L)*. DOI: 10.1109/LRA.2022.3194675.

Polyak, B. T. (1964). "Some methods of speeding up the convergence of iteration methods." In: *USSR Computational Mathematics and Mathematical Physics* 4.5. DOI: 10.1016/0041-5553(64)90137-5.

Proudman, A., M. Ramezani, S. T. Digumarti, N. Chebrolu, and M. Fallon (2022). "Towards real-time forest inventory using handheld LiDAR." In: *Journal of Robotics and Autonomous Systems* 157. DOI: 10.1016/j.robot.2022.104240.

Qin, C., H. Ye, C. Pranata, J. Han, S. Zhang, and M. Liu (2020). "LINS: A Lidar-Inertial State Estimator for Robust and Efficient Navigation." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9197567.

Qin, K. (1998). "General Matrix Representations for B-splines." In: *Pacific Conference on Computer Graphics and Applications (PCCGA)*. DOI: 10.1109/PCCGA.1998.731996.

Qu, C., S. Shivakumar, W. Liu, and C. Taylor (2022). "LLOL: Low-Latency Odometry for Spinning Lidars." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA46639.2022.9811605.

Quenzel, J. and S. Behnke (2021). "Real-time Multi-Adaptive-Resolution-Surfel 6D LiDAR Odometry using Continuous-time Trajectory Optimization." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS51168.2021.9636763.

Quenzel, J., J. Horn, S. Houben, and S. Behnke (2018). "Keyframe-based Photometric Online Calibration and Color Correction." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593595.

Quenzel, J., L. T. Mallwitz, B. T. Arnold, and S. Behnke (2024). "LiDAR-Based Registration Against Georeferenced Models for Globally Consistent Allocentric Maps." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR62954.2024.10770000.

Quenzel, J., M. Nieuwenhuisen, D. Droeschel, M. Beul, S. Houben, and S. Behnke (2019). "Autonomous MAV-based Indoor Chimney Inspection with 3D Laser Localization and Textured Surface Reconstruction." In: *Journal of Intelligent & Robotic Systems (JINT)* 93.1. DOI: 10.1007/s10846-018-0791-y.

Quenzel, J., R. A. Rosu, S. Houben, and S. Behnke (2017). "Online Depth Calibration for RGB-D Cameras using Visual SLAM." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2017.8206043.

Quenzel, J., R. A. Rosu, T. Läbe, C. Stachniss, and S. Behnke (2020). "Beyond Photometric Consistency: Gradient-based Dissimilarity for Improving Visual Odometry and Stereo Matching." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9197483.

Rakha, T. and A. Gorodetsky (2018). "Review of Unmanned Aerial System (UAS) applications in the built environment: Towards automated building inspection procedures using drones." In: *Automation in Construction* 93. DOI: 10.1016/j.autcon.2018.05.002.

Ramezani, M., Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon (2020). "The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS45743.2020.9340849.

Ranftl, R., K. Lasinger, D. Hafner, K. Schindler, and V. Koltun (2022). "Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-Shot Cross-Dataset Transfer." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 44.3. DOI: 10.1109/TPAMI.2020.3019967.

Ray, H. M., R. Singer, and N. Ahmed (2022). "A Review of the Operational Use of UAS in Public Safety Emergency Incidents." In: *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*. DOI: 10.1109/ICUAS54217.2022.9836061.

Reinke, A., M. Palieri, B. Morrell, Y. Chang, K. Ebadi, L. Carlone, and A.-A. Agha-Mohammadi (2022). "LOCUS 2.0: Robust and Computationally Efficient Lidar Odometry for Real-Time 3D Mapping." In: *IEEE Robotics and Automation Letters (RA-L)* 7.4. DOI: 10.1109/LRA.2022.3181357.

Reizenstein, J., R. Shapovalov, P. Henzler, L. Sbordone, P. Labatut, and D. Novotny (2021). "Common Objects in 3D: Large-Scale Learning and Evaluation of Real-life 3D Category Reconstruction." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV48922.2021.01072.

Renaut, L., H. Frei, and A. Nüchter (2023). "Lidar Pose Tracking of a Tumbling Spacecraft Using the Smoothed Normal Distribution Transform." In: *Remote Sensing* 15.9. DOI: 10.3390/rs15092286.

Rosinol, A., M. Abate, Y. Chang, and L. Carlone (2020). "Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9196885.

Rosten, E. and T. Drummond (2006). "Machine learning for high-speed corner detection." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/11744023_34.

Rosu, R. A., J. Quenzel, and S. Behnke (2019a). "Reconstruction of Textured Meshes for Fire and Heat Source Detection." In: *Proceedings of the IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. DOI: 10.1109/SSRR.2019.8848943.

– (2019b). "Semi-supervised Semantic Mapping Through Label Propagation with Semantic Texture Meshes." In: *International Journal of Computer Vision (IJCV)* 128.5. DOI: 10.1007/s11263-019-01187-z.

Rosu, R. A., P. Schütt, J. Quenzel, and S. Behnke (2020). "LatticeNet: Fast Point Cloud Segmentation Using Permutohedral Lattices." In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2020.XVI.006.

Rublee, E., V. Rabaud, K. Konolige, and G. Bradski (2011). "ORB: An efficient alternative to SIFT or SURF." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: ICCV.2011.6126544.

Salas-Moreno, R. F., R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison (2013). "SLAM++: Simultaneous Localisation and Mapping at the Level of Objects." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/CVPR.2013.178.

Scharr, H. (2004). "Optimal Filters for Extended Optical Flow." In: *International Workshop on Complex Motion (IWCM)*. DOI: 10.1007/978-3-540-69866-1_2.

Scharstein, D., H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling (2014). "High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth." In: *Proceedings of the German Conference on Pattern Recognition (GCPR)*. DOI: 10.1007/978-3-319-11752-2_3.

Scharstein, D. and R. Szeliski (2002). "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms." In: *International Journal of Computer Vision (IJCV)* 47. DOI: 10.1023/A:1014573219977.

Schauer, J. and A. Nüchter (2018). "The Peopleremover—Removing Dynamic Objects From 3D Point Cloud Data by Traversing a Voxel Occupancy Grid." In: *IEEE Robotics and Automation Letters (RA-L)* 3.3. DOI: 10.1109/LRA.2018.2801797.

Schischmanow, A., D. Dahlke, D. Baumbach, I. Ernst, and M. Linkiewicz (2022). "Seamless Navigation, 3D Reconstruction, Thermographic and Semantic Mapping for Building Inspection." In: *Sensors* 22.13. DOI: 10.3390/s22134745.

Schleich, D. and S. Behnke (2022). "Predictive Angular Potential Field-based Obstacle Avoidance for Dynamic UAV Flights." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS47612.2022.9981677.

Schleich, D., M. Beul, J. Quenzel, and S. Behnke (2021). "Autonomous Flight in Unknown GNSS-denied Environments for Disaster Examination." In: *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS)*. DOI: 10.1109/ICUAS51884.2021.9476790.

Schmid, L., O. Andersson, A. Sulser, P. Pfreundschuh, and R. Siegwart (2023). "Dynablox: Real-time Detection of Diverse Dynamic Objects in Complex Environments." In: 8.10. DOI: 10.1109/LRA.2023.3305239.

Schneider, J., F. Schindler, T. Läbe, and W. Förstner (2012). "Bundle Adjustment for Multi-camera Systems with Points at Infinity." In: *International Arch. Photogramm. Remote Sens. Spatial Inf. Sci. (ISPRS)*. DOI: 10.5194/isprsannals-I-3-75-2012.

Schönberger, J. L. and J.-M. Frahm (2016). "Structure-from-Motion Revisited." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2016.445.

Schönberger, J., E. Zheng, J. Frahm, and M. Pollefeys (2016). "Pixelwise View Selection for Unstructured Multi-View Stereo." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-319-46487-9_31.

Schubert, E. and M. Gertz (2018). "Numerically Stable Parallel Computation of (Co-)Variance." In: *Proceedings of the International Conference on Scientific and Statistical Database Management*. DOI: 10.1145/3221269.3223036.

Segal, A., D. Haehnel, and S. Thrun (2009). "Generalized-ICP." In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2009.V.021.

Shan, T. and B. Englot (2018). "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8594299.

Shan, T., B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus (2020). "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping." In:

Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). DOI: 10.1109/IROS45743.2020.9341176.

Shan, T., B. Englot, C. Ratti, and D. Rus (2021). "LVI-SAM: Tightly-coupled LiDAR-Visual-Inertial Odometry via Smoothing and Mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48506.2021.9561996.

Shen, H., Z. Wu, Y. Hui, W. Wang, Q. Lyu, T. Deng, Y. Zhu, B. Tian, and D. Wang (2025). "CTE-MLO: Continuous-Time and Efficient Multi-LiDAR Odometry With Localizability-Aware Point Cloud Sampling." In: *IEEE Transactions on Field Robotics (T-FR)* 2. DOI: 10.1109/TFR.2025.3543142.

Shi, J. and C. Tomasi (1994). "Good features to track." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.1994.323794.

Shoemake, K. (1985). "Animating Rotation with Quaternion Curves." In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. DOI: 10.1145/325334.325242.

Smith, M., I. Baldwin, W. Churchill, R. Paul, and P. Newman (2009). "The New College Vision and Laser Data Set." In: *The International Journal of Robotics Research (IJRR)* 28.5. DOI: 10.1177/0278364909103911.

Solà, J., J. Deray, and D. Atchuthan (2018). *A micro Lie theory for state estimation in robotics*. arXiv: 1812.01537 [cs.RO].

Sommer, C., V. Usenko, D. Schubert, N. Demmel, and D. Cremers (2020). "Efficient Derivative Computation for Cumulative B-Splines on Lie Groups." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR42600.2020.01116.

Splietker, M. and S. Behnke (2023). "Rendering the Directional TSDF for Tracking and Multi-Sensor Registration with Point-To-Plane Scale ICP." In: *Journal of Robotics and Autonomous Systems* 162. DOI: 10.1016/j.robot.2022.104337.

Steinbrücker, F., J. Sturm, and D. Cremers (2014). "Volumetric 3D mapping in real-time on a CPU." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2014.6907127.

Stückler, J. and S. Behnke (2014). "Multi-resolution surfel maps for efficient dense 3D modeling and tracking." In: *Journal of Visual Communication and Image Representation* 25.1. DOI: 10.1016/j.jvcir.2013.02.008.

Stückler, J., B. Waldvogel, H. Schulz, and S. Behnke (2014). "Dense Real-Time Mapping of Object-Class Semantics from RGB-D Video." In: *Journal of Real-Time Image Processing (JRTIP)* 10. DOI: 10.1007/s11554-013-0379-5.

Sturm, J., N. Engelhard, F. Endres, W. Burgard, and D. Cremers (2012). "A Benchmark for the evaluation of RGB-D SLAM systems." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2012.6385773.

Sun, C., M. Sun, and H.-T. Chen (2022). "Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52688.2022.00538.

Sun, J., G. Yuan, L. Song, and H. Zhang (2024). "Unmanned Aerial Vehicles (UAVs) in Landslide Investigation and Monitoring: A Review." In: *Drones* 8.1. DOI: 10.3390/drones8010030.

Sun, L., Z. Yan, A. Zaganidis, C. Zhao, and T. Duckett (2018). "Recurrent-OctoMap: Learning State-Based Map Refinement for Long-Term Semantic Mapping With 3D-Lidar Data." In: *IEEE Robotics and Automation Letters (RA-L)* 3.4. DOI: 10.1109/LRA.2018.2856268.

Surmann, H. et al. (2024). "Lessons from robot-assisted disaster response deployments by the German Rescue Robotics Center task force." In: *Journal of Field Robotics (JFR)* 41.3. DOI: 10.1002/rob.22275.

Sutherland, N., S. Marsh, G. Priestnall, P. Bryan, and J. Mills (2023). "InfraRed Thermography and 3D-Data Fusion for Architectural Heritage: A Scoping Review." In: *Remote Sensing* 15.9. DOI: 10.3390/rs15092422.

Talbot, W., J. Nubert, T. Tuna, C. Cadena, F. Dümbgen, J. Tordesillas, T. Barfoot, and M. Hutter (2025). "Continuous-Time State Estimation Methods in Robotics: A Survey." In: *IEEE Transactions on Robotics (T-RO)* 41. DOI: 10.1109/TRO.2025.3593079.

Tancik, M. et al. (2023). "Nerfstudio: A Modular Framework for Neural Radiance Field Development." In: *International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. DOI: 10.1145/3588432.3591516.

Tao, Y., Y. Bhalgat, L. F. T. Fu, M. Mattamala, N. Chebrolu, and M. Fallon (2024). "SiLVR: Scalable Lidar-Visual Reconstruction with Neural Radiance Fields for Robotic Inspection." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA57147.2024.10611278.

Taylor, Z., J. Nieto, and D. Johnson (2015). "Multi-modal sensor calibration using a gradient orientation measure." In: *JFR* 32.5. DOI: 10.1002/rob.21523.

Thrun, S., W. Burgard, and D. Fox (2005). *Probabilistic Robotics*. The MIT Press. ISBN: 9780262201629.

Tomasi, C. and T. Kanade (1992). "Shape and motion from image streams under orthography: a factorization method." In: *International Journal of Computer Vision (IJCV)* 9.2. DOI: 10.1007/BF00129684.

Tong, C., P. Furgale, and T. D. Barfoot (2013). "Gaussian Process Gauss–Newton for non-parametric simultaneous localization and mapping." In: *The International Journal of Robotics Research (IJRR)* 32.5. DOI: 10.1177/0278364913478672.

Transtrum, M. and J. Sethna (2012). *Improvements to the Levenberg-Marquardt algorithm for nonlinear least-squares minimization*. arXiv: 1201.5885 [physics.data-an].

Triggs, B., P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon (2000). "Bundle Adjustment — A Modern Synthesis." In: *Vision Algorithms: Theory and Practice*. DOI: 10.1007/3-540-44480-7_21.

Tuna, T., J. Nubert, Y. Nava, S. Khattak, and M. Hutter (2024). "X-ICP: Localizability-Aware LiDAR Registration for Robust Localization in Extreme Environments." In: *IEEE Transactions on Robotics (T-RO)* 40. DOI: 10.1109/TRO.2023.3335691.

Tuna, T., J. Nubert, P. Pfreundschuh, C. Cadena, S. Khattak, and M. Hutter (2025). "Informed, Constrained, Aligned: A Field Analysis on Degeneracy-Aware Point

Cloud Registration in the Wild." In: *IEEE Transactions on Robotics (T-RO)* 2, pp. 485–515. DOI: 10.1109/TFR.2025.3576053.

Umeyama, S. (1991). "Least-Squares Estimation of Transformation Parameters Between Two Point Patterns." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 13.4. DOI: 10.1109/34.88573.

Usenko, V., N. Demmel, and D. Cremers (2018). "The Double Sphere Camera Model." In: *Proceedings of the International Conference on 3D Vision (3DV)*. DOI: 10.1109/3DV.2018.00069.

Usenko, V., N. Demmel, D. Schubert, J. Stückler, and D. Cremers (2020). "Visual-Inertial Mapping With Non-Linear Factor Recovery." In: *IEEE Robotics and Automation Letters (RA-L)* 5.2. DOI: 10.1109/LRA.2019.2961227.

USSF, National Coordination Office for Space-Based Positioning, Navigation, and Timing, and NOAA (2022). *GPS Accuracy*. https://www.gps.gov/systems/gps/performance/accuracy/.

Utreras, F. and M. Varas (1991). "Monotone interpolation of scattered data in R^s." In: *Constructive Approximation* 7.1. DOI: 10.1007/BF01888146.

Valentin, J., S. Sengupta, J. Warrell, A. Shahrokni, and P. Torr (2013). "Mesh Based Semantic Modelling for Indoor and Outdoor Scenes." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR.2013.269.

Vespa, E., N. Nikolov, M. Grimm, L. Nardi, P. H. J. Kelly, and S. Leutenegger (2018). "Efficient Octree-Based Volumetric SLAM Supporting Signed-Distance and Occupancy Mapping." In: *IEEE Robotics and Automation Letters (RA-L)* 3.2. DOI: 10.1109/LRA.2018.2792537.

Visser, M., S. Stramigioli, and C. Heemskerk (2006). "Cayley-Hamilton for roboticists." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2006.281911.

Vizzo, I., X. Chen, N. Chebrolu, J. Behley, and C. Stachniss (2021). "Poisson Surface Reconstruction for LiDAR Odometry and Mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48506.2021.9562069.

Vizzo, I., T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss (2023). "KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way." In: *IEEE Robotics and Automation Letters (RA-L)* 8.2. DOI: 10.1109/LRA.2023.3236571.

Waechter, M., N. Moehrle, and M. Goesele (2014). "Let There Be Color! Large-Scale Texturing of 3D Reconstructions." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-319-10602-1_54.

Wan, E. A. and R. Van Der Merwe (2000). "The unscented Kalman filter for nonlinear estimation." In: *IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium (AS-SPCC)*. DOI: 10.1109/ASSPCC.2000.882463.

Wang, C., S. Lucey, F. Perazzi, and O. Wang (2019). "Web Stereo Video Supervision for Depth Prediction from Dynamic Scenes." In: *Proceedings of the International Conference on 3D Vision (3DV)*. DOI: 10.1109/3DV.2019.00046.

Wang, F., Q. Zhu, D. Chang, Q. Gao, J. Han, T. Zhang, R. Hartley, and M. Pollefeys (2024a). *Learning-based Multi-View Stereo: A Survey*. arXiv: 2408.15235 [cs.CV].

Wang, J., N. Karaev, C. Rupprecht, and D. Novotny (2024b). "VGGSfM: Visual Geometry Grounded Deep Structure from Motion." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.02049.

Wang, J., X. Guan, Z. Sun, T. Shen, D. Huang, F. Liu, and H. Cui (2025). "OMEGA: Efficient Occlusion-Aware Navigation for Air-Ground Robots in Dynamic Environments via State Space Model." In: *IEEE Robotics and Automation Letters (RA-L)* 10.2. DOI: 10.1109/LRA.2024.3518076.

Wang, S., V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud (2024c). "DUSt3R: Geometric 3D Vision Made Easy." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.01956.

Wang, Y., H. Chen, S. Zhang, and W. Lu (2022). "Automated camera-exposure control for robust localization in varying illumination environments." In: *Autonomous Robots* 46 (4). DOI: 10.1007/s10514-022-10036-x.

Wei, X., K. Zhang, S. Bi, H. Tan, F. Luan, V. Deschaintre, K. Sunkavalli, H. Su, and Z. Xu (2024). *MeshLRM: Large Reconstruction Model for High-Quality Mesh*. arXiv: 2404.12385 [cs.CV].

Weinzaepfel, P., T. Lucas, V. Leroy, Y. Cabon, V. Arora, R. Brégier, G. Csurka, L. Antsfeld, B. Chidlovskii, and J. Revaud (2023). "CroCo v2: Improved Cross-view Completion Pre-training for Stereo Matching and Optical Flow." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV51070.2023.01647.

Wen, B., J. Tremblay, V. Blukis, S. Tyree, T. Müller, A. Evans, D. Fox, J. Kautz, and S. Birchfield (2023). "BundleSDF: Neural 6-DoF Tracking and 3D Reconstruction of Unknown Objects." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52729.2023.00066.

Wen, W., Y. Zhou, G. Zhang, S. Fahandezh-Saadi, X. Bai, W. Zhan, M. Tomizuka, and L.-T. Hsu (2020). "UrbanLoco: A Full Sensor Suite Dataset for Mapping and Localization in Urban Scenes." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA40945.2020.9196526.

Whelan, T., H. Johannsson, M. Kaess, J. J. Leonard, and J. McDonald (2013). "Robust real-time visual odometry for dense RGB-D mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2013.6631400.

Whelan, T., S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison (2015). "ElasticFusion: Dense SLAM without a pose graph." In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2015.XI.001.

Wu, Y., D. Hu, M. Wu, and X. Hu (2006). "A Numerical-Integration Perspective on Gaussian Filters." In: *IEEE Transactions on Signal Processing* 54.8. DOI: 10.1109/TSP.2006.875389.

Wu, Y., D. J. Yoon, K. Burnett, S. Kammel, Y. Chen, H. Vhavle, and T. D. Barfoot (2023). "Picking up Speed: Continuous-Time Lidar-Only Odometry Using Doppler Velocity Measurements." In: *IEEE Robotics and Automation Letters (RA-L)* 8.1. DOI: 10.1109/LRA.2022.3226068.

Wu, Y., J. Y. Lee, C. Zou, S. Wang, and D. Hoiem (2025). "MonoPatchNeRF: Improving Neural Radiance Fields with Patch-Based Monocular Guidance." In: *Proceedings of the International Conference on 3D Vision (3DV)*. DOI: 10.1109/3DV66043.2025.00049.

Xia, W. and J.-H. Xue (2023). "A Survey on Deep Generative 3D-aware Image Synthesis." In: *ACM Computing Surveys* 56.4. DOI: 10.1145/3626193.

Xie, D., S. Bi, Z. Shu, K. Zhang, Z. Xu, Y. Zhou, S. Pirk, A. Kaufman, X. Sun, and H. Tan (2024). "LRM-Zero: Training Large Reconstruction Models with Synthesized Data." In: *Advances in Neural Information Processing Systems (NeurIPS)*. URL: https://openreview.net/forum?id=MtRvzJBsBA.

Xie, Z., X. Yang, Y. Yang, Q. Sun, Y. Jiang, H. Wang, Y. Cai, and M. Sun (2023). "S3IM: Stochastic Structural SIMilarity and Its Unreasonable Effectiveness for Neural Fields." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV51070.2023.01652.

Xu, W., Y. Cai, D. He, J. Lin, and F. Zhang (2022). "FAST-LIO2: Fast Direct LiDAR-Inertial Odometry." In: *IEEE Transactions on Robotics (T-RO)* 38.4. DOI: 10.1109/TRO.2022.3141876.

Xu, W. and F. Zhang (2021). "FAST-LIO: A Fast, Robust LiDAR-Inertial Odometry Package by Tightly-Coupled Iterated Kalman Filter." In: *IEEE Robotics and Automation Letters (RA-L)* 6.2. DOI: 10.1109/LRA.2021.3064227.

Xue, W., L. Zhang, X. Mou, and A. C. Bovik (2014). "Gradient Magnitude Similarity Deviation: A Highly Efficient Perceptual Image Quality Index." In: *IEEE Transactions on Image Processing* 23.2. DOI: 10.1109/TIP.2013.2293423.

Yan, C., D. Qu, D. Xu, B. Zhao, Z. Wang, D. Wang, and X. Li (2024). "GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.01853.

Yang, L., B. Kang, Z. Huang, Z. Zhao, X. Xu, J. Feng, and H. Zhao (2024). "Depth Anything V2." In: *Advances in Neural Information Processing Systems (NeurIPS)*. URL: https://openreview.net/forum?id=cFTi3gLJ1X.

Yang, S., Y. Huang, and S. Scherer (2017). "Semantic 3D occupancy mapping through efficient high order CRFs." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2017.8202212.

Ye, H., Y. Chen, and M. Liu (2019). "Tightly Coupled 3D Lidar Inertial Odometry and Mapping." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2019.8793511.

Ye, T., Q. Wu, J. Deng, G. Liu, L. Liu, S. Xia, L. Pang, W. Yu, and L. Pei (2024). "Thermal-NeRF: Neural Radiance Fields from an Infrared Camera." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS58592.2024.10802480.

Yu, Z., T. Sattler, and A. Geiger (2024). "Gaussian Opacity Fields: Efficient Adaptive Surface Reconstruction in Unbounded Scenes." In: *ACM Transactions on Graphics* 43.6. DOI: 10.1145/3687937.

Yuan, C., W. Xu, X. Liu, X. Hong, and F. Zhang (2022). "Efficient and Probabilistic Adaptive Voxel Mapping for Accurate Online LiDAR Odometry." In: *IEEE Robotics and Automation Letters (RA-L)* 7.3. DOI: 10.1109/LRA.2022.3187250.

Yuan, Y. and A. Nüchter (2024). "Uni-Fusion: Universal Continuous Mapping." In: *IEEE Transactions on Robotics (T-RO)* 40. DOI: 10.1109/TRO.2024.3351548.

Yuan, Z., F. Lang, T. Xu, R. Ming, C. Zhao, and X. Yang (2025). "Semi-Elastic LiDAR-Inertial Odometry." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA55743.2025.11128078.

Zaganidis, A., L. Sun, T. Duckett, and G. Cielniak (2018). "Integrating Deep Semantic Segmentation Into 3D Point Cloud Registration." In: *IEEE Robotics and Automation Letters (RA-L)* 3.4. DOI: 10.1109/LRA.2018.2848308.

Zhang, C., Z. Li, Y. Cheng, R. Cai, H. Chao, and Y. Rui (2015). "MeshStereo: A global stereo model with mesh alignment regularization for view interpolation." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV.2015.238.

Zhang, J., M. Kaess, and S. Singh (2016). "On degeneracy of optimization-based state estimation problems." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2016.7487211.

Zhang, J. and S. Singh (2014). "LOAM: Lidar Odometry and Mapping in Real-time." In: *Proceedings of Robotics: Science and Systems (RSS)*. DOI: 10.15607/RSS.2014.X.007.

Zhang, K., S. Bi, H. Tan, Y. Xiangli, N. Zhao, K. Sunkavalli, and Z. Xu (2024a). "GS-LRM: Large Reconstruction Model for 3D Gaussian Splatting." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. DOI: 10.1007/978-3-031-72670-5_1.

Zhang, L., M. Camurri, D. Wisth, and M. Fallon (2021). *Multi-Camera LiDAR Inertial Extension to the Newer College Dataset*. arXiv: 2112.08854 [cs.RO].

Zhang, T., X. Zhang, Z. Liao, X. Xia, and Y. Li (2024b). "AS-LIO: Spatial Overlap Guided Adaptive Sliding Window LiDAR-Inertial Odometry for Aggressive FOV Variation." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS58592.2024.10801561.

Zhang, Y., Y. Tian, W. Wang, G. Yang, Z. Li, F. Jing, and M. Tan (2023a). "RI-LIO: Reflectivity Image Assisted Tightly-Coupled LiDAR-Inertial Odometry." In: *IEEE Robotics and Automation Letters (RA-L)* 8.3. DOI: 10.1109/LRA.2023.3243528.

Zhang, Y., Z. Zhu, and D. Du (2023b). "OccFormer: Dual-path Transformer for Vision-based 3D Semantic Occupancy Prediction." In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. DOI: 10.1109/ICCV51070.2023.00865.

Zhang, Z. and D. Scaramuzza (2018). "A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS.2018.8593941.

Zhao, S., H. Zhang, P. Wang, L. Nogueira, and S. Scherer (2021). "Super Odometry: IMU-centric LiDAR-Visual-Inertial Estimator for Challenging Environments." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS51168.2021.9635862.

Zhao, S. et al. (2024). "SubT-MRS Dataset: Pushing SLAM Towards All-weather Environments." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52733.2024.02137.

Zheng, C., Q. Zhu, W. Xu, X. Liu, Q. Guo, and F. Zhang (2022). "FAST-LIVO: Fast and Tightly-coupled Sparse-Direct LiDAR-Inertial-Visual Odometry." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. DOI: 10.1109/IROS47612.2022.9981107.

Zheng, X. and J. Zhu (2024a). *Traj-LIO: A Resilient Multi-LiDAR Multi-IMU State Estimator Through Sparse Gaussian Process*. arXiv: 2402.09189 [cs.RO].

– (2024b). "Traj-LO: In Defense of LiDAR-Only Odometry Using an Effective Continuous-Time Trajectory." In: *IEEE Robotics and Automation Letters (RA-L)* 9.2. DOI: 10.1109/LRA.2024.3352360.

Zheng, X., Z. Moratto, M. Li, and A. I. Mourikis (2017). "Photometric patch-based visual-inertial odometry." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA.2017.7989372.

Zheng, Y., S. Lin, C. Kambhamettu, J. Yu, and S. B. Kang (2009). "Single-Image Vignetting Correction." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 31.12. DOI: 10.1109/TPAMI.2008.263.

Zhong, X., Y. Pan, J. Behley, and C. Stachniss (2023). "SHINE-Mapping: Large-Scale 3D Mapping Using Sparse Hierarchical Implicit Neural Representations." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. DOI: 10.1109/ICRA48891.2023.10160907.

Zhou, B., J. Wu, Y. Pan, and C. Lu (2024). "ATI-CTLO: Adaptive Temporal Interval-Based Continuous-Time LiDAR-Only Odometry." In: *IEEE Robotics and Automation Letters (RA-L)* 9.12. DOI: 10.1109/LRA.2024.3486233.

Zhou, Q.-Y. and V. Koltun (2014). "Color Map Optimization for 3D Reconstruction with Consumer Depth Cameras." In: *ACM Transactions on Graphics* 33.4. DOI: 10.1145/2601097.2601134.

Zhu, K., X. Jiang, Z. Fang, Y. Gao, H. Fujita, and J.-N. Hwang (2021). "Photometric transfer for direct visual odometry." In: *Knowledge-Based Systems* 213. DOI: 10.1016/j.knosys.2020.106671.

Zhu, X., H. Zhou, T. Wang, F. Hong, W. Li, Y. Ma, H. Li, R. Yang, and D. Lin (2022a). "Cylindrical and Asymmetrical 3D Convolution Networks for LiDAR-Based Perception." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)* 44.10. DOI: 10.1109/TPAMI.2021.3098789.

Zhu, Z., J. Lu, M. Wang, S. Zhang, R. R. Martin, H. Liu, and S.-M. Hu (2018). "A Comparative Study of Algorithms for Realtime Panoramic Video Blending." In: *IEEE Transactions on Image Processing* 27.6. DOI: 10.1109/TIP.2018.2808766.

Zhu, Z., S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. Oswald, and M. Pollefeys (2022b). "NICE-SLAM: Neural Implicit Scalable Encoding for SLAM." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. DOI: 10.1109/CVPR52688.2022.01245.

Zollhöfer, M., P. Stotko, A. Görlitz, C. Theobalt, M. Nießner, R. Klein, and A. Kolb (2018). "State of the Art on 3D Reconstruction with RGB-D Cameras." In: *Computer Graphics Forum*. DOI: 10.1111/cgf.13386.