

# Photometric LiDAR and RGB-D Bundle Adjustment

Luca Di Giammarino Emanuele Giacomini Leonardo Brizi Omar Salem Giorgio Grisetti

**Abstract**—The joint optimization of the sensor trajectory and 3D map is a crucial characteristic of Simultaneous Localization and Mapping (SLAM) systems. To achieve this, the gold standard is Bundle Adjustment (BA). Modern 3D LiDARs now retain higher resolutions that enable the creation of point cloud images resembling those taken by conventional cameras. Nevertheless, the typical effective global refinement techniques employed for RGB-D sensors are not widely applied to LiDARs. This paper presents a novel BA photometric strategy that accounts for both RGB-D and LiDAR in the same way. Our work can be used on top of any SLAM/GNSS estimate to improve and refine the initial trajectory. We conducted different experiments using these two depth sensors on public benchmarks. Our results show that our system performs on par or better compared to other state-of-the-art ad-hoc SLAM/BA strategies, free from data association and without making assumptions about the environment. In addition, we present the benefit of jointly using RGB-D and LiDAR within our unified method. We finally release an open-source CUDA/C++ implementation<sup>1</sup>.

## I. INTRODUCTION

SLAM has gained significant popularity in the field of robotics, and thanks to approximately three decades of research, there are now effective solutions available. As SLAM is widely used in various sectors, such as autonomous driving, augmented reality, and space exploration, it continues to attract significant attention from academia and industry. Modern SLAM systems typically comprise two key components: a front-end that estimates sensor odometry in real-time and a back-end that optimizes previous sensor poses and, eventually, the 3D map. The gold standard for the back-end optimization is BA [24]. Photometric (or direct) approaches have been used by the computer vision community to tackle the SLAM or Structure from Motion (SfM) problems. The direct techniques address registration by minimizing the pixel-wise error between image pairs. By not relying on specific features and having the potential of operating at subpixel resolution on the entire image, direct approaches do not require explicit data association and boost the registration accuracy [15], [20]. Although these methods have been successfully employed on monocular, stereo, or RGB-D sensors, their use on 3D LiDAR data has not been widely exploited. Della Corte *et al.* [4] presented a general photometric registration methodology that extends direct approaches to different projective models and enhances the optimization robustness by considering additional channels such as normals. Nowadays, 3D LiDARs offer up to 128

All authors are with the Department of Computer, Control, and Management Engineering "Antonio Ruberti", Sapienza University of Rome, Italy, Email: {digiammarino, giacomini, brizi, salem, grisetti}@diag.uniroma1.it.

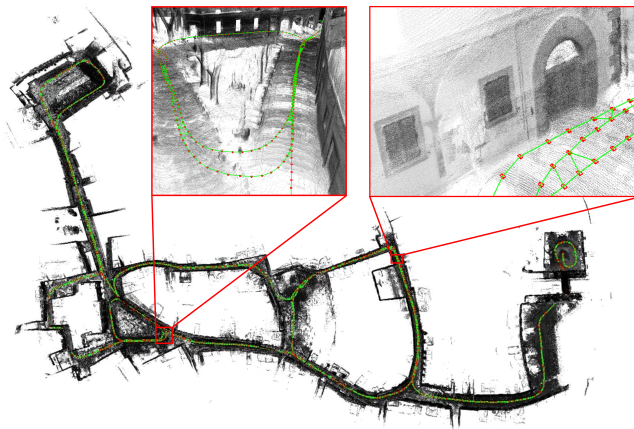


Fig. 1: Reconstruction of Viterbo city-center (Italy) using our data recorded with an OS0-128. The trajectory, which is about 2 km long, has been estimated first with MD-SLAM [10] and then refined with our photometric BA strategy. This image highlights both the global and local consistencies. We show reconstruction details with multiple scans of the same places acquired over time.

channels of resolution, measuring more than 5M points per second. The enhancement of density, the increment in vertical resolution, and the capacity to measure the reflectivity of a surface make it logical to render a scan onto a panoramic image. This consideration makes photometric approaches attractive for 3D LiDARs.

The main contribution of this paper is a unified photometric BA strategy that works for both RGB-D and LiDAR. Our method aims to refine the trajectory coming from a SLAM/GNSS system to maximize its photometric consistency. Our approach implicitly addresses the data association and straightforwardly supports multiple heterogeneous sensors. We performed a comparative evaluation of benchmark data concerning state-of-the-art sensor-specific refinement strategies and SLAM algorithms. Results show that our simple optimization schema is very effective, performing on par or better than methods specialized for RGB-D and LiDAR data. We also demonstrate how our photometric BA strategy can be improved by fusing 3D LiDAR and RGB-D. We release an open-source CUDA/C++ implementation of this work<sup>1</sup>.

Fig. 1 shows a reconstruction of the Viterbo city-center (Italy) using our self-recorded data. From the detailed views, it is possible to appreciate the fine map resolution after performing our BA strategy.

<sup>1</sup>[https://github.com/digiamm/ba\\_md\\_slam](https://github.com/digiamm/ba_md_slam)

## II. RELATED WORK

Approaches for global refinement such as BA are widely used in Visual SLAM and SfM systems but are less common for LiDAR. In this work, we provide a unified photometric global registration method for both RGB-D and LiDAR that can improve the accuracy of the trajectory - and hence the map - obtained by standard SLAM systems that rely on Pose-Graph Optimization (PGO). PGO formulation reduces the optimization problem's size but approximates the original problem by marginalizing the projective observations.

In the RGB-D SLAM field, BA can be classified into two categories: *direct* and *indirect*. In the following, we will explain the difference between these two paradigms and review the state-of-the-art of both approaches. We finally discuss BA applications in LiDAR SLAM.

Indirect methods are based on feature detection and matching between images. The camera poses and the 3D structure are then estimated by minimizing the reprojection error of this set of feature points. These methods are preferred in SLAM applications because they are faster since they operate on much fewer data. The data reduction, employed by extracting features, renders indirect methods less sensitive to calibration and synchronization issues [20]. State-of-the-art indirect SLAM implementations perform windowed BA to refine the map under construction in a neighborhood of the current sensor location. Global BA is invoked upon loop closures on the entire trajectory. To keep the size of the problem tractable during global BA, the trajectory is subsampled in a set of keyframes, and only some salient feature points are considered in the optimization [16], [18].

Direct methods, on the other hand, also known as photometric, do not rely on feature detection and matching. Instead, they directly minimize the photometric error between overlapping images using all information available in the image. More specifically, this error is calculated as the difference between the measured and the predicted pixel intensities. Computing such a prediction relies on an estimate of the 3D structure captured by the camera and its parameters. Delaunoy and Pollefeys propose an offline dense 3D reconstruction methodology, in which refinement of the 3D scene and camera parameters is performed simultaneously, with the primary goal of minimizing photometric error [6]. The scene is represented by triangular meshes, requiring frequent remeshing during the optimization. This makes the approach computationally demanding. Goldlücke *et al.* presents a new variational framework that enables the precise representation of a 3D scene by estimating a super-resolution curved surface. Unlike the previous work using typical triangular meshes, the authors utilize a smooth surface [11]. Decoupling the camera motion from the optimization process leads to a convex objective functional, resulting in an improved estimate and high-quality texture output. Slavcheva *et al.*, instead of building a mesh, perform pairwise alignment registering directly two consecutive signed distance functions (SDFs) generated from the depth images. This kind of registration is then used also as global

refinement [22]. The direct methods presented previously are computationally heavy, being mainly used for offline 3D reconstructions but not suitable for on-line estimation.

To enhance the computation, some researchers started to exploit the decomposability of the problem by leveraging its structure. Others investigated data reduction strategies to reduce the problem's size while preserving enough information to obtain an accurate estimate. To reduce the problem size, Hatem Alismail *et al.* leverage the fact that most of the image points do not contribute equally to the optimization and consider in the error function only pixels with reasonable gradient [1].

To reduce computational demand for large scale BA problems, Eriksson *et al.* propose a consensus-based optimization to parallelize Bundle Adjustment in SfM applications [8]. In a more recent development, Demmel *et al.* proposed a novel solution in Distributed BA [7]. Specifically, the author broke down the original problem into smaller, more manageable subparts using the k-means clustering method. To allow more efficient processing, the resulting subgraphs are structured into relative well-constrained connected segments.

In between the class of direct and indirect methods, we find the work of Forster *et al.* [9], which proposes a hybrid method to estimate the camera's motion. First, an initial guess of the sensor location is computed by minimizing the reprojection error of the world points. Then the estimate is refined by minimizing the photometric error of the patches around the feature points. The final map refinement step is done by performing direct BA.

In the last years, thanks to the technology enhancement, the community focused on embedding BA refinement in SLAM applications. To combine the accuracy of direct methods with the robustness of feature-based ones, hybrid approaches gained traction. These methods mix both direct and indirect error terms in their optimization strategies. One such instance is Bundle Fusion [5], which refine the global estimate by interleaving feature-based and photometric BA. The photometric refinement does not take into account the structure, but only the camera poses. Similarly, in BAD-SLAM the motion estimation and global refinement processes are accomplished by minimizing a cost function that accounts for geometric and photometric errors [20]. The global refinement process is broken down into three main steps: first, the 3D scene modeled by surfel is refined, then the camera poses are optimized, keeping the model fixed, and finally, the camera's intrinsics are refined.

In parallel, the community approached LiDAR-based SLAM by seeking alternative representations for the dense 3D point clouds. Given the accuracy of these measurements, the robotics community addressed the problem of building a map incrementally registering new scans.

Many registration techniques have been exploited using LiDAR data. These include 3D salient features [14], [28], subsampled clouds [25] or Normal Distributed Transform (NDT) [23]. Nowadays, LiDAR Odometry and Mapping (LOAM) is perhaps one of the most popular methods for LiDAR odometry [28], [29]. It extracts distinct features

corresponding to surfaces and corners, then used to determine point-to-plane and point-to-line distances to a voxel grid-based map representation. A revised approach (Lego-LOAM) has been suggested [21], which takes advantage of a ground surface in its segmentation and optimization steps. Odometry estimation techniques, or more generally 3D point clouds registration routines, coupled with place recognition and PGO show satisfying results within LiDAR SLAM [10], [21]. This makes PGO the gold standard optimization method in LiDAR community. A pose-graph represents the trajectory, and observations between pairs of poses along the path are computed by registering the two overlapping clouds. Hence, an observation is a relative transform and potentially a covariance matrix. With this approximation, the constraints between the poses can be represented in a relatively compact manner. The optimum of a PGO is the configuration of poses that is maximally consistent with the transforms in the measurements. Albeit efficient, PGO approaches operate on approximating the original problem since pairwise measurements are computed once during the SLAM phase and never revised. Unavoidable drifts will accumulate, and wrong behavior of the place recognition might lead to inconsistent graphs as shown in [27].

In order to remove these inconsistencies, recently Liu and Zhang presented a global geometrical optimization methodology that considers cloud measurements. In particular, it formulates a cost function based on LOAM features (i.e., edges and planes) and globally optimizes the trajectory to maximize the features' overlap. This approach performs a static data association based on the co-visibility of landmarks; thus, it requires a good initial guess to operate.

In recent years, the vertical resolution of modern 3D LiDARs increased, and these devices provide scans that resemble more and more dense panoramic images. This allows us to transfer results about direct global refinement from vision to LiDAR. In this work, we present a unified BA strategy that works independently for LiDAR and RGB-D in the same way. By operating directly on images, our method constantly refines the data association during the optimization; hence it is less sensitive to poor initial guesses. Our algorithm's capabilities are demonstrated through quantitative and qualitative analyses, consistently improving the initial estimates provided by any SLAM systems.

### III. BASICS

Our method models the problem of photometric BA as an optimization problem. In this section, we present some notation and review some concepts used in the remainder of the paper: parameterization of the transformations and projective models for RGB-D and LiDAR.

We parameterize the sensor motion to the class of rigid body motions forming the special euclidean group  $\mathbb{SE}(3)$ . A common representation for rigid body motions is a transformation matrix  $\mathbf{X}$ ,

$$\mathbf{X} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \quad \mathbf{X}^{-1} = \begin{bmatrix} \mathbf{R}^T & -\mathbf{R}^T \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \quad (1)$$

here  $\mathbf{R} \in \mathbb{SO}(3)$  is a  $3 \times 3$  rotation matrix and  $\mathbf{t}$  is a 3D translation vector.

Our method deals with both LiDAR and RGB-D cameras homogeneously. The only adaptation to be done for a specific sensor is to define the appropriate projective model. These two models are illustrated in the remainder of this section.

A projection is a mapping  $\pi : \mathbb{R}^3 \rightarrow \Gamma \subset \mathbb{R}^2$  from a world point  $\mathbf{p} = [x, y, z]^T$  to image coordinates  $\mathbf{u} = [u, v]^T$ . Knowing the depth or the range  $d$  of an image point  $\mathbf{u}$ , we can calculate the inverse mapping  $\pi^{-1} : \Gamma \times \mathbb{R} \rightarrow \mathbb{R}^3$ , more explicitly  $\mathbf{p} = \pi^{-1}(\mathbf{u}, d)$ . We will refer to this operation as unprojection. We remind that in the case of LiDAR,  $d$  represents the *range*, namely the distance between the endpoint and the origin of the observer. Differently for RGB-D,  $d$  represents the *depth*, which is the distance between the endpoint and the image plane. For compactness, we will only refer to  $d$  as depth in the remainder of the paper.

**Pinhole Model (RGB-D):** Let  $\mathbf{K}$  be the camera matrix. Then, the pinhole projection of a point  $\mathbf{p}$  is computed as

$$\pi_p(\mathbf{p}) = \phi(\mathbf{K}\mathbf{p}), \quad (2)$$

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (3)$$

$$\phi(\mathbf{v}) = \frac{1}{v_z} \begin{bmatrix} v_x \\ v_y \end{bmatrix}, \quad (4)$$

with the intrinsic camera parameters for the focal length  $f_x$ ,  $f_y$  and the principle point  $c_x$ ,  $c_y$ . The function  $\phi(\mathbf{v})$  is the homogeneous normalization with  $\mathbf{v} = [v_x, v_y, v_z]^T$ .

**Spherical Model (LiDAR):** Let  $\mathbf{K}$  be a camera matrix in the form of Eq. (3), where  $f_x$  and  $f_y$  specify respectively the resolution of azimuth and elevation and  $c_x$  and  $c_y$  their offset in pixels. The function  $\psi$  maps a 3D point to azimuth and elevation. Thus the spherical projection of a point is given by

$$\pi_s(\mathbf{p}) = \mathbf{K}_{[1,2]} \psi(\mathbf{p}), \quad (5)$$

$$\psi(\mathbf{v}) = \begin{bmatrix} \text{atan2}(v_y, v_x) \\ \text{atan2}\left(v_z, \sqrt{v_x^2 + v_y^2}\right) \\ 1 \end{bmatrix}. \quad (6)$$

In the spherical model  $\mathbf{K}_{[1,2]} \in \mathbb{R}^{2 \times 3}$ , being the third row in  $\mathbf{K}$  suppressed.

### IV. OUR APPROACH

The goal of our approach is to compute the set of sensor poses  $\mathbf{X}_{1:N}$  whose photometric error between overlapping frames is minimized. The input of our system is a set of triplets  $\langle \mathbf{X}_i, \mathcal{I}^g, \mathcal{I}^d \rangle$ , containing the initial guess of the sensor pose, the grayscale/intensity image  $\mathcal{I}^g$  and the depth image  $\mathcal{I}^d$ . The workflow of our method is illustrated in Fig. 2. The first step is to generate an augmented image pyramid from each pair of images in a triplet (Sec. IV-A). The second step is determining which pairs of images observe a common structure, which is discussed in Sec. IV-B. Finally, these pairs will be used to instantiate a photometric optimization problem presented in Sec. IV-C. Without

using geometrical error functions or structure optimization, our approach is entirely free from any feature association. This makes our method simple, compatible with multiple depth sensors, and competitive with ad-hoc state-of-the-art approaches.

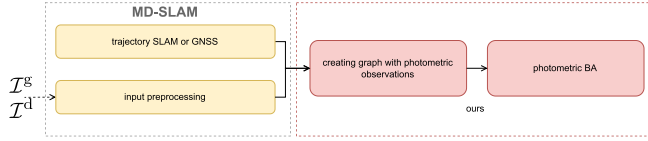


Fig. 2: The flow of our approach. The input of our system contains the initial guess of the sensor pose, the intensity image  $\mathcal{I}^g$  and the depth image  $\mathcal{I}^d$ . MD-SLAM already provides the input in the correct format since the preprocessing step is embedded in the SLAM system. The core of our refinement strategy consists in creating a graph with photometric observations (Sec. IV-B) and running a global optimization on the set of poses  $\mathbf{X}_{1:N}$ , as detailed in Sec. IV-C.

### A. Input Preprocessing

The input of our photometric refinement method is a pair of intensity  $\mathcal{I}^g$  and depth  $\mathcal{I}^d$  images for each sensor pose. The output of the preprocessing step is a five-channel image pyramid for each input pair. The first two channels of an image in the pyramid are intensity and depth, while the other three channels encode the surface normals. To calculate the normal at pixel  $\mathbf{u}$  we unproject the pixels in the neighborhood  $\mathcal{U} = \{\mathbf{u}' : \|\mathbf{u} - \mathbf{u}'\| < \tau_{\mathbf{u}}\}$  of a radius  $\tau_{\mathbf{u}}$  inversely proportional to the range at the pixel  $\mathcal{I}^d(\mathbf{u})$ . The normal  $\mathbf{n}_{\mathbf{u}}$  is the one of the plane that best fits the unprojected points from the set  $\mathcal{U}$ , and is oriented towards the observer. All valid normals are assembled in a normal image  $\mathcal{I}^n$ , so that  $\mathcal{I}^n(\mathbf{u}) = \mathbf{n}_{\mathbf{u}}$ . Hence, the final five-channel image is obtained by stacking together  $\mathcal{I}^g$ ,  $\mathcal{I}^d$ , and  $\mathcal{I}^n$ . In the remainder, we will refer to the generic channel as a *cue*  $\mathcal{I}^c$ .

Photometric approaches perform an implicit data association at a pixel level. Whereas attractive for their accuracy, these methods suffer from relatively small convergence basins that decrease with the image’s resolution: the higher the image resolution, the narrower the convergence basin will be. To lessen this effect, we generate multiple copies of the same image at decreasing resolution to form a pyramid. The optimization will proceed from the coarser to the finest level (Fig. 4).

Each level of a pyramid consists of a multi-cue image generated from  $\mathcal{I}^g$ ,  $\mathcal{I}^d$  and  $\mathcal{I}^n$ , by downscaling at user-selected resolutions. In our experiments, we typically use three scaling levels, each half the resolution of the previous level.

### B. Graph Construction

The input of our system is a set of triplets  $\langle \mathbf{X}_i, \mathcal{I}^g, \mathcal{I}^d \rangle$ , containing the initial guess of the sensor pose and the intensity/depth image pairs. To instantiate Eq. (9) we need to determine the matching pairs  $\langle i, j \rangle$ . The problem can be visualized as an undirected graph, where each node is a

triplet, and an edge between two nodes encodes a potential match.

To compute the matches, we start from the initial assignment of poses  $\{\mathbf{X}_n\}$  and add edges to the graph based on the input data. To this extent, we use a straightforward criterion that generates a matching pair if two poses  $\mathbf{X}_i, \mathbf{X}_j$  are close in space, and their orientations are similar. More specifically, we create a pair between  $\mathbf{X}_i, \mathbf{X}_j$  if all these conditions are satisfied:

- 1) the angle between the poses is below a threshold (typically 30 deg);
- 2) the translation between the poses is below a threshold (typically below 1 meter);
- 3) the ratio of reprojected valid points from  $\mathbf{X}_i$  onto  $\mathbf{X}_j$  is sufficiently high (typically 1/3).

In addition to these criteria, if the data come from a sequential acquisition, we add matches between subsequent triplets to model odometry-like constraints. An example of pose-pair associated is shown in Fig. 3.

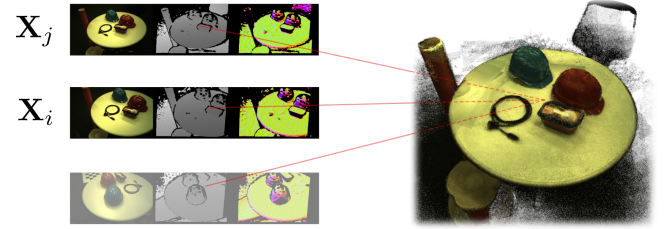


Fig. 3: An example of pose-pairs associated, our BA strategy relies on the association of  $\mathbf{X}_i$  and  $\mathbf{X}_j$  if they share observations. In the picture above, the input images with depth and normals are on the left, and on the right is the reconstructed model using our methodology. Note that in reality, inputs of our BA are pyramids (as discussed in Sec. IV-A, i.e., images of different resolutions). Here, we show just one level for simplicity. The data used is from ETH3D.

### C. Photometric Error Minimization

Our method seeks to find the set of transformations  $\mathbf{X}_{1:N}^* \in \mathbb{SE}(3)^N$  that minimizes the photometric error between each candidate pair of sensor poses that observe a common portion of the environment. Let  $\mathcal{I}_i$  and  $\mathcal{I}_j$  be two images acquired from poses  $\mathbf{X}_i$  and  $\mathbf{X}_j$  that form a matching pair. Let  $\mathcal{I}(\mathbf{u})$  be the value of the pixel  $\mathbf{u}$  in the image  $\mathcal{I}$ .

The photometric error at image coordinates  $\mathbf{u}$  in the matching pair is the difference between  $\mathcal{I}_i^g(\mathbf{u})$  and the pixel  $\mathcal{I}_j^g(\mathbf{u}')$  of the second image. The evaluation point  $\mathbf{u}'$  is computed by unprojecting the pixel  $\mathbf{u}$  from  $\mathcal{I}_i^g$  onto the image plane of  $\mathcal{I}_j^g$ . This accounts for the relative transform  $\mathbf{X}_{j,i} = \mathbf{X}_j^{-1}\mathbf{X}_i$  between the two frames, as follows:

$$\mathbf{u}' = \pi \left( \mathbf{R}_j^T \left( \mathbf{R}_i \pi^{-1}(\mathbf{u}, d) + \mathbf{t}_i - \mathbf{t}_j \right) \right). \quad (7)$$

To carry out this operation, the depth at the pixel  $d = \mathcal{I}^d(\mathbf{u})$  needs to be known. Standard photometric optimization seeks to find the following minimum:

$$\mathbf{X}_{1:N}^* = \operatorname{argmin}_{\mathbf{X}_{1:N}} \sum_{\langle i, j \rangle} \sum_{\mathbf{u}} \|\mathcal{I}_i^g(\mathbf{u}) - \mathcal{I}_j^g(\mathbf{u}')\|^2. \quad (8)$$

Here the inner summation computes the photometric error of a matching pair  $\langle i, j \rangle$  as the squared norm of the error of all pixels  $\mathbf{u}$  while the outer summation spans over all matching image pairs.

Eq. (8) models classical photometric error minimization assuming that the cues are unaffected by  $\mathbf{X}_{j,i}$ . Whereas this is true when operating with pure intensity/grayscale values, normals, and depths change when mapped from the frame  $\mathbf{X}_i$  to the frame  $\mathbf{X}_j$ . As in in [4] these mappings can be encapsulated by the function  $\zeta^c(\mathbf{X}_{j,i}, \mathcal{I}_i^c(\mathbf{u}))$  that calculates the *pixel* value of the  $c^{\text{th}}$  cue after applying the transform  $\mathbf{X}_{j,i}$  to the original channel value  $\mathcal{I}_i^c(\mathbf{u})$ . By extension, let  $\hat{\mathcal{I}}_i^c = \zeta^c(\mathbf{X}_{j,i}, \mathcal{I}_i^c)$  be the image obtained by remapping  $\mathcal{I}_i^c$ , according to  $\mathbf{X}_{j,i}$ . We can thus rewrite a more general form of Eq. (8) that accounts for all cues and captures this effect as follows:

$$F(\mathbf{X}_{1:N}) = \sum_{i,j} \sum_{\mathbf{u}} \rho \left\| \sum_c \hat{\mathcal{I}}_i^c(\mathbf{u}) - \mathcal{I}_j^c(\mathbf{u}') \right\|_{\Omega^c}^2 \quad (9)$$

$$\mathbf{X}_{1:N}^* = \underset{\mathbf{X}_{1:N}}{\operatorname{argmin}} F(\mathbf{X}_{1:N}) \quad (10)$$

where  $\rho$  is a Huber robust M-estimator. More details about Eq. (9) can be found in our supplementary material <sup>1</sup>. To carry on the minimization in Eq. (10) we employ the Levenberg-Marquardt algorithm implemented in the `srrg2_solver` [12]. At each iteration, we suppress the occluded portions of the images before evaluating Eq. (9). The optimization proceeds by seeking the optimum of all poses, starting from the coarser level. Once convergence is reached at one level, our system switches to the next finer one, and the optimization proceeds by choosing the solution computed so far as an initial guess. Fig. 4 shows the effect of this hierarchical approach applied to a BA problem. Generally, the worse the initial guess, the more the required levels.

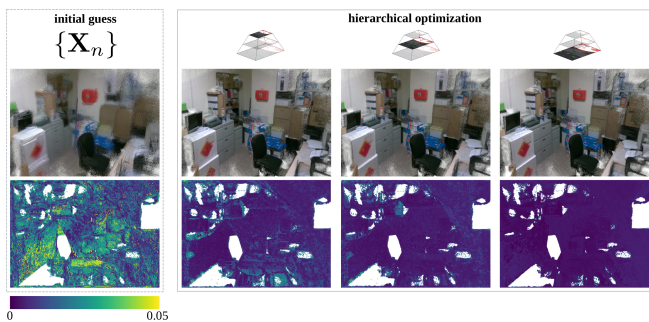


Fig. 4: The effect of hierarchical optimization when performing BA. From left to right, we show how the quality of the estimate improves, starting from a bad initial guess. The heatmap is normalized between 0 and 0.005 [m] for better visualization. The data is self-recorded using an Intel D455.

## V. EXPERIMENTAL EVALUATION

In this section, we report the results of our method on different public benchmark datasets. To the best of our knowledge, our approach is the only open-source photometric BA strategy that can deal with RGB-D and LiDAR in

a unified manner. Therefore, to evaluate our system, we compare it with state-of-the-art SLAM and BA packages developed specifically for each of these sensor types.

To run the experiments, we used a PC with an Intel Core i7-7700K CPU @ 4.20GHz, 32GB of RAM and a Zotac Geforce GTX 1070 X 8G. Our BA schema is implemented on the GPU using CUDA 11. Since this work is focused on global consistency, we perform our quantitative evaluation using the RMSE on the absolute trajectory error (ATE) with  $\text{SE}(3)$  alignment. The metric’s alignment is computed using the Horn method [13], and the timestamps are used to determine the associations. Then, we calculate the RMSE of the translational differences between all matched poses. In Sec. V-1, we discuss the approaches and the datasets used for comparison with RGB-D sensors, while in Sec. V-2 we present the results for LiDAR data. Since our implementation can run both on GPU and CPU, we report the runtimes of our algorithm for various pyramid resolutions using different commercial GPUs and our processor (Fig. 6). The image reports timings for each complete optimization iteration at different resolutions.

In our experiments, to switch from one level to the other, we use a simple termination criterion involving the variation of the error in Eq. (9) over the iterations. The number of iterations for each level differs from the quality of the initial guess. In our experiments, we typically observe successful around 10 iterations on coarser, 5 on the middle, and just a couple on the finest level.

1) *RGB-D*: As a public benchmark for RGB-D we used several sequences of ETH3D [20]. This dataset is acquired with global shutter cameras and accurate active stereo depth. Modeling rolling shutter effects and light changes cannot be encapsulated in the projection function  $\pi(\cdot)$ , our only pipeline component that differs between the RGB-D and LiDAR. For this reason, we restrict our comparison to the setting mentioned above.

The work proposed in this paper refines the map from a reasonable initial guess. We compute such initial configurations employing an improved online CUDA version of MD-SLAM [10], our previous SLAM system that unifies depth sensors through hierarchical photometric odometry estimation and feature-based loop-closures.

We compare different approaches representative of different classes of SLAM and BA algorithms specific for RGB-D sensors: DVO-SLAM [15], ElasticFusion [26], BundleFusion [5] BAD-SLAM [20], ORB-SLAM2 [18]. DVO SLAM implements a mixed geometry-based and direct registration. Internally the alignment between pairs of keyframes is obtained by jointly minimizing point-to-plane and photometric residuals. This is similar to ElasticFusion, whose estimate consists of a mesh model of the environment and the current sensor location instead of the trajectory. BundleFusion refines the global estimate by interleaving feature-based and photometric BA. Similar to us, their photometric refinement does not consider the structure, but only the sensor poses. This method highly depends on data association, employing correspondences based on sparse features and dense

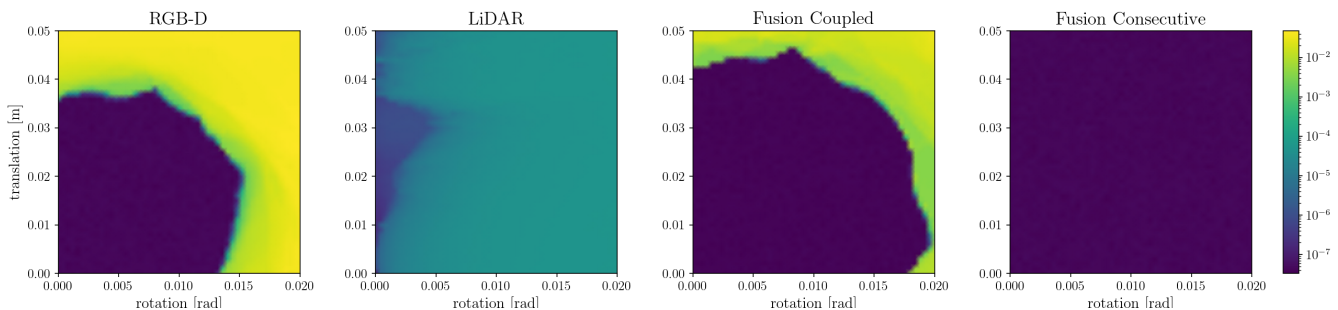


Fig. 5: Results of fusion experiments. Plots show how the initial perturbation affects the convergence basin and the algorithm’s accuracy. On the x-y axis, respectively, translation [m] and rotation [rad] perturbations, the value is denoted by the mean between rotation and translation error in log scale. Fusing the two sensors with our straightforward approach always shows better results compared to single sensors.

geometric/photometric matching. BAD-SLAM is a surfel-based direct Bundle Adjusted SLAM system that combines photometric and geometric errors alternating optimization of motion and structure. In contrast to these approaches, ORB-SLAM2 implements a traditional visual SLAM pipeline, where a local map of landmarks around the RGB-D sensor is constructed from ORB features [19]. The map is constantly optimized as the camera moves by performing local and global BA.

Most of the compared approaches run global refinement on a separate thread in an anytime fashion. The work presented in this paper addresses only this global aspect. Reporting the timings of this experiment would be unfair since our method addresses only a part of the problem.

ETH3D provides images of 740×460 pixels. We compute a 3-level pyramid from these images with scales 1/2, 1/4, and 1/8. In Tab. I, we can see that our photometric refinement performs on par (second after BAD-SLAM) with other state-of-the-art ad-hoc RGB-D SLAM and BA systems. Our method reduces the trajectory error to a few millimeters. More importantly, it improves by 60% the accuracy of the initial guess provided by MD-SLAM. Fig. 4 and Fig. 3 illustrate the effect of the hierarchical optimization on self-recorded data.

Summarizing, using our general pipeline of MD-SLAM and photometric BA presented in this paper provides results comparable with other RGB-D specific approaches, being second only to BAD-SLAM.

2) *3D LiDAR*: To validate our approach on LiDAR measurements we used both our data and public benchmarks. We used the Newer College Dataset [30] as a public benchmark. The dataset is recorded at 10 Hz with Ouster OS0-128. More specifically, we used the *cloister*, *quad* (easy), and *stairs* sequences. The *quad* sequence contains two loops that explore the Oxford campus courtyard, *cloister* mixes outdoor and indoor scenes while *stairs* captures an indoor scenario with multiple floors. Being based on image comparison, our approach operates well on LiDAR data having a good vertical resolution. LiDARs with fewer beams (i.e., 64, 32, 16) would produce an unbalanced horizontal image, reducing the converge basin of the algorithm.

We compare our method with BALM2 [17], which, to

	ElasticFusion	ORB-SLAM2	DVO-SLAM	BundleFusion	BAD-SLAM	MD-SLAM	MD-SLAM + Ours
table3	—	0.007	0.008	0.017	<b>0.002</b>	0.016	0.009
table4	0.012	0.008	0.018	—	<b>0.002</b>	0.023	0.008
table7	—	0.010	0.007	0.010	<b>0.003</b>	0.018	0.009
cables1	0.018	0.007	<b>0.004</b>	0.022	0.007	0.021	0.006
plant2	0.017	0.003	0.003	0.004	<b>0.001</b>	0.005	<b>0.001</b>
planar2	0.011	0.005	<b>0.002</b>	0.003	0.003	0.009	0.004
mean	0.014	0.007	0.007	0.011	0.003	0.015	0.006
std	0.003	0.002	0.005	0.008	0.002	0.007	0.003

TABLE I: ATE RMSE [m] on ETH3D benchmark, recorded with global shutter camera and synchronous streams. ElasticFusion fails in *table3* and *table7*, BundleFusion fails in *table4*.

the best of our knowledge, is the only publicly available LiDAR global refinement approach. BALM2 is based on the overall consistency of points, lines, and planes. This system requires the same input as our method, namely an initial guess trajectory and the point clouds.

To compute the initial guess, we used several SLAM algorithms specific for LiDAR: LeGO-LOAM [21], SuMA [2] and our unified MD-SLAM. LeGO-LOAM is a pure geometric feature-based frame-to-model LiDAR SLAM system, where the optimization on roll, yaw, and z-axis (pointing up) is decoupled from the planar parameters. SuMa constructs a surfel-based map and estimates the changes in the sensor’s pose by exploiting the projective data association in a frame-to-model or frame-to-frame fashion.

Tab. II reports the accuracy of our method and BALM2, for each dataset, and each initial guess. The ATE of the SLAM solution measures the quality of an initial guess. We observe that LeGO-LOAM provides a good guess on all planar data but fails on the *stairs* dataset resulting in an ATE of more than 3 meters. MD-SLAM performs reasonably well with a maximum ATE of 0.36 meters, while SuMA yields an acceptable initial guess only in the stairs dataset. If the initial guess is good, both BALM2 and our method perform well, improving the initial estimate. However, as the initial guess degrades, our unified global refinement’s accuracy remains

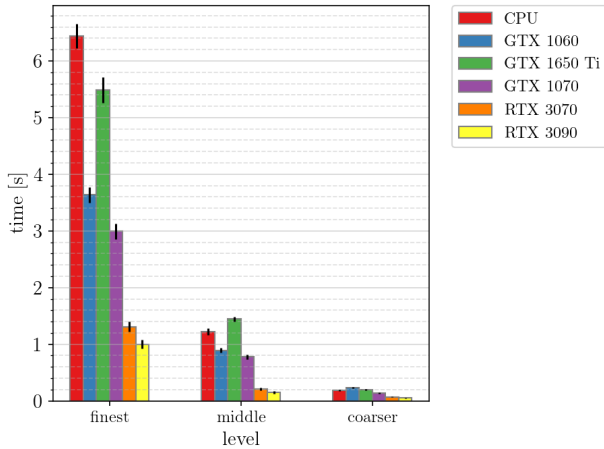


Fig. 6: Runtimes of our BA strategy evaluated on multiple commercial GPUs and our Intel Core i7-7700K CPU. Specifically, the value is represented by the cumulative time reached to complete an iteration during the optimization process, while the error bar indicates the standard deviation. In this experiment, the finest level includes around 86M pixels, the middle level includes 22M pixels, and the coarser level has around 5M pixels. Time is expressed in seconds.

stable by systematically improving the trajectory estimate. On these data, we observed BALM2 to be particularly sensible to rotations and less dense trajectories (i.e., trajectories sampled with fewer keyframe poses). Quantitative results show that our strategy is successful within LiDAR data, raising the initial estimate close to 60% improvement when a good initial guess is provided (i.e., *stairs* with MD-SLAM).

	LeGO-LOAM			MD-SLAM			SuMA		
	SLAM	BALM2	Ours	SLAM	BALM2	Ours	SLAM	BALM2	Ours
cloister	0.20	0.25	<b>0.16</b>	0.36	0.37	<b>0.32</b>	3.34	2.67	<b>2.53</b>
quad	<b>0.09</b>	<b>0.09</b>	<b>0.09</b>	0.25	1.98	<b>0.17</b>	1.74	1.72	<b>1.68</b>
stairs	<b>3.20</b>	5.13	3.48	0.23	0.38	<b>0.10</b>	0.67	0.86	<b>0.60</b>

TABLE II: ATE RMSE [m] on Newer College Dataset, recorded with OS0-128. We always improve the SLAM baseline, a part for *stairs* starting from LeGO-LOAM estimate.

The convergence basin of these global refinement strategies are bounded by inconsistency in the laser measurements (i.e., skewed point clouds); this is why none of the systems can further enhance the trajectory in the LeGO-LOAM *quad* experiment.

Fig. 1 shows our large-scale reconstruction of the historical part of Viterbo (Italy) from self-recorded data using a handheld Ouster OS0-128 LiDAR. The trajectory is about 2 km long, and the dataset is available from our repository <sup>1</sup>.

3) *RGB-D + 3D LiDAR*: Thanks to the unified nature of our pipeline, it allows the straightforward integration of multiple heterogeneous sensors. In Sec. IV-C, we formulated an optimization problem that estimates the sensor’s trajectory  $\mathbf{X}_{1:N}$ .

In a multiple sensors configuration, we can express the optimization problem as a function of the trajectory of

a multi-device platform, to which all sensors are rigidly attached. The multiple sensors setting slightly modify the cost function presented in Eq. (9) to include the constant sensor offsets. Our supplementary material <sup>1</sup> reports this extended formulation. The cost function for multi-sensor photometric refinement is just the sum of the cost functions of each device. In this section, we report an analysis of our optimization strategy with both RGB-D and LiDAR. For these experiments, we mounted an Intel D455 on the top of an OS0-128 and accurately calibrate the offset between the two sensors using a robust point-to-plane alignment [3]. We recorded some static indoor sequences, each of them containing synchronized grayscale and depth images from D455 and its corresponding intensity and range images from the OS0-128.

We carried out an experiment to evaluate the accuracy and converge basin of our approach in this configuration. To this extent, we perform our photometric alignment strategy using a single sensor frame consisting of a RGB-D and a LiDAR measurement pair. Since we align a sensor frame with respect to itself, we expect the computed estimate to be as close as possible to the identity. The optimization is carried on starting from increasingly wrong initial guesses.

In Fig. 5, we show the result of this experiment, starting from single sensor optimization as a baseline. Due to their different characteristics, the two sensors behave very differently during photometric alignment. The 3D LiDAR has an inferior resolution but a very large 360 deg horizontal FoV. This results in a relatively large convergence basin, but the limited resolution affects the minimum. Conversely, RGB-D, provides a high-resolution image with a much smaller FoV of around 70 deg. This results in better minima at the expense of a smaller converge basin.

We can refine the data in two ways that we name *coupled* and *consecutive*. The coupled approach aims to simultaneously finding the minimum of both photometric errors of RGB-D  $F^r(\mathbf{X}_{1:N})$  and LiDAR  $F^l(\mathbf{X}_{1:N})$ , as follows

$$\mathbf{X}_{1:N}^{\text{coupled}} = \underset{\mathbf{X}_{1:N}}{\operatorname{argmin}} F^r(\mathbf{X}_{1:N}) + F^l(\mathbf{X}_{1:N}) \quad (11)$$

The *consecutive* optimization combines the strengths of the two sensors, and operates by first carrying on an optimization where only the LiDAR is used. Subsequently, it uses the solution of this first run to find a minimum for the RGB-D problem.

The results suggest that fusing the two sensors always performs better than single sensors (Fig. 5). Specifically, *coupled* is generally more accurate compared to RGB-D only, having the converge basin incremented by the LiDAR. Similarly, *consecutive* seems to be the best since it takes full advantage of the large convergence basin of the LiDAR and benefits from the resolution of RGB-D. We believe that this proof of concept opens ways for 3D reconstruction algorithms that symmetrically fuse the two sensors, taking benefits from both.

## VI. CONCLUSION

In this paper, we presented a photometric BA methodology that operates both with RGB-D and LiDAR in a unified manner. To the best of our knowledge, our approach is the only open-source BA system that works independently for depth sensors and specifically exploits the photometric capabilities of the LiDAR image. Comparative experiments show that our simple schema performs on par or better compared to existing sensor-specific state-of-the-art approaches without making any assumption about the environment and free from data association. Thanks to our photometric registration methodology's inherent data separation, we develop our software in CUDA and release an open-source implementation. Considering the larger convergence basin and accuracy obtained fusing both RGB-D and LiDAR within our registration strategy, we envision a SLAM/BA pipeline that uses both depth sensors jointly. Furthermore, to complete our universal 3D reconstruction schema, future research would involve finding generic structure representation to optimize both RGB-D and LiDAR measurements.

## REFERENCES

- [1] H. Alismail, B. Browning, and S. Lucey. Photometric bundle adjustment for vision-based slam. In *Proc. of the Asian Conf. on Computer Vision (ACCV)*, pages 324–341. Springer, 2017.
- [2] J. Behley and C. Stachniss. Efficient surfel-based slam using 3d laser range data in urban environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.
- [3] Y. Chen and G. Medioni. Object modelling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155, 1992.
- [4] B. Della Corte, I. Bogoslavskyi, C. Stachniss, and G. Grisetti. A general framework for flexible multi-cue photometric point cloud registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 1–8, 2018.
- [5] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. on Graphics*, 36(4):1, 2017.
- [6] A. Delaunoy and M. Pollefeys. Photometric bundle adjustment for dense multi-view 3d modeling. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1486–1493, 2014.
- [7] N. Demmel, M. Gao, E. Laude, T. Wu, and D. Cremers. Distributed photometric bundle adjustment. In *Proc. of the International Conference on 3D Vision (3DV)*, pages 140–149. IEEE, 2020.
- [8] A. Eriksson, J. Bastian, T. Chin, and M. Isaksson. A consensus-based framework for distributed bundle adjustment. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1754–1762, 2016.
- [9] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Trans. on Robotics (TRO)*, 33(2):249–265, 2016.
- [10] L. Di Giammarino, L. Brizi, T. Guadagnino, C. Stachniss, and G. Grisetti. Md-slam: Multi-cue direct slam. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 11047–11054. IEEE, 2022.
- [11] B. Goldlücke, M. Aubry, K. Kolev, and D. Cremers. A super-resolution framework for high-accuracy multiview reconstruction. *Intl. Journal of Computer Vision (IJCV)*, 106:172–191, 2014.
- [12] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel. Least squares optimization: From theory to practice. *Robotics*, 9(3):51, 2020.
- [13] B. Horn, H. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988.
- [14] E. Olson J. Serafin and G. Grisetti. Fast and robust 3d feature extraction from sparse point clouds. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 4105–4112, 2016.
- [15] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2100–2106, 2013.
- [16] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007.
- [17] Z. Liu and F. Zhang. Balm: Bundle adjustment for lidar mapping. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3184–3191, 2021.
- [18] R. Mur-Artal and J. D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. on Robotics (TRO)*, 33(5):1255–1262, 2017.
- [19] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, pages 2564–2571, 2011.
- [20] T. Schops, T. Sattler, and M. Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 134–144, 2019.
- [21] T. Shan and B. Englot. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 4758–4765, 2018.
- [22] M. Slavcheva, W. Kehl, N. Navab, and S. Ilic. Sdf-2-sdf: Highly accurate 3d object reconstruction. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pages 680–696. Springer, 2016.
- [23] T. Stoyanov, M. Magnusson, H. Andreasson, and A. J. Lilienthal. Fast and accurate scan registration through minimization of the distance between compact 3d ndt representations. *Intl. Journal of Robotics Research (IJRR)*, 31(12):1377–1393, 2012.
- [24] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment - a modern synthesis. In *International Workshop on Vision Algorithms*, pages 298–372. Springer, 2000.
- [25] M. Velas, M. Spänel, and A. Herout. Collar line segments for fast odometry estimation from velodyne point clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 4486–4495, 2016.
- [26] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison. Elasticfusion: Dense slam without a pose graph. In *Proc. of Robotics: Science and Systems (RSS)*, 2015.
- [27] J. Zhang, M. Kaess, and S. Singh. On degeneracy of optimization-based state estimation problems. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 809–816, 2016.
- [28] J. Zhang and S. Singh. Loam: Lidar odometry and mapping in real-time. In *Proc. of Robotics: Science and Systems (RSS)*, 2014.
- [29] J. Zhang and S. Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pages 2174–2181, 2015.
- [30] L. Zhang, M. Camurri, and M. Fallon. Multi-camera lidar inertial extension to the newer college dataset. *arXiv*, 2112.08854. 2021.



SUPPLEMENTARY MATERIAL

In this section, we explain in detail our photometric error term presented in Sec. 9 and provide analytic expressions for the Jacobian matrices. In addition, we illustrate some additional trajectory plots related to the LiDAR experiments. We did not provide the same for RGB-D since trajectory errors are too low.

We represent as  $\Delta \mathbf{x}$  the Lie algebra  $\mathfrak{se}(3)$  associated with the group  $\mathbb{SE}(3)$ , parameterized as  $\Delta \mathbf{x} = [\Delta \mathbf{t}, \Delta \mathbf{q}]^T$ .  $\Delta \mathbf{t} \in \mathbb{R}^3$  is the translation, and  $\Delta \mathbf{q} \in \mathbb{R}^3$  is the imaginary part of a unit quaternion. The rotation matrix can be calculated from the perturbation vector using the *exponential map* at the identity  $\Delta \mathbf{R} = \exp(\Delta \mathbf{q})$ . We extend the notation of the exponential map to refer to the transformation encoded in  $\Delta \mathbf{x}$ . Let  $\Delta \mathbf{X} = \exp(\Delta \mathbf{x})$ , be this transformation whose rotation is  $\Delta \mathbf{R}$  and translation  $\Delta \mathbf{t}$ . We use the  $\boxplus$  operator to denote applying a perturbation to a transform  $\mathbf{X} \exp(\Delta \mathbf{x}) := \mathbf{X} \boxplus \Delta \mathbf{x}$ .

In the reminder, the error term differs slightly from the one presented in Eq. (9). Here we insert the offset mapping RGB-D or LiDAR with respect to the reference frame of a multi-device sensor platform. This is fundamental to fuse the two sensors as illustrated in Sec. V-3. The constant rigid transformation comprises rotation  $\mathbf{R}_o$  and translation  $\mathbf{t}_o$ .

For compactness, we define two quantities  $\mathbf{p}_u$  as the point transformed by this offset and  $\bar{\mathbf{p}}_u$  as the point  $\mathbf{p}_u$  transformed by the estimate and the inverse of the offset as follows:

$$\mathbf{p}_u = \mathbf{R}_o \pi^{-1}(\mathbf{u}, d) + \mathbf{t}_o, \quad (12)$$

$$\bar{\mathbf{p}}_u = \mathbf{R}_o^T (\mathbf{R}_j^T (\mathbf{R}_i \mathbf{p}_u + \mathbf{t}_i - \mathbf{t}_j) - \mathbf{t}_o). \quad (13)$$

Thus, we can rewrite our error in the following way:

$$\mathbf{e}_u^c = \zeta^c(\mathbf{X}_{j,i}, \mathcal{I}_i^c(\mathbf{u})) - \mathcal{I}_j^c(\mathbf{u}') = \zeta^c(\bar{\mathbf{p}}_u) - \mathcal{I}_j^c(\pi(\bar{\mathbf{p}}_u)). \quad (14)$$

Applying the perturbations  $\Delta \mathbf{x}_i$  and  $\Delta \mathbf{x}_j$  on the right hand side in Eq. (14) leads to:

$$\mathbf{e}_u^c(\mathbf{X}_i \boxplus \Delta \mathbf{x}_i) = \zeta^c(\bar{\mathbf{p}}_{u|i}) - \mathcal{I}_j^c(\pi(\bar{\mathbf{p}}_{u|i})), \quad (15)$$

$$\mathbf{e}_u^c(\mathbf{X}_j \boxplus \Delta \mathbf{x}_j) = \zeta^c(\bar{\mathbf{p}}_{u|j}) - \mathcal{I}_j^c(\pi(\bar{\mathbf{p}}_{u|j})). \quad (16)$$

with  $\bar{\mathbf{p}}_{u|i}$  and  $\bar{\mathbf{p}}_{u|j}$  respectively defined as:

$$\bar{\mathbf{p}}_{u|i} = \mathbf{R}_o^T (\mathbf{R}_j^T (\mathbf{R}_i (\mathbf{R}(\Delta \mathbf{q}) \mathbf{p}_u + \Delta \mathbf{t}) + \mathbf{t}_i - \mathbf{t}_j) - \mathbf{t}_o), \quad (17)$$

$$\bar{\mathbf{p}}_{u|j} = \mathbf{R}_o^T (\mathbf{R}(-\Delta \mathbf{q}) \mathbf{R}_j^T (\mathbf{R}_i \mathbf{p}_u + \mathbf{t}_i - \mathbf{t}_j) - \Delta \mathbf{t} - \mathbf{t}_o). \quad (18)$$

Deriving these two quantities with respect to the perturbations leads to the following Jacobians:

$$\frac{\partial \bar{\mathbf{p}}_{u|i}}{\partial \Delta \mathbf{x}_i} = \mathbf{R}_o^T \mathbf{R}_j^T \mathbf{R}_i [\mathbf{I}_{3 \times 3} \quad 2[\mathbf{p}_u]_{\times}], \quad (19)$$

$$\frac{\partial \bar{\mathbf{p}}_{u|j}}{\partial \Delta \mathbf{x}_j} = -\mathbf{R}_o^T [\mathbf{I}_{3 \times 3} \quad 2[\mathbf{R}_j^T (\mathbf{R}_i \mathbf{p}_u + \mathbf{t}_i - \mathbf{t}_j)]_{\times}]. \quad (20)$$

Projective Jacobians depends on the projection model, differing RGB-D and LiDAR, these are calculated respectively deriving Eq. (2) and Eq. (5) with respect to the transformed point  $\bar{\mathbf{p}}_u$ .

**Pinhole model:**

$$\frac{\partial \pi_p(\bar{\mathbf{p}}_u)}{\partial \bar{\mathbf{p}}_u} = \frac{\partial \phi(\mathbf{K} \bar{\mathbf{p}}_u)}{\partial \bar{\mathbf{p}}_u} = \frac{1}{v_z^2} \begin{bmatrix} v_z & 0 & -v_x \\ 0 & v_z & -v_y \end{bmatrix} \Big|_{[v_x, v_y, v_z] = \mathbf{K} \bar{\mathbf{p}}_u} \mathbf{K}. \quad (21)$$

**Spherical model:**

$$\frac{\partial \pi_s(\bar{\mathbf{p}}_u)}{\partial \bar{\mathbf{p}}_u} = \frac{\partial \mathbf{K}_{[1,2]} \psi(\bar{\mathbf{p}}_u)}{\partial \bar{\mathbf{p}}_u} = \mathbf{K}_{[1,2]} \left[ \begin{array}{c} \frac{1}{v_x^2 + v_y^2} \begin{bmatrix} -v_y & v_x & 0 \end{bmatrix} \\ \frac{1}{v_x^2 + v_y^2 + v_z^2} \begin{bmatrix} -\frac{v_x v_z}{\sqrt{v_x^2 + v_y^2}} & -\frac{v_y v_z}{\sqrt{v_x^2 + v_y^2}} & \sqrt{v_x^2 + v_y^2} \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} \end{array} \right] \Big|_{[v_x, v_y, v_z] = \bar{\mathbf{p}}_u}. \quad (22)$$

Image jacobians are numerically computed for each channel  $c$  with pixel-wise derivation:

$$\begin{aligned} \frac{\partial \mathcal{I}_{r,c}^c}{\partial r} &= \frac{1}{2} (\mathcal{I}_{r+1,c}^c - \mathcal{I}_{r-1,c}^c) \\ \frac{\partial \mathcal{I}_{r,c}^c}{\partial c} &= \frac{1}{2} (\mathcal{I}_{r,c+1}^c - \mathcal{I}_{r,c-1}^c) \end{aligned} \quad (23)$$

Jacobians on the mapping function  $\zeta^c$  for each channel grayscale/intensity, range/depth and normals, respectively  $\{g, d, n\}$  are computed also with respect to perturbations, since the estimates appears also to the left of the error function Eq. (14). In the reminder we specifically write mapping function and Jacobians of each cue.

### Intensity

The mapping function does not affect the intensity/grayscale channel, therefore we have:

$$\zeta^g(\bar{\mathbf{p}}_u) = \mathcal{I}^g(\bar{\mathbf{p}}_u), \quad \frac{\partial \zeta^g(\bar{\mathbf{p}}_{u|i})}{\Delta \mathbf{x}_i} = 0, \quad \frac{\partial \zeta^g(\bar{\mathbf{p}}_{u|j})}{\Delta \mathbf{x}_j} = 0. \quad (24)$$

We need to differ between the two depth sensor models, since RGB-D provides depth while LiDAR range measurements. Hence, leading to different Jacobians.

### Depth

$$\zeta^d(\bar{\mathbf{p}}_u) = [0 \ 0 \ 1] \bar{\mathbf{p}}_u, \quad \frac{\partial \zeta^d(\bar{\mathbf{p}}_{u|i})}{\Delta \mathbf{x}_i} = [0 \ 0 \ 1] \frac{\partial \bar{\mathbf{p}}_{u|i}}{\partial \Delta \mathbf{x}_i}, \quad \frac{\partial \zeta^d(\bar{\mathbf{p}}_{u|j})}{\Delta \mathbf{x}_j} = [0 \ 0 \ 1] \frac{\partial \bar{\mathbf{p}}_{u|j}}{\partial \Delta \mathbf{x}_j}. \quad (25)$$

### Range

$$\zeta^d(\bar{\mathbf{p}}_u) = \|\bar{\mathbf{p}}_u\|, \quad \frac{\partial \zeta^d(\bar{\mathbf{p}}_{u|i})}{\Delta \mathbf{x}_i} = \frac{\bar{\mathbf{p}}_u^T}{\|\bar{\mathbf{p}}_u\|} \frac{\partial \bar{\mathbf{p}}_{u|i}}{\partial \Delta \mathbf{x}_i}, \quad \frac{\partial \zeta^d(\bar{\mathbf{p}}_{u|j})}{\Delta \mathbf{x}_j} = \frac{\bar{\mathbf{p}}_u^T}{\|\bar{\mathbf{p}}_u\|} \frac{\partial \bar{\mathbf{p}}_{u|j}}{\partial \Delta \mathbf{x}_j}. \quad (26)$$

$$(27)$$

### Normals

Normals, differently, affect only rotation, hence is convenient to rewrite our error function expressed in Eq. (14). This reduces to:

$$\mathbf{e}_u^n = \zeta^n(\mathbf{R}_o^T \mathbf{R}_j^T \mathbf{R}_i \mathbf{R}_o \mathbf{n}_u) - \mathcal{I}^n(\pi(\bar{\mathbf{p}}_u)). \quad (28)$$

It is trivial to derive Eq. (14) with respect to the angular parts of the perturbations. Note that  $\mathbf{n}_u$  is the normal prior to any transformation, since everything is enrolled in Eq. (28).

$$\frac{\partial \zeta^n(\mathbf{R}_o^T \mathbf{R}_j^T \mathbf{R}_i \mathbf{R}_o \mathbf{n}_u)}{\Delta \mathbf{q}_i} = 2 \mathbf{R}_o^T \mathbf{R}_j \mathbf{R}_i \mathbf{R}_o [\mathbf{n}_u]_{\times}, \quad (29)$$

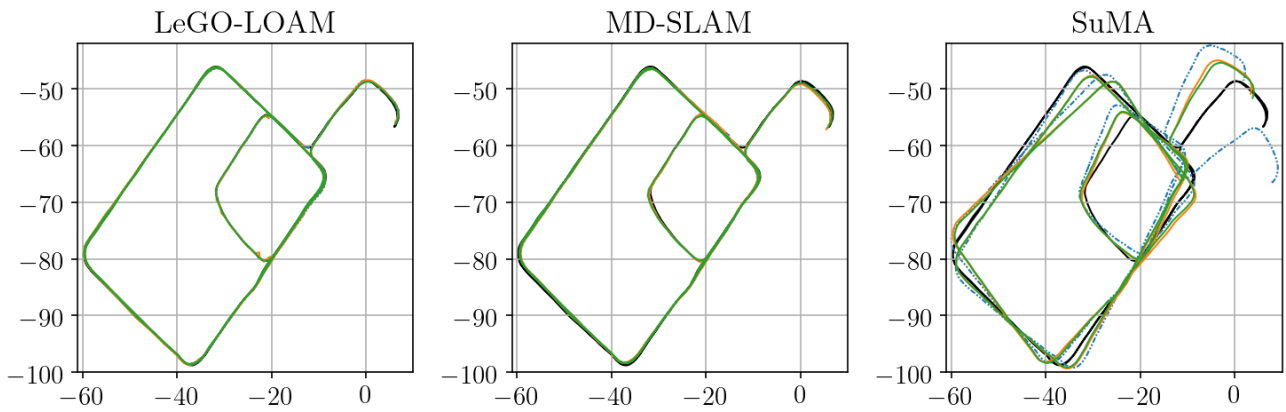
$$\frac{\partial \zeta^n(\mathbf{R}_o^T \mathbf{R}(-\Delta \mathbf{q}_j) \mathbf{R}_j^T \mathbf{R}_i \mathbf{R}_o \mathbf{n}_u)}{\Delta \mathbf{q}_j} = -2 \mathbf{R}_o^T [\mathbf{R}_j^T \mathbf{R}_i \mathbf{R}_o \mathbf{n}_u]_{\times}. \quad (30)$$

Finally, we can reconstruct the full jacobians:

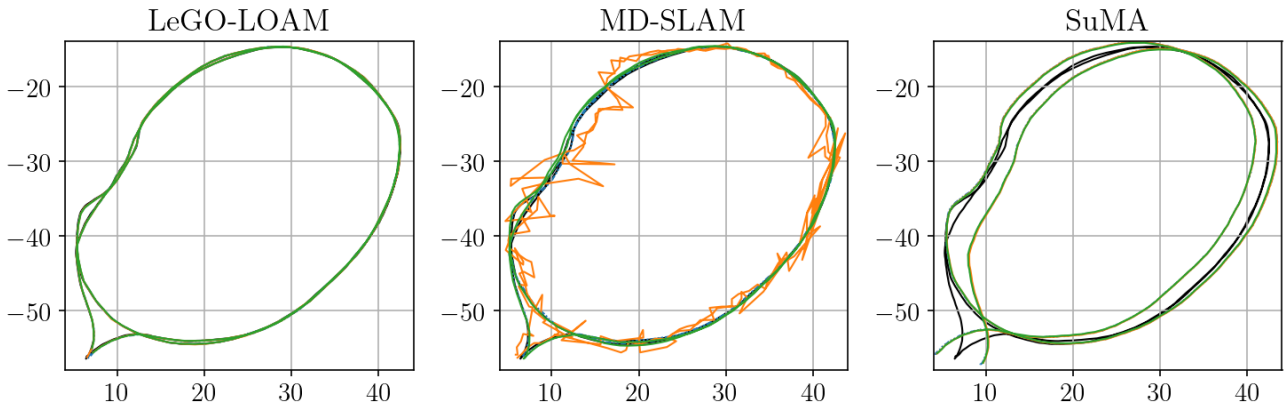
$$\mathbf{J}_i = \frac{\partial \zeta^c(\bar{\mathbf{p}}_{u|i})}{\Delta \mathbf{x}_i} - \frac{\partial \mathcal{I}_{r,c}^c}{\partial r, c} \frac{\partial \pi(\bar{\mathbf{p}}_u)}{\partial \bar{\mathbf{p}}_u} \frac{\partial \bar{\mathbf{p}}_{u|i}}{\partial \Delta \mathbf{x}_i}, \quad (31)$$

$$\mathbf{J}_j = \frac{\partial \zeta^c(\bar{\mathbf{p}}_{u|j})}{\Delta \mathbf{x}_j} - \frac{\partial \mathcal{I}_{r,c}^c}{\partial r, c} \frac{\partial \pi(\bar{\mathbf{p}}_u)}{\partial \bar{\mathbf{p}}_u} \frac{\partial \bar{\mathbf{p}}_{u|j}}{\partial \Delta \mathbf{x}_j}. \quad (32)$$

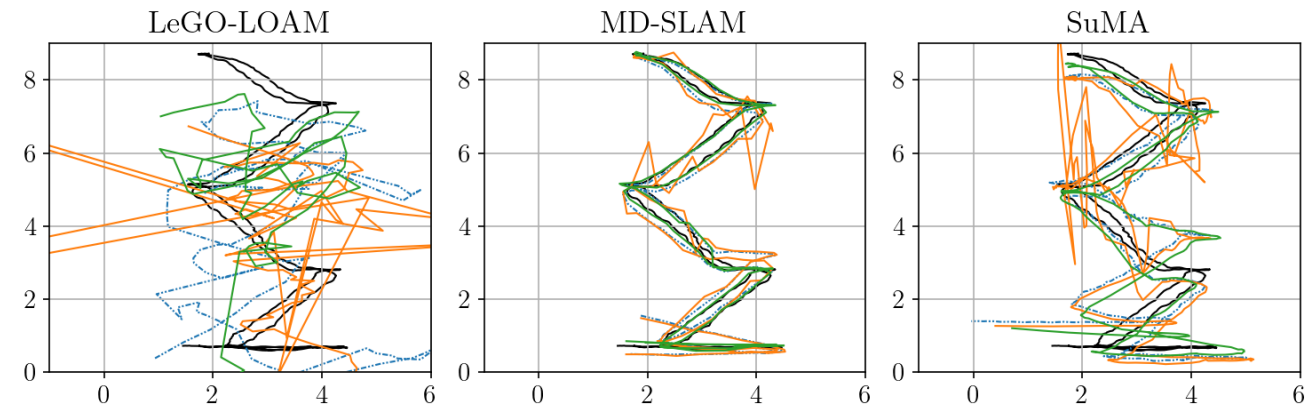
SLAM    BALM2    Ours    groundtruth



(a) Trajectory estimates of *cloister* sequence of Newer College Dataset. SLAM estimate in blue, BALM2 estimate in orange, Ours in green and groundtruth in black.



(b) Trajectory estimates of *quad* sequence of Newer College Dataset. SLAM estimate in blue, BALM2 estimate in orange, Ours in green and groundtruth in black. BALM2 suffer of less dense trajectories, since MD-SLAM spawn less keyframe poses to reduce odometry drift.



(c) Trajectory estimates of *stairs* sequence of Newer College Dataset. SLAM estimate in blue, BALM2 estimate in orange, Ours in green and groundtruth in black. BALM2 suffer of rotations, as it is observable from MD-SLAM and SuMA initial guess. For completeness we include the LeGO-LOAM plot, however both our and BALM2 fail due to bad initial guess.